

LPIC-1

Version 5.0
Français

102

Table of Contents

THÈME 105: SHELLS ET SCRIPTS SHELL	1
105.1 Personnalisation et utilisation de l'environnement du shell	2
105.1 Leçon 1	4
Introduction	4
Types de shell : interactif vs. non interactif et connexion vs. simple	5
Exercices guidés	19
Exercices d'approfondissement	21
Résumé	23
Réponses aux exercices guidés	25
Réponses aux exercices d'approfondissement	27
105.1 Leçon 2	29
Introduction	29
Variables : Affectation et référence	29
Variables locales ou variables du shell	33
Variables globales ou variables d'environnement	36
Exercices guidés	46
Exercices d'approfondissement	49
Résumé	51
Réponses aux exercices guidés	53
Réponses aux exercices d'approfondissement	58
105.1 Leçon 3	60
Introduction	60
Créer des alias	60
Créer des fonctions	65
Exercices guidés	75
Exercices d'approfondissement	78
Résumé	79
Réponses aux exercices guidés	81
Réponses aux exercices d'approfondissement	86
105.2 Personnalisation ou écriture de scripts simples	87
105.2 Leçon 1	89
Introduction	89
Structure et exécution d'un script	90
Les variables	92
Les expressions arithmétiques	96
L'exécution conditionnelle	96
Afficher les résultats d'un script	98
Exercices guidés	100

Exercices d'approfondissement	101
Résumé	102
Réponses aux exercices guidés	103
Réponses aux exercices d'approfondissement	104
105.2 Leçon 2	105
Introduction	105
Tests étendus	105
Les structures de boucles	111
Un exemple plus élaboré	113
Exercices guidés	117
Exercices d'approfondissement	119
Résumé	120
Réponses aux exercices guidés	121
Réponses aux exercices d'approfondissement	123
THÈME 106: INTERFACES ET BUREAUX UTILISATEUR	124
106.1 Installation et configuration de X11	125
106.1 Leçon 1	126
Introduction	126
Architecture du système X Window	127
Configuration du serveur X	130
Wayland	135
Exercices guidés	137
Exercices d'approfondissement	138
Résumé	139
Réponses aux exercices guidés	140
Réponses aux exercices d'approfondissement	141
106.2 Bureaux graphiques	142
106.2 Leçon 1	143
Introduction	143
Le système X Window	144
L'environnement de bureau	144
Environnements de bureau populaires	146
Interopérabilité des bureaux	148
Accès non local	150
Exercices guidés	152
Exercices d'approfondissement	153
Résumé	154
Réponses aux exercices guidés	155
Réponses aux exercices d'approfondissement	156
106.3 Accessibilité	157

106.3 Leçon 1	158
Introduction	158
Paramètres d'accessibilité	158
Assistance pour le clavier et la souris	159
Déficiences visuelles	161
Exercices guidés	163
Exercices d'approfondissement	164
Résumé	165
Réponses aux exercices guidés	166
Réponses aux exercices d'approfondissement	167
THÈME 107 : TÂCHES D'ADMINISTRATION	168
107.1 Gestion des comptes utilisateurs et des groupes ainsi que des fichiers systèmes concernés	169
107.1 Leçon 1	171
Introduction	171
Ajouter des comptes utilisateurs	171
Modifier les comptes utilisateurs	173
Supprimer des comptes utilisateurs	175
Ajouter, modifier et supprimer des groupes	175
Le répertoire squelette	176
Le fichier <code>/etc/login.defs</code>	177
La commande <code>passwd</code>	178
La commande <code>chage</code>	179
Exercices guidés	181
Exercices d'approfondissement	182
Résumé	183
Réponses aux exercices guidés	185
Réponses aux exercices d'approfondissement	187
107.1 Leçon 2	190
Introduction	190
<code>/etc/passwd</code>	191
<code>/etc/group</code>	192
<code>/etc/shadow</code>	192
<code>/etc/gshadow</code>	193
Filtrer les bases de données de mots de passe et des groupes	194
Exercices guidés	195
Exercices d'approfondissement	197
Résumé	198
Réponses aux exercices guidés	199
Réponses aux exercices d'approfondissement	201

107.2 Automate system administration tasks by scheduling jobs	203
107.2 Leçon 1	205
Introduction	205
Planifier des tâches avec Cron	205
Crontabs utilisateur	206
Crontabs système	207
Spécifications horaires particulières	208
Variables Crontab	208
Créer des tâches Cron utilisateurs	209
Créer des tâches Cron système	210
Configurer l'accès à la planification des tâches	211
Une alternative à Cron	211
Exercices guidés	214
Exercices d'approfondissement	216
Résumé	217
Réponses aux exercices guidés	218
Réponses aux exercices d'approfondissement	220
107.2 Leçon 2	222
Introduction	222
Planifier les tâches avec at	222
Afficher les tâches planifiées avec atq	224
Supprimer des tâches avec atrm	224
Configurer l'accès à la planification des tâches	225
Spécifications horaires	225
Une alternative à at	225
Exercices guidés	227
Exercices d'approfondissement	228
Résumé	229
Réponses aux exercices guidés	230
Réponses aux exercices d'approfondissement	231
107.3 Paramètres régionaux et langues	233
107.3 Leçon 1	235
Introduction	235
Les fuseaux horaires	236
L'heure d'été	240
Langue et encodage des caractères	241
Conversion de l'encodage	244
Exercices guidés	246
Exercices d'approfondissement	247
Résumé	248

Réponses aux exercices guidés	249
Réponses aux exercices d'approfondissement	250
THÈME 108: SERVICES SYSTÈMES ESSENTIELS	251
108.1 Gestion de l'horloge système	252
108.1 Leçon 1	254
Introduction	254
Heure locale et heure universelle	255
La date	255
Horloge matérielle	257
timedatectl	258
Régler le fuseau horaire sans timedatectl	260
Régler la date et l'heure sans timedatectl	261
Exercices guidés	263
Exercices d'approfondissement	265
Résumé	266
Réponses aux exercices guidés	268
Réponses aux exercices d'approfondissement	270
108.1 Leçon 2	271
Introduction	271
timedatectl	273
Le service NTP	274
Configuration NTP	275
pool.ntp.org	276
ntpdate	276
ntpq	276
chrony	277
Exercices guidés	282
Exercices d'approfondissement	284
Résumé	285
Réponses aux exercices guidés	286
Réponses aux exercices d'approfondissement	288
108.2 Journaux systèmes	289
108.2 Leçon 1	291
Introduction	291
Journalisation du système	292
Exercices guidés	312
Exercices d'approfondissement	314
Résumé	315
Réponses aux exercices guidés	316
Réponses aux exercices d'approfondissement	319

108.2 Leçon 2	320
Introduction	320
Bases de systemd	320
Le journal du système : systemd-journal	321
Exercices guidés	340
Exercices d'approfondissement	343
Résumé	344
Réponses aux exercices guidés	346
Réponses aux exercices d'approfondissement	349
108.3 Bases sur l'agent de transfert de courrier (MTA)	350
108.3 Leçon 1	351
Introduction	351
MTA local et distant	352
MTAs pour Linux	353
La commande mail et les applications clientes	358
Personnalisation de la livraison	360
Exercices guidés	362
Exercices d'approfondissement	363
Résumé	364
Réponses aux exercices guidés	365
Réponses aux exercices d'approfondissement	366
108.4 Gestion des imprimantes et de l'impression	367
108.4 Leçon 1	368
Introduction	368
Le service CUPS	369
Installer une imprimante	373
Gérer les imprimantes	376
Soumettre des tâches d'impression	377
Gérer les tâches d'impression	380
Supprimer des imprimantes	381
Exercices guidés	383
Exercices d'approfondissement	384
Résumé	385
Réponses aux exercices guidés	387
Réponses aux exercices d'approfondissement	388
THÈME 109: NOTIONS ÉLÉMENTAIRES SUR LES RÉSEAUX	390
109.1 Notions élémentaires sur les protocoles Internet	391
109.1 Leçon 1	392
Introduction	392
Protocole internet (IP <i>Internet Protocol</i>)	392

Exercices guidés	402
Exercices d'approfondissement	403
Résumé	404
Réponses aux exercices guidés	405
Réponses aux exercices d'approfondissement	406
109.1 Leçon 2	407
Introduction	407
Transmission Control Protocol (TCP)	409
User Datagram Protocol (UDP)	409
Internet Control Message Protocol (ICMP)	409
IPv6	410
Exercices guidés	413
Exercices d'approfondissement	414
Résumé	415
Réponses aux exercices guidés	416
Réponses aux exercices d'approfondissement	417
109.2 Configuration réseau persistante	418
109.2 Leçon 1	419
Introduction	419
L'interface réseau	419
Les noms des interfaces	421
Gestion des interfaces	422
Noms locaux et distants	424
Exercices guidés	429
Exercices d'approfondissement	430
Résumé	431
Réponses aux exercices guidés	432
Réponses aux exercices d'approfondissement	433
109.2 Leçon 2	434
Introduction	434
NetworkManager	434
systemd-networkd	439
Exercices guidés	442
Exercices d'approfondissement	443
Résumé	444
Réponses aux exercices guidés	445
Réponses aux exercices d'approfondissement	446
109.3 Résolution de problèmes réseaux simples	447
109.3 Leçon 1	449
Introduction	449

À propos de la commande <code>ip</code>	450
Aperçu du masque de réseau et du routage	451
Configurer une interface	452
La table de routage	454
Exercices guidés	458
Exercices d'approfondissement	459
Résumé	460
Réponses aux exercices guidés	461
Réponses aux exercices d'approfondissement	463
109.3 Leçon 2	465
Introduction	465
Tester les connexions avec <code>ping</code>	465
Retracer les itinéraires	466
Trouver les MTU avec <code>tracpath</code>	469
Créer des connexions arbitraires	470
Afficher les connexions en cours et les clients d'écoute	471
Exercices guidés	473
Exercices d'approfondissement	474
Résumé	475
Réponses aux exercices guidés	477
Réponses aux exercices d'approfondissement	479
109.4 Configuration de la résolution de noms	481
109.4 Leçon 1	482
Introduction	482
Le processus de résolution des noms	482
Les classes DNS	483
Les outils de résolution de noms	486
Exercices guidés	492
Exercices d'approfondissement	493
Résumé	494
Réponses aux exercices guidés	495
Réponses aux exercices d'approfondissement	496
THÈME 110 : SECURITÉ	498
110.1 Tâches d'administration de sécurité	499
110.1 Leçon 1	501
Introduction	501
Rechercher les fichiers avec les permissions SUID et SGID	501
Gestion et expiration des mots de passe	504
Dépistage des ports ouverts	508
Limiter les connexions des utilisateurs, les processus et l'utilisation de la mémoire	514

Gérer les utilisateurs connectés	517
Configuration et utilisation de base de <code>sudo</code>	520
Exercices guidés	526
Exercices d'approfondissement	530
Résumé	531
Réponses aux exercices guidés	533
Réponses aux exercices d'approfondissement	537
110.2 Configuration de la sécurité du système	538
110.2 Leçon 1	539
Introduction	539
Améliorer la sécurité de l'authentification avec les mots de passe cachés	539
Utiliser un super-démon pour surveiller les connexions réseau entrantes	541
Vérifier les services pour détecter les démons inutiles	546
Les TCP Wrappers comme une sorte de pare-feu simple	548
Exercices guidés	550
Exercices d'approfondissement	551
Résumé	552
Réponses aux exercices guidés	554
Réponses aux exercices d'approfondissement	555
110.3 Sécurisation des données avec le chiffrement	556
110.3 Leçon 1	558
Introduction	558
Configuration et utilisation de base du client OpenSSH	559
Fonction des clés d'hôte du serveur OpenSSH	564
Tunnels de ports SSH	566
Exercices guidés	571
Exercices d'approfondissement	573
Résumé	574
Réponses aux exercices guidés	575
Réponses aux exercices d'approfondissement	578
110.3 Leçon 2	579
Introduction	579
Configuration de base de GnuPG, utilisation et révocation	579
Utiliser GPG pour chiffrer, déchiffrer, signer et vérifier des fichiers	585
Exercices guidés	591
Exercices d'approfondissement	593
Résumé	594
Réponses aux exercices guidés	595
Réponses aux exercices d'approfondissement	597
Impression	598



**Linux
Professional
Institute**

Thème 105: Shells et scripts Shell



105.1 Personnalisation et utilisation de l'environnement du shell

Référence aux objectifs de LPI

LPIC-1 version 5.0, Exam 102, Objective 105.1

Valeur

4

Domaines de connaissance les plus importants

- Définition des variables environnement (par exemple le PATH) utilisées lors de la connexion ou au lancement d'un nouveau shell.
- Réalisation de fonctions BASH pour des séquences de commandes fréquentes.
- Mise à jour des répertoires squelette pour les nouveaux comptes utilisateurs.
- Définition correcte de la liste des chemins d'accès pour les commandes.

Partial list of the used files, terms and utilities

- `.`
- `source`
- `/etc/bash.bashrc`
- `/etc/profile`
- `env`
- `export`
- `set`
- `unset`
- `~/.bash_profile`
- `~/.bash_login`

- `~/.profile`
- `~/.bashrc`
- `~/.bash_logout`
- `function`
- `alias`



105.1 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	105 Shells et scripts shell
Objectif :	105.1 Personnaliser et utiliser l'environnement shell
Leçon :	1 sur 3

Introduction

L'interpréteur de commandes (*shell*) est probablement l'outil le plus puissant d'un système Linux. Il peut être défini comme une interface entre l'utilisateur et le cœur du système d'exploitation. Le shell interprète les commandes saisies par l'utilisateur. C'est pourquoi tous les administrateurs système doivent savoir utiliser l'interpréteur de commandes. Comme nous le savons sans doute déjà, le Bourne Again Shell (*Bash*) est l'interpréteur de commandes utilisé par défaut par la grande majorité des distributions Linux.

Une fois qu'il est lancé, la première chose que fait Bash — tout comme n'importe quel autre shell d'ailleurs — c'est d'exécuter une série de scripts de démarrage. Ces scripts permettent de personnaliser l'environnement de la session. Certains scripts concernent la totalité du système alors que d'autres sont propres à l'utilisateur. Nous pouvons y intégrer nos préférences personnelles et les paramètres adaptés aux besoins des utilisateurs sous forme de variables, d'alias et de fonctions.

La séquence exacte de fichiers de démarrage dépend d'un paramètre très important : le type de shell. Jetons un œil aux différents types de shells qui existent.

Types de shell : interactif vs. non interactif et connexion vs. simple

Pour commencer, nous allons mettre au clair les concepts *interactif* et *connexion* dans le contexte du shell :

Shells interactifs / non interactifs

Ce type de shell fait référence à l'interaction qui a lieu entre l'utilisateur et le shell. L'utilisateur fournit des données en tapant des commandes dans le terminal à l'aide du clavier ; l'interpréteur de commandes fournit des résultats en affichant des messages à l'écran.

Shells de connexion / simples

Ce type de shell fait référence au cas de figure où un utilisateur accède à un système informatique en fournissant son identifiant et son mot de passe.

Les shells interactifs et non interactifs peuvent être des shells de connexion ou simples, et toute combinaison possible de ces types de shell a ses utilisations spécifiques.

Les *shells de connexion interactifs* sont lancés lorsque les utilisateurs se connectent au système. On les utilise pour personnaliser les configurations des utilisateurs en fonction de leurs besoins. Un bon exemple de ce type de shell serait celui d'un groupe d'utilisateurs appartenant au même service et qui ont besoin d'une variable particulière dans leurs sessions.

Les *shells interactifs simples* font référence à tous les autres shells ouverts par l'utilisateur une fois qu'il est connecté au système. Les utilisateurs se servent de ces interpréteurs de commandes pendant les sessions pour effectuer des tâches d'administration et de maintenance le réglage de l'heure, la définition de variables, la copie de fichiers, l'écriture de scripts, etc.

D'autre part, les *shells non-interactifs* ne requièrent aucune interaction humaine. Ces interpréteurs de commandes ne demandent aucune entrée à l'utilisateur et leur sortie — s'il y en a une — est généralement écrite dans un journal.

Les *shells de connexion non interactifs* sont aussi rares que peu commodes. Leur utilisation est marginale et nous ne les abordons ici que pour mieux comprendre le comportement de l'interpréteur de commandes. Quelques exemples insolites incluent le fait de forcer l'exécution d'un script à partir d'un shell de connexion avec `/bin/bash --login <some_script>` ou de faire passer la sortie standard (*stdout*) d'une commande dans l'entrée standard (*stdin*) d'une connexion ssh :

```
<some_command> | ssh <some_user>@<some_server>
```

Quant au *shell simple non interactif*, il n’y a ni connexion ni interaction de la part de l’utilisateur, et nous faisons référence ici à l’utilisation de scripts automatisés. Ces scripts sont utilisés principalement pour effectuer des tâches administratives répétitives comme celles que l’on trouve habituellement dans les *cronjobs*. Dans ce cas, `bash` ne lit aucun fichier de démarrage.

Ouvrir un terminal

Lorsque nous sommes dans un environnement de bureau, nous pouvons ouvrir une application de terminal ou basculer vers l’une des consoles du système. De ce fait, un nouveau shell est soit un shell `pts` lorsqu’il est ouvert à partir d’un émulateur de terminal dans l’interface graphique, soit un shell `tty` lorsqu’il est exécuté à partir d’une console système. Dans le premier cas, il ne s’agit pas d’un terminal mais d’un émulateur de terminal. Dans le contexte d’une session graphique, les émulateurs de terminal tels que *gnome-terminal* ou *konsole* sont riches en fonctionnalités et conviviaux par rapport aux terminaux de type texte. Parmi les émulateurs de terminal moins complets, on peut mentionner — entre autres — *XTerm* et *sakura*.

Les raccourcis clavier `Ctrl + Alt + F1-F6` permettent d’accéder aux consoles qui ouvrent un shell de connexion interactif en mode texte. `Ctrl + Alt + F7` ramène la session sur le bureau.

NOTE `tty` signifie téléscripteur (*teletypewriter*); `pts` représente un pseudo-terminal secondaire (*pseudo terminal slave*). Pour plus d’informations : `man tty` et `man pts`.

Lancer un shell avec `bash`

Une fois que vous vous êtes connecté, tapez `bash` dans un terminal pour lancer un nouvel interpréteur de commandes. Techniquement, ce shell est un processus enfant du shell en cours.

Lorsqu’on démarre le processus enfant `bash`, on peut utiliser plusieurs options pour définir le type d’interpréteur de commandes que l’on veut initier. Voici quelques options fondamentales pour invoquer `bash` :

`bash -l` ou `bash --login`

invoque un shell de connexion.

`bash -i`

invoque un shell interactif.

`bash --noprofile`

avec les shells de connexion ignorera à la fois le fichier de démarrage du système `/etc/profile` et les fichiers de démarrage au niveau utilisateur `~/.bash_profile`, `~/.bash_login` et `~/.profile`.

bash --norc

avec les shells interactifs ignorera à la fois le fichier de démarrage du système `/etc/bash.bashrc` et le fichier de démarrage de l'utilisateur `~/.bashrc`.

bash --rcfile <file>

avec les shells interactifs utilisera `<fichier>` comme fichier de démarrage en ignorant `/etc/bash.bashrc` au niveau système et `~/.bashrc` au niveau utilisateur.

Les différents fichiers de démarrage seront abordés ci-dessous.

Lancer un shell avec su et sudo

Ces deux programmes similaires permettent d'obtenir des types de shells spécifiques :

su

Changer l'ID de l'utilisateur ou devenir superutilisateur (`root`). Cette commande permet d'invoquer les shells de connexion et des shells simples :

- `su - user2`, `su -l user2` ou `su --login user2` lance un shell de connexion interactif en tant que `user2`.
- `su user2` lance un shell interactif simple en tant que `user2`.
- `su - root` ou `su -` lance un shell de connexion interactif en tant que `root`.
- `su root` ou `su` lance un shell interactif simple en tant que `root`.

sudo

Exécuter des commandes sous une autre identité (y compris celle du super-utilisateur). Étant donné que cette commande est principalement utilisée pour obtenir temporairement les privilèges de `root`, l'utilisateur qui y a recours doit figurer dans le fichier `sudoers`. Pour ajouter des utilisateurs à `sudoers`, nous devons devenir `root` et ensuite exécuter :

```
root@debian:~# usermod -aG sudo user2
```

Tout comme `su`, `sudo` nous permet d'invoquer des shells de connexion ou des shells simples :

- `sudo su - user2`, `sudo su -l user2` ou `sudo su --login user2` lance un shell interactif de connexion en tant que `user2`.
- `sudo su user2` lance un shell interactif simple en tant que `user2`.
- `sudo -u user2 -s` lance un shell interactif simple en tant que `user2`.

- `sudo su - root` ou `sudo su -` lance un shell interactif de connexion en tant que `root`.
- `sudo -i` lance un shell interactif de connexion en tant que `root`.
- `sudo -i <some_command>` lance un shell interactif de connexion en tant que `root`, exécute la commande et retourne à l'utilisateur initial.
- `sudo su root` ou `sudo su` lance un shell interactif simple en tant que `root`.
- `sudo -s` ou `sudo -u root -s` lance un shell simple en tant que `root`.

Lorsque nous utilisons `su` ou `sudo`, nous devons prendre en compte le contexte dans lequel nous démarrons un nouvel interpréteur de commandes : Est-ce que nous avons besoin de l'environnement de l'utilisateur cible ou non ? Si c'est le cas, nous utiliserons les options qui invoquent un shell de connexion ; dans le cas contraire, nous utiliserons celles qui invoquent un shell simple.

Quel type de shell avons-nous ?

Pour connaître le type de shell dans lequel nous travaillons, nous pouvons taper `echo $0` dans le terminal et obtenir ce qui suit :

Shell de connexion interactif

`-bash` ou `-su`

Shell interactif simple

`bash` or `/bin/bash`

Shell simple non-interactif (script)

`<nom_du_script>`

Combien de shells avons-nous ?

Pour voir combien de shells `bash` sont lancés sur le système, nous pouvons utiliser la commande `ps aux | grep bash` :

```
user2@debian:~$ ps aux | grep bash
user2      5270  0.1  0.1 25532  5664 pts/0    Ss   23:03   0:00 bash
user2      5411  0.3  0.1 25608  5268 tty1      S+   23:03   0:00 -bash
user2      5452  0.0  0.0 16760   940 pts/0    S+   23:04   0:00 grep --color=auto bash
```

`user2` sur le hôte `debian` s'est connecté à une session graphique (GUI ou *X Window System*) et a ouvert *gnome-terminal*. Puis il a appuyé sur `Ctrl` + `Alt` + `F1` pour ouvrir une session terminal `tty`.

Enfin, il est retourné dans la session graphique en appuyant sur `Ctrl + Alt + F7` et a tapé la commande `ps aux | grep bash`. Ainsi, l'affichage montre un shell interactif simple via l'émulateur de terminal (`pts/0`) et un shell de connexion interactif via la console de texte (`tty1`). Notez aussi que la dernière colonne de chaque ligne (la commande) est `bash` pour le premier cas de figure et `-bash` pour le second.

D'où les shells tirent leur configuration : les fichiers de démarrage

Maintenant que nous connaissons les différents types de shells que l'on peut trouver dans un système Linux, il est temps de voir les fichiers de démarrage propres à chaque type de shell. Notez que les scripts système ou globaux sont rangés dans le répertoire `/etc/` alors que les scripts locaux ou propres à l'utilisateur se trouvent dans le répertoire personnel de l'utilisateur (`~`). De plus, lorsqu'il y a plus d'un fichier à rechercher, une fois que l'un d'entre eux est trouvé et exécuté, les autres sont ignorés. Explorez et étudiez ces fichiers vous-même avec votre éditeur de texte préféré ou en tapant `less <startup_file>`.

NOTE

Les fichiers de démarrage peuvent être classés en fichiers spécifiques à Bash (ceux qui sont limités uniquement aux configurations et commandes `bash`) et en fichiers génériques (concernant la plupart des interpréteurs de commandes).

Shell de connexion interactif

Niveau global

`/etc/profile`

C'est le fichier `.profile` du système pour le shell Bourne et les shells compatibles Bourne (y compris `bash`). Au moyen d'une série d'instructions `if`, ce fichier définit un certain nombre de variables telles que `PATH` et `PS1` et prend en compte—s'ils existent—le fichier `/etc/bash.bashrc` et ceux du répertoire `/etc/profile.d`.

`/etc/profile.d/*`

Ce répertoire est susceptible de contenir des scripts qui seront exécutés par `/etc/profile`.

Niveau local

`~/.bash_profile`

Ce fichier spécifique à Bash est utilisé pour configurer l'environnement utilisateur. Il peut également être utilisé pour prendre en compte `~/.bash_login` et `~/.profile`.

`~/.bash_login`

Également spécifique à Bash, ce fichier ne sera exécuté qu'en l'absence de fichier

`~/.bash_profile`. Son nom suggère qu'il est utilisé pour exécuter les commandes nécessaires à l'ouverture d'une session.

`~/.profile`

Ce fichier n'est pas spécifique à Bash et n'est utilisé que si les fichiers `~/.bash_profile` et `~/.bash_login` n'existent pas — ce qui est normalement le cas. Ainsi, le rôle principal de `~/.profile` consiste à vérifier si un shell Bash est en cours d'exécution et — si c'est le cas — à prendre en compte `~/.bashrc` s'il existe. Il définit habituellement la variable `PATH` de manière à inclure le répertoire privé `~/bin` de l'utilisateur s'il existe.

`~/.bash_logout`

S'il existe, ce fichier spécifique à Bash effectue des opérations de nettoyage lors de la fermeture du shell. Cela peut s'avérer pratique pour les sessions distantes, par exemple.

Explorer les fichiers de configuration du shell de connexion interactif

Voyons quelques-uns de ces fichiers en action en modifiant `/etc/profile` et `/home/user2/.profile`. Nous ajoutons à chacun d'eux une ligne qui va signaler le fichier en cours d'exécution :

```
root@debian:~# echo 'echo Hello from /etc/profile' >> /etc/profile
root@debian:~# echo 'echo Hello from ~/.profile' >> /home/user2/.profile
```

NOTE

Deux chevrons de redirection `>>` ajoutent le résultat d'une commande à un fichier existant, sans l'écraser. Si le fichier n'existe pas, il sera créé.

Ainsi, grâce à l'affichage des commandes `echo` respectives, nous saurons à quel moment chacun de ces fichiers est lu et exécuté. Pour le vérifier, voyons ce qui se passe lorsque `user2` se connecte via `ssh` depuis une autre machine :

```
user2@debian:~$ ssh user2@192.168.1.6
user2@192.168.1.6's password:
Linux debian 4.9.0-8-amd64 #1 SMP Debian 4.9.130-2 (2018-10-27) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Nov 27 19:57:19 2018 from 192.168.1.10
```



```
Hello from /etc/profile
Hello from /home/user2/.profile
```

Comme on peut le voir dans les deux dernières lignes, ça a marché. Par ailleurs, notez trois détails :

- Le fichier global a été exécuté en premier.
- Il n'y a pas de fichiers `.bash_profile` ou `.bash_login` dans le répertoire personnel de `user2`.
- Le tilde (`~`) a été développé pour indiquer le chemin complet du fichier (`/home/user2/.profile`).

Shell interactif simple

Niveau global

`/etc/bash.bashrc`

C'est le fichier `.bashrc` du système pour les shells `bash` interactifs. En l'exécutant, `bash` s'assure qu'il est lancé de manière interactive, vérifie la taille de la fenêtre après chaque commande (en mettant à jour les valeurs de `LINES` et `COLUMNS` si nécessaire) et définit quelques variables.

Niveau local

`~/ .bashrc`

En dehors des tâches similaires à celles décrites pour `/etc/bash.bashrc` au niveau de l'utilisateur (comme la vérification de la taille de la fenêtre ou de l'exécution interactive), ce fichier spécifique à Bash définit habituellement quelques variables de l'historique et prend en compte `~/ .bash_aliases` s'il existe. En dehors de cela, ce fichier est normalement utilisé pour stocker les alias et les fonctions spécifiques de l'utilisateur.

De même, il est intéressant de noter que `~/ .bashrc` est lu si `bash` détecte que son entrée standard (`<stdin>`) est une connexion réseau, comme c'était le cas avec la connexion SSH dans l'exemple ci-dessus.

Explorer les fichiers de configuration du shell interactif simple

À présent, modifions `/etc/bash.bashrc` et `/home/user2/.bashrc` :

```
root@debian:~# echo 'echo Hello from /etc/bash.bashrc' >> /etc/bash.bashrc
root@debian:~# echo 'echo Hello from ~/.bashrc' >> /home/user2/.bashrc
```

Voici ce qui se passe lorsque `user2` démarre un nouveau shell :

```
user2@debian:~$ bash
Hello from /etc/bash.bashrc
Hello from /home/user2/.bashrc
```

Là encore, les deux fichiers ont été lus et exécutés.

WARNING

N'oubliez pas qu'en raison de l'ordre d'exécution des fichiers, les fichiers locaux ont la primauté sur les fichiers globaux.

Shell de connexion non-interactif

Un shell non-interactif avec les options `-l` ou `--login` est forcé de se comporter comme un shell de connexion. Les fichiers de démarrage à exécuter seront donc les mêmes que ceux du shell de connexion interactif.

Pour le prouver, écrivons un simple script shell et rendons-le exécutable. Nous n'allons pas inclure de *shebang* étant donné que nous allons invoquer l'exécutable `bash` (`/bin/bash` avec l'option `login`) depuis la ligne de commande.

1. On crée le script `test.sh` avec la ligne `echo 'Hello from a script'` pour prouver que le script s'exécute correctement :

```
user2@debian:~$ echo "echo 'Hello from a script'" > test.sh
```

2. On rend notre script exécutable :

```
user2@debian:~$ chmod +x ./test.sh
```

3. Enfin, on invoque `bash` avec l'option `-l` pour exécuter le script :

```
user2@debian:~$ bash -l ./test.sh
Hello from /etc/profile
Hello from /home/user2/.profile
Hello from a script
```

Ça marche ! Avant d'exécuter le script, la connexion a eu lieu et les fichiers `/etc/profile` et `~/.profile` ont été exécutés.

NOTE

Nous aborderons le *shebang* et d'autres aspects des scripts shell dans les leçons à venir.

Nous allons maintenant faire passer la sortie standard (*stdout*) de la commande `echo` vers l'entrée standard (*stdin*) d'une connexion `ssh` à l'aide d'un pipe (`|`) :

```
user2@debian:~$ echo "Hello-from-a-noninteractive-login-shell" | ssh user2@192.168.1.6
Pseudo-terminal will not be allocated because stdin is not a terminal.
user2@192.168.1.6's password:
Linux debian 4.9.0-8-amd64 #1 SMP Debian 4.9.130-2 (2018-10-27) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Hello from /etc/profile
Hello from /home/user2/.profile
-bash: line 1: Hello-from-a-noninteractive-login-shell: command not found
```

Là encore, `/etc/profile` et `~/.profile` sont exécutés. À part ça, la première et la dernière ligne du résultat sont assez révélatrices du comportement du shell.

Shell simple non-interactif

Les scripts ne lisent aucun des fichiers mentionnés ci-dessus. Ils recherchent la variable d'environnement `BASH_ENV`, développent sa valeur si nécessaire et l'utilisent comme nom d'un fichier de démarrage pour lire et exécuter des commandes. Nous verrons plus en détail les *variables d'environnement* dans la prochaine leçon.

Comme nous l'avons vu plus haut, `/etc/profile` et `~/.profile` s'assurent que `/etc/bash.bashrc` et `~/.bashrc` sont exécutés après une connexion réussie. Le résultat de la commande suivante illustre ce comportement :

```
root@debian:~# su - user2
Hello from /etc/bash.bashrc
Hello from /etc/profile
Hello from /home/user2/.bashrc
Hello from /home/user2/.profile
```

Si l'on garde à l'esprit les lignes que nous avons précédemment ajoutées aux scripts de démarrage — et si l'on invoque un shell de connexion interactif au niveau utilisateur avec `su - user2` — les quatre lignes de résultat peuvent être expliquées de la sorte :

1. `Hello from /etc/bash.bashrc` signifie que `/etc/profile` a pris en compte `/etc/bash.bashrc`.
2. `Hello from /etc/profile` signifie que `/etc/profile` a été entièrement lu et exécuté.
3. `Hello from /home/user2/.bashrc` signifie que `~/.profile` a pris en compte `~/.bashrc`.
4. `Hello from /home/user2/.profile` signifie que `~/.profile` a été entièrement lu et exécuté.

Notez qu'avec `su - <username>` (ainsi que `su -l <username>` et `su --login <username>`) nous garantissons l'invocation d'un shell de connexion, alors que `su <username>` n'aurait invoqué que `/etc/bash.bashrc` et `~/.bashrc`.

Sourcer des fichiers

Dans les sections précédentes, nous avons vu que certains scripts de démarrage incluent ou exécutent d'autres scripts. Ce mécanisme de prise en compte, appelé *sourcing*, est présenté dans cette section.

Sourcer des fichiers avec `.`

Le point (`.`) se trouve normalement dans les fichiers de démarrage.

Dans le fichier `.profile` de notre serveur Debian, nous pouvons trouver — par exemple — le bloc suivant :

```
# include .bashrc if it exists
if [ -f "$HOME/.bashrc" ]; then
. "$HOME/.bashrc"
fi
```

Nous avons déjà vu comment l'exécution d'un script peut conduire à celle d'un autre. Ainsi, l'instruction `if` garantit que le fichier `$HOME/.bashrc` — s'il existe (`-f`) — sera sourcé (c'est-à-dire lu et exécuté) à l'ouverture de la session :

```
. "$HOME/.bashrc"
```

NOTE

Comme nous le verrons dans la prochaine leçon, `$HOME` est une variable d'environnement qui se développe pour indiquer le chemin absolu vers le répertoire personnel de l'utilisateur.

Par ailleurs, nous pouvons utiliser le `.` lorsque nous avons modifié un fichier de démarrage et que nous souhaitons que les changements soient effectifs sans redémarrage. Par exemple, nous pouvons :

- ajouter un alias à `~/ .bashrc`:

```
user2@debian:~$ echo "alias hi='echo We salute you.'" >> ~/.bashrc
```

WARNING

Lorsque vous envoyez la sortie d'une commande dans un fichier, ne confondez pas l'ajout (`>>`) et l'écrasement (`>`).

- afficher la dernière ligne de `~/ .bashrc` pour vérifier si tout s'est bien passé :

```
user2@debian:~$ tail -n 1 !$
tail -n 1 ~/.bashrc
alias hi='echo We salute you.'
```

NOTE

`!$` remplace le dernier argument de la commande précédente, c'est-à-dire `~/ .bashrc` :

- sourcer le fichier en ligne de commande :

```
user2@debian:~$ . ~/.bashrc
```

- et invoquer l'alias pour montrer que ça marche :

```
user2@debian:~$ hi
We salute you.
```

NOTE

Reportez-vous à la prochaine leçon pour en savoir plus sur les *alias* et les *variables*.

Sourcer les fichiers avec `source`

La commande `source` est un synonyme de `..`. Ainsi, pour sourcer `~/ .bashrc`, nous pouvons également le faire comme ceci :

```
user2@debian:~$ source ~/.bashrc
```

L'origine des fichiers de démarrage du shell : SKEL

SKEL est une variable dont la valeur est le chemin absolu vers le répertoire skel. Ce répertoire sert de modèle pour la structure du système de fichiers des répertoires personnels des utilisateurs. Il inclut les fichiers qui seront hérités par tout nouveau compte utilisateur créé (y compris, bien sûr, les fichiers de configuration des interpréteurs de commandes). SKEL et les autres variables associées sont stockées dans /etc/adduser.conf, qui est le fichier de configuration pour adduser :

```
user2@debian:~$ grep SKEL /etc/adduser.conf
# The SKEL variable specifies the directory containing "skeletal" user
SKEL=/etc/skel
# If SKEL_IGNORE_REGEX is set, adduser will ignore files matching this
SKEL_IGNORE_REGEX="dpkg-(old|new|dist|save)"
```

SKEL est défini à /etc/skel ; c'est là que sont rangés les scripts de démarrage qui configurent nos shells :

```
user2@debian:~$ ls -a /etc/skel/
.  ..  .bash_logout  .bashrc  .profile
```

WARNING

Rappelez-vous que les fichiers qui commencent par un `.` sont cachés, nous devons donc utiliser `ls -a` pour les voir lorsque nous affichons le contenu d'un répertoire.

Ajoutons un répertoire dans /etc/skel pour tous les nouveaux utilisateurs afin qu'ils y stockent leurs scripts personnels :

1. En tant que root, nous nous rendons dans /etc/skel :

```
root@debian:~# cd /etc/skel/
root@debian:/etc/skel#
```

2. Nous affichons son contenu :

```
root@debian:/etc/skel# ls -a
```

```
. .. .bash_logout .bashrc .profile
```

3. Nous créons notre répertoire et nous vérifions que tout s'est passé comme prévu :

```
root@debian:/etc/skel# mkdir my_personal_scripts
root@debian:/etc/skel# ls -a
. .. .bash_logout .bashrc my_personal_scripts .profile
```

4. Supprimons `user2` et son répertoire `home` :

```
root@debian:~# deluser --remove-home user2
Looking for files to backup/remove ...
Removing files ...
Removing user `user2' ...
Warning: group `user2' has no more members.
Done.
```

5. Nous ajoutons à nouveau `user2` de manière à ce qu'il dispose d'un nouveau répertoire utilisateur :

```
root@debian:~# adduser user2
Adding user `user2' ...
Adding new group `user2' (1001) ...
Adding new user `user2' (1001) with group `user2' ...
Creating home directory `/home/user2' ...
Copying files from `/etc/skel' ...
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for user2
Enter the new value, or press ENTER for the default
  Full Name []:
  Room Number []:
  Work Phone []:
  Home Phone []:
  Other []:
Is the information correct? [Y/n] y
```

6. Enfin, nous nous connectons en tant que `user2` et affichons tous les fichiers dans `/home/user2` pour voir si tout s'est passé comme prévu :

```
root@debian:~# su - user2
user2@debian:~$ pwd
/home/user2
user2@debian:~$ ls -a
.  ..  .bash_history  .bash_logout  .bashrc  my_personal_scripts  .profile
```

C'est le cas.

Exercices guidés

1. Regardez comment les shells ont été lancés dans la colonne "Shell démarré avec..." et complétez avec les informations requises :

Shell démarré avec...	Interactif ?	Connexion ?	Résultat de <code>echo \$0</code>
<code>sudo ssh</code> <code>user2@machine2</code>			
<code>Ctrl + Alt + F2</code>			
<code>su - user2</code>			
<code>gnome-terminal</code>			
Un utilisateur lambda utilise <i>konsole</i> pour lancer une instance de <i>sakura</i>			
Un script nommé <code>test.sh</code> qui contient la commande <code>echo \$0</code>			

2. Écrivez les commandes `su` et `sudo` pour lancer le shell spécifié :

Shell de connexion interactif en tant que `user2`

`su:`

`sudo:`

Shell de connexion interactif en tant que `root`

`su:`

`sudo:`

Shell interactif simple en tant que `root`

`su:`

`sudo:`

Shell interactif simple en tant que `user2`

`su:`

sudo:

3. Quel fichier de démarrage est lu lorsque le shell dans la colonne "Type de shell" est démarré ?

Type de shell	/etc/profile	/etc/bash.bashrc	~/.profile	~/.bashrc
Shell de connexion interactif en tant que user2				
Shell de connexion interactif en tant que root				
Shell interactif simple en tant que root				
Shell interactif simple en tant que user2				

Exercices d'approfondissement

1. Dans Bash, nous pouvons écrire une fonction simple `Hello world!` en incorporant le code suivant dans un fichier vide :

```
function hello() {  
    echo "Hello world!"  
}
```

- Que devons-nous faire ensuite pour rendre la fonction utilisable par le shell ?
- Une fois qu'elle est disponible dans le shell en cours, comment l'invoquer ?
- Pour automatiser les choses, dans quel fichier pourriez-vous ranger la fonction et son invocation pour qu'elle soit exécutée lorsque `user2` ouvre un terminal depuis une session graphique ? De quel type de shell s'agit-il ?
- Dans quel fichier pourriez-vous ranger la fonction et son invocation pour qu'elle soit exécutée lorsque `root` lance un nouveau shell interactif, qu'il s'agisse d'un shell de connexion ou d'un shell simple ?

2. Jetez un œil à ce simple script Bash `Hello world!` :

```
#!/bin/bash  
  
#hello_world: a simple bash script to discuss interaction in scripts.  
  
echo "Hello world!"
```

- Admettons que nous rendions le script exécutable et que nous l'exécutons. Est-ce qu'il s'agirait d'un script interactif ? Pourquoi ?
- Qu'est-ce qui rend un script interactif ?

3. Imaginez que vous ayez changé les valeurs de certaines variables dans `~/ .bashrc` et que vous souhaitiez que ces changements prennent effet sans redémarrage. Depuis votre répertoire personnel, comment pourriez-vous y parvenir de deux manières différentes ?

4. John vient de démarrer une session graphique sur un serveur Linux. Il ouvre un émulateur de terminal pour effectuer quelques tâches administratives mais, à sa grande surprise, la session se bloque et il doit ouvrir un terminal.

- Comment peut-il ouvrir ce shell `ttty` ?

- Quels sont les fichiers de démarrage qui seront sourcés ?

5. Linda utilise un serveur Linux. Elle demande gentiment à l'administrateur de disposer d'un fichier `~/ .bash_login` de manière à ce que l'heure et la date s'affichent à l'écran lorsqu'elle se connecte. D'autres utilisateurs apprécient cette idée et font de même. L'administrateur a du mal à créer le fichier pour tous les autres utilisateurs du serveur. Il décide donc d'ajouter une nouvelle politique et de créer `~/ .bash_login` pour tous les nouveaux utilisateurs potentiels. Comment l'administrateur peut-il accomplir cette tâche ?

Résumé

Voici ce que nous avons appris dans cette leçon :

- Les shells définissent l'environnement des utilisateurs dans un système Linux.
- Bash est le principal interpréteur de commandes de toutes les distributions GNU/Linux.
- La première tâche d'un shell consiste à lire et à exécuter un ou plusieurs fichiers de démarrage.
- Les concepts d'interaction et d'e connexion en rapport avec les shells.
- Comment lancer différents types de shells avec `bash`, `su`, `sudo` et `Ctrl + Alt + F1-F6`.
- Comment vérifier le type de shell avec `echo $0`.
- Les fichiers de démarrage locaux `~/.bash_profile`, `~/.profile`, `~/.bash_login`, `~/.bash_logout` et `~/.bashrc`.
- Les fichiers de démarrage globaux `/etc/profile`, `/etc/profile.d/*`, `/etc/bash.bashrc`.
- Les fichiers locaux prennent le pas sur les fichiers globaux.
- Comment rediriger la sortie d'une commande avec `>` (écrasement) et `>>` (ajout).
- La signification du répertoire `skel`.
- Comment sourcer des fichiers.

Commandes utilisées dans cette leçon :

bash

Créer un nouveau shell.

su

Créer un nouveau shell.

sudo

Créer un nouveau shell.

usermod

Modifier un compte utilisateur.

echo

Afficher une ligne de texte.

ps

Afficher un aperçu des processus en cours.

less

Un visualiseur pour les fichiers longs.

ssh

Démarrer une connexion Open SSH (à distance).

chmod

Modifier les permissions d'un fichier, pour le rendre exécutable par exemple.

grep

Afficher les lignes qui correspondent à un motif.

ls

Afficher le contenu d'un répertoire.

cd

Changer de répertoire.

mkdir

Créer un répertoire.

deluser

Supprimer un utilisateur.

adduser

Ajouter un nouvel utilisateur.

.

Sourcer un fichier.

source

Sourcer un fichier.

tail

Afficher la dernière partie d'un fichier.

Réponses aux exercices guidés

1. Regardez comment les shells ont été lancés dans la colonne "Shell démarré avec..." et complétez avec les informations requises :

Shell démarré avec...	Interactif ?	Connexion ?	Résultat de <code>echo \$0</code>
<code>sudo ssh</code> <code>user2@machine2</code>	Oui	Oui	<code>-bash</code>
<code>Ctrl</code> + <code>Alt</code> + <code>F2</code>	Oui	Oui	<code>-bash</code>
<code>su - user2</code>	Oui	Oui	<code>-bash</code>
<code>gnome-terminal</code>	Oui	Non	<code>bash</code>
Un utilisateur lambda utilise <i>konsole</i> pour lancer une instance de <i>sakura</i>	Oui	Non	<code>/bin/bash</code>
Un script nommé <code>test.sh</code> qui contient la commande <code>echo \$0</code>	Non	Non	<code>./test.sh</code>

2. Écrivez les commandes `su` et `sudo` pour lancer le shell spécifié :

Shell de connexion interactif en tant que `user2`

`su`

`su - user2`, `su -l user2` ou `su --login user2`

`sudo`

`sudo su - user2`, `sudo su -l user2` ou `sudo su --login user2`

Shell de connexion interactif en tant que `root`

`su`

`su - root` ou `su -`

`sudo`

`sudo su - root`, `sudo su -` ou `sudo -i`

Shell interactif simple en tant que root

su

su root ou su

sudo

sudo su root, sudo su, sudo -s ou sudo -u root -s

Shell interactif simple en tant que user2

su

su user2

sudo

sudo su user2 ou sudo -u user2 -s

3. Quel fichier de démarrage est lu lorsque le shell dans la colonne "Type de shell" est démarré ?

Type de shell	/etc/profile	/etc/bash.bashrc	~/.profile	~/.bashrc
Shell de connexion interactif en tant que user2	Oui	Oui	Oui	Oui
Shell de connexion interactif en tant que root	Oui	Oui	Oui	Oui
Shell interactif simple en tant que root	Non	Oui	Non	Oui
Shell interactif simple en tant que user2	Non	Oui	Non	Oui

Réponses aux exercices d'approfondissement

1. Dans Bash, nous pouvons écrire une fonction simple `Hello world!` en incorporant le code suivant dans un fichier vide :

```
function hello() {  
    echo "Hello world!"  
}
```

- Que devons-nous faire ensuite pour rendre la fonction utilisable par le shell ?

Pour rendre la fonction utilisable par le shell en cours, nous devons sourcer le fichier dans lequel elle figure.

- Une fois qu'elle est disponible dans le shell en cours, comment l'invoquer ?

Nous pouvons l'invoquer en tapant son nom dans le terminal.

- Pour automatiser les choses, dans quel fichier pourriez-vous ranger la fonction et son invocation pour qu'elle soit exécutée lorsque `user2` ouvre un terminal depuis une session graphique ? De quel type de shell s'agit-il ?

Le meilleur endroit pour la ranger serait `/home/user2/.bashrc`. Le shell invoqué serait un shell interactif simple.

- Dans quel fichier pourriez-vous ranger la fonction et son invocation pour qu'elle soit exécutée lorsque `root` lance un nouveau shell interactif, qu'il s'agisse d'un shell de connexion ou d'un shell simple ?

Dans `/etc/bash.bashrc` puisque ce fichier est exécuté pour tous les shells interactifs, qu'il s'agisse d'un shell de connexion ou d'un shell simple.

2. Jetez un œil à ce simple script Bash `Hello world!` :

```
#!/bin/bash  
  
#hello_world: a simple bash script to discuss interaction in scripts.  
  
echo "Hello world!"
```

- Admettons que nous rendions le script exécutable et que nous l'exécutons. Est-ce qu'il

s'agirait d'un script interactif ? Pourquoi ?

Non, étant donné qu'il n'y a pas d'interaction avec un humain et qu'aucune commande n'est saisie par l'utilisateur.

- Qu'est-ce qui rend un script interactif ?

Le fait qu'il nécessite l'intervention de l'utilisateur.

3. Imaginez que vous ayez changé les valeurs de certaines variables dans `~/ .bashrc` et que vous souhaitiez que ces changements prennent effet sans redémarrage. Depuis votre répertoire personnel, comment pourriez-vous y parvenir de deux manières différentes ?

```
$ source .bashrc
```

ou bien

```
$ . .bashrc
```

4. John vient de démarrer une session graphique sur un serveur Linux. Il ouvre un émulateur de terminal pour effectuer quelques tâches administratives mais, à sa grande surprise, la session se bloque et il doit ouvrir un terminal.

- Comment peut-il ouvrir ce shell `tty` ?

Il peut le faire en appuyant sur `Ctrl + Alt + F1-F6` pour entrer dans l'un des six shells `tty`.

- Quels sont les fichiers de démarrage qui seront sourcés ?

```
/etc/profile
```

```
/home/john/.profile
```

5. Linda utilise un serveur Linux. Elle demande gentiment à l'administrateur de disposer d'un fichier `~/ .bash_login` de manière à ce que l'heure et la date s'affichent à l'écran lorsqu'elle se connecte. D'autres utilisateurs apprécient cette idée et font de même. L'administrateur a du mal à créer le fichier pour tous les autres utilisateurs du serveur. Il décide donc d'ajouter une nouvelle politique et de créer `~/ .bash_login` pour tous les nouveaux utilisateurs potentiels. Comment l'administrateur peut-il accomplir cette tâche ?

Il peut y parvenir en intégrant `.bash_login` dans le répertoire `/etc/skel`.



105.1 Leçon 2

Certification :	LPIC-1
Version :	5.0
Thème :	105 Shells et scripts shell
Objectif :	105.1 Personnaliser et utiliser l'environnement shell
Leçon :	2 sur 3

Introduction

Une variable est une sorte de boîte imaginaire dans laquelle on range provisoirement un bout d'information. Comme pour les scripts d'initialisation, Bash distingue deux types de variables : *shell/local* (celles qui n'existent que dans les limites du shell dans lequel elles ont été créées) et *environment/global* (celles qui sont héritées par les shells et/ou les processus enfants). Dans la leçon précédente, nous avons examiné les shells et leurs scripts de configuration ou d'initialisation. La puissance de ces fichiers de démarrage réside dans le fait qu'ils nous permettent d'utiliser des variables — ainsi que des alias et des fonctions — qui nous aident à créer et à personnaliser l'environnement shell de notre choix.

Variables : Affectation et référence

Une variable peut être définie comme un nom qui contient une valeur.

Dans Bash, l'attribution d'une valeur à un nom est appelée *affectation de variable* et c'est la manière par laquelle nous créons ou définissons des variables. D'autre part, le processus qui permet d'accéder à la valeur contenue dans le nom est appelé *référencement de variable*.

Voici la syntaxe pour affecter des variables :

```
<variable_name>=<variable_value>
```

Par exemple :

```
$ distro=zorinos
```

La variable `distro` est égale à `zorinos`, c'est-à-dire qu'il existe une portion de mémoire qui contient la valeur `zorinos` — et `distro` constitue le pointeur vers cette valeur.

Notez toutefois qu'il ne peut y avoir d'espace de part et d'autre du signe égal lors de l'affectation d'une variable :

```
$ distro =zorinos
-bash: distro: command not found
$ distro= zorinos
-bash: zorinos: command not found
```

À cause de notre erreur, Bash a interprété `distro` et `zorinos` comme des commandes.

Pour référencer une variable (c'est-à-dire pour connaître sa valeur), on utilise la commande `echo` en faisant précéder le nom de la variable du signe `$` :

```
$ echo $distro
zorinos
```

Noms de variables

Lorsque nous choisissons les noms des variables, nous devons prendre en compte un certain nombre de règles.

Le nom d'une variable peut contenir des lettres (`a-z`, `A-Z`), des chiffres (`0-9`) et des tirets bas (`_`) :

```
$ distro=zorinos
$ echo $distro
zorinos
$ DISTRO=zorinos
$ echo $DISTRO
```

```
zorinos
$ distro_1=zorinos
$ echo $distro_1
zorinos
$ _distro=zorinos
$ echo $_distro
zorinos
```

Il ne doit pas commencer par un chiffre, autrement Bash va s'embrouiller :

```
$ 1distro=zorinos
-bash: 1distro=zorinos: command not found
```

Il ne peut pas contenir d'espaces (même entre guillemets) ; par convention, on utilise plutôt les tirets bas :

```
$ "my distro "=zorinos
-bash: my: command not found
$ my_distro=zorinos
$ echo $my_distro
zorinos
```

Valeurs des variables

En ce qui concerne la référence ou la valeur des variables, il faut également tenir compte d'un certain nombre de règles.

Les variables peuvent contenir tous les caractères alphanumériques (a-z, A-Z, 0-9) ainsi que la plupart des autres caractères (?, !, *, ., /, etc.) :

```
$ distro=zorin12.4?
$ echo $distro
zorin12.4?
```

Les valeurs des variables doivent être placées entre guillemets si elles contiennent des espaces simples :

```
$ distro=zorin 12.4
-bash: 12.4: command not found
$ distro="zorin 12.4"
```

```
$ echo $distro
zorin 12.4
$ distro='zorin 12.4'
$ echo $distro
zorin 12.4
```

Les valeurs des variables doivent également être placées entre guillemets si elles contiennent des caractères tels que ceux utilisés pour la redirection (<,>) ou la barre verticale (|). La seule chose que fait la commande suivante est de créer un fichier vide nommé `zorin` :

```
$ distro=>zorin
$ echo $distro

$ ls zorin
zorin
```

En revanche, ça fonctionne avec les guillemets :

```
$ distro=">zorin"
$ echo $distro
>zorin
```

En revanche, les guillemets simples et les guillemets doubles ne sont pas toujours interchangeables. Selon ce que l'on fait avec une variable (assignation ou référencement), l'utilisation de l'un ou de l'autre a des implications et donnera des résultats différents. Dans le contexte de l'affectation d'une variable, les guillemets simples prennent tous les caractères de la valeur de la variable *littéralement*, alors que les guillemets doubles permettent une substitution de la variable :

```
$ lizard=uromastyx
$ animal='My $lizard'
$ echo $animal
My $lizard
$ animal="My $lizard"
$ echo $animal
My uromastyx
```

Par ailleurs, lorsqu'on référence une variable dont la valeur inclut des espaces initiaux (ou supplémentaires) — parfois combinés avec des astérisques — il faut impérativement utiliser des guillemets doubles après la commande `echo` pour éviter le *fractionnement des champs* et

l'expansion des chemins :

```
$ lizard="  genus  |  uromastyx"
$ echo $lizard
genus | uromastyx
$ echo "$lizard"
genus  |  uromastyx
```

Si la référence de la variable contient un point d'exclamation final, celui-ci doit constituer le dernier caractère de la chaîne (sans quoi Bash estimera qu'il s'agit d'un événement `history`) :

```
$ distro=zorin.?!os
-bash: !os: event not found
$ distro=zorin.?!
$ echo $distro
zorin.?!
```

Les antislashes doivent être échappés par un autre antislash. De plus, si une barre oblique inverse est le dernier caractère de la chaîne et que nous ne l'échappons pas, Bash va considérer que nous souhaitons un retour à la ligne et va nous afficher une nouvelle ligne :

```
$ distro=zorinos\
>
$ distro=zorinos\\
$ echo $distro
zorinos\
```

Dans les deux prochaines sections, nous allons présenter les principales différences entre les variables *locales* et les variables *d'environnement*.

Variables locales ou variables du shell

Les variables locales ou variables du shell n'existent que dans le shell dans lequel elles ont été créées. Par convention, les variables locales sont écrites en caractères minuscules.

Pour effectuer quelques tests, nous allons créer une variable locale. Comme nous l'avons vu plus haut, nous choisissons un nom de variable approprié et nous l'associons à une valeur appropriée. Par exemple :

```
$ reptile=tortoise
```

Utilisons maintenant la commande `echo` pour référencer notre variable et vérifier que tout s'est passé comme prévu :

```
$ echo $reptile
tortoise
```

Dans certains contextes, par exemple lors de l'écriture de scripts, l'immutabilité peut constituer une caractéristique intéressante pour les variables. Si nous souhaitons que nos variables soient immuables, nous pouvons soit les créer `readonly` :

```
$ readonly reptile=tortoise
```

Ou les rendre telles une fois qu'elles ont été créées :

```
$ reptile=tortoise
$ readonly reptile
```

Dorénavant, si nous essayons de changer la valeur de `reptile`, Bash va refuser :

```
$ reptile=lizard
-bash: reptile: readonly variable
```

NOTE

Pour afficher la liste de toutes les variables en lecture seule dans notre session actuelle, tapez `readonly` ou `readonly -p` dans le terminal.

Une commande pratique pour manipuler les variables locales est `set`.

`set` affiche toutes les variables et toutes les fonctions du shell actuellement assignées. Comme cela peut représenter beaucoup de lignes (essayez vous-même !), il est recommandé de l'utiliser en combinaison avec un visualiseur comme `less` :

```
$ set | less
BASH=/bin/bash
BASHOPTS=checkwinsize:cmdhist:complete_fullquote:expand_aliases:extglob:extquote:force_igno
re:histappend:interactive_comments:login_shell:progcomp:promptvars:sourcpath
BASH_ALIASES=( )
```



```

BASH_ARGC=()
BASH_ARGV=()
BASH_CMDS=()
BASH_COMPLETION_COMPAT_DIR=/etc/bash_completion.d
BASH_LINENO=()
BASH_SOURCE=()
BASH_VERSINFO=( [0]="4" [1]="4" [2]="12" [3]="1" [4]="release" [5]="x86_64-pc-linux-gnu" )
BASH_VERSION='4.4.12(1)-release'
(...)

```

Est-ce que notre variable `reptile` est là ?

```

$ set | grep reptile
reptile=tortoise

```

Oui, la voilà !

En revanche, la variable locale `reptile` ne sera pas héritée par les processus enfants créés à partir du shell actuel :

```

$ bash
$ set | grep reptile
$

```

Et, bien évidemment, nous ne pouvons pas non plus afficher sa valeur avec `echo` :

```

$ echo $reptile
$

```

NOTE

En tapant la commande `bash` dans le terminal, nous ouvrons un nouveau shell (enfant).

Pour réinitialiser des variables (locales ou globales), nous utilisons la commande `unset` :

```

$ echo $reptile
tortoise
$ unset reptile
$ echo $reptile
$

```

NOTE `unset` doit être suivi du seul nom de la variable (sans le symbole \$).

Variables globales ou variables d'environnement

Les variables globales ou variables d'environnement existent pour le shell actuel ainsi que pour tous les processus ultérieurs créés à partir de celui-ci. Par convention, les variables d'environnement s'écrivent en majuscules :

```
$ echo $SHELL
/bin/bash
```

Nous pouvons transmettre récursivement la valeur de ces variables à d'autres variables :

```
$ my_shell=$SHELL
$ echo $my_shell
/bin/bash
$ your_shell=$my_shell
$ echo $your_shell
/bin/bash
$ our_shell=$your_shell
$ echo $our_shell
/bin/bash
```

Pour qu'une variable locale du shell devienne une variable d'environnement, il faut utiliser la commande `export` :

```
$ export reptile
```

Avec `export reptile` nous avons transformé notre variable locale en une variable d'environnement de sorte que les shells enfants puissent la reconnaître et l'utiliser :

```
$ bash
$ echo $reptile
tortoise
```

De même, `export` peut être utilisé pour définir et exporter une variable d'une traite :

```
$ export amphibian=frog
```

Nous pouvons désormais lancer une nouvelle instance de Bash et référencer la nouvelle variable avec succès :

```
$ bash
$ echo $amphibian
frog
```

NOTE

Avec `export -n <NOM-DE-VARIABLE>` la variable sera retransformée en une variable locale du shell.

La commande `export` nous affichera également une liste de toutes les variables d'environnement existantes lorsqu'elle est invoquée sans argument (ou avec l'option `-p`) :

```
$ export
declare -x HOME="/home/user2"
declare -x LANG="en_GB.UTF-8"
declare -x LOGNAME="user2"
(...)
declare -x PATH="/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games"
declare -x PWD="/home/user2"
declare -x SHELL="/bin/bash"
declare -x SHLVL="1"
declare -x SSH_CLIENT="192.168.1.10 49330 22"
declare -x SSH_CONNECTION="192.168.1.10 49330 192.168.1.7 22"
declare -x SSH_TTY="/dev/pts/0"
declare -x TERM="xterm-256color"
declare -x USER="user2"
declare -x XDG_RUNTIME_DIR="/run/user/1001"
declare -x XDG_SESSION_ID="8"
declare -x reptile="tortoise"
```

NOTE

La commande `declare -x` est équivalente à `export`.

Deux autres commandes peuvent être utilisées pour afficher la liste de toutes les variables d'environnement : `env` et `printenv` :

```
$ env
SSH_CONNECTION=192.168.1.10 48678 192.168.1.7 22
LANG=en_GB.UTF-8
XDG_SESSION_ID=3
USER=user2
```

```
PWD=/home/user2
HOME=/home/user2
SSH_CLIENT=192.168.1.10 48678 22
SSH_TTY=/dev/pts/0
MAIL=/var/mail/user2
TERM=xterm-256color
SHELL=/bin/bash
SHLVL=1
LOGNAME=user2
XDG_RUNTIME_DIR=/run/user/1001
PATH=/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games
_=/usr/bin/env
```

En plus d'être un synonyme de `env`, la commande `printenv` peut être utilisée de la même manière que la commande `echo` pour vérifier la valeur d'une variable :

```
$ echo $PWD
/home/user2
$ printenv PWD
/home/user2
```

Notez cependant qu'avec `printenv`, le nom de la variable n'est pas précédé de `$`.

NOTE

`PWD` contient le chemin du répertoire de travail actuel. Nous aurons l'occasion d'en apprendre davantage sur cette variable et d'autres variables d'environnement courantes un peu plus tard.

Exécuter un programme dans un environnement modifié

`env` peut être utilisé pour modifier l'environnement du shell lors de l'exécution d'un programme.

Pour démarrer une nouvelle session Bash avec un environnement aussi vide que possible — en effaçant la plupart des variables (ainsi que les fonctions et les alias) — nous utiliserons `env` avec l'option `-i` :

```
$ env -i bash
```

À présent, la plupart de nos variables d'environnement ont disparu :

```
$ echo $USER
```

```
$
```

Et il n'en reste que quelques-unes :

```
$ env
LS_COLORS=
PWD=/home/user2
SHLVL=1
_=/usr/bin/printenv
```

Nous pouvons également utiliser `env` pour définir une variable donnée pour un programme en particulier.

Dans la leçon précédente, lors de la discussion sur les shells simples non-interactifs, nous avons vu comment les scripts ne lisent pas les fichiers de démarrage standard, mais recherchent plutôt la valeur de la variable `BASH_ENV` pour l'utiliser en tant que fichier de démarrage si elle existe.

Voyons comment cela se passe :

1. Nous créons notre propre fichier de démarrage appelé `.startup_script` avec le contenu suivant :

```
CROCODILIAN=caiman
```

2. Nous écrivons un script Bash nommé `test_env.sh` avec le contenu suivant :

```
#!/bin/bash

echo $CROCODILIAN
```

3. Nous rendons notre script `test_env.sh` exécutable :

```
$ chmod +x test_env.sh
```

4. Enfin, nous utilisons `env` pour définir `BASH_ENV` à `.startup_script` pour `test_env.sh` :

```
$ env BASH_ENV=/home/user2/.startup_script ./test_env.sh
caiman
```

La commande `env` est implicite même si nous ne l'utilisons pas :

```
$ BASH_ENV=/home/user2/.startup_script ./test_env.sh
caiman
```

NOTE

Si vous ne comprenez pas la ligne `#!/bin/bash` ou la commande `chmod +x`, pas de panique ! Nous apprendrons tout ce qu'il faut savoir sur les scripts shell dans les prochaines leçons. Pour l'instant, rappelez-vous simplement que pour exécuter un script depuis son propre répertoire, nous utilisons `./some_script`.

Variables d'environnement courantes

Le moment est venu de passer en revue les variables d'environnement les plus pertinentes qui sont définies dans les fichiers de configuration de Bash.

DISPLAY

Liée au serveur X, la valeur de cette variable est normalement constituée de trois éléments :

- Le nom d'hôte (l'absence de celui-ci signifie `localhost`) où le serveur X est exécuté.
- Un deux-points comme délimiteur.
- Un chiffre (normalement `0` et se réfère à l'affichage de l'ordinateur).

```
$ printenv DISPLAY :0
```

Une valeur vide pour cette variable signifie que le serveur n'a pas de système graphique. Un chiffre supplémentaire — comme dans `my.xserver:0:1` — ferait référence au numéro de l'écran s'il en existe plus d'un.

HISTCONTROL

Cette variable contrôle les commandes qui sont sauvegardées dans le `HISTFILE` (voir ci-dessous). Il y a trois valeurs possibles :

ignorespace

Les commandes qui commencent par une espace ne sont pas enregistrées.

ignoredups

Une commande identique à la commande précédente ne sera pas enregistrée.

ignoreboth

Les commandes qui entrent dans l'une des deux catégories précédentes ne seront pas enregistrées.

```
$ echo $HISTCONTROL
ignoreboth
```

HISTSIZE

Définit le nombre de commandes à stocker en mémoire pendant la durée de la session du shell.

```
$ echo $HISTSIZE
1000
```

HISTFILESIZE

Définit le nombre de commandes à enregistrer dans HISTFILE au début et à la fin de la session :

```
$ echo $HISTFILESIZE
2000
```

HISTFILE

Le nom du fichier qui enregistre toutes les commandes au fur et à mesure qu'elles sont tapées. Par défaut, ce fichier se situe dans `~/.bash_history` :

```
$ echo $HISTFILE
/home/user2/.bash_history
```

NOTE

Pour voir le contenu de HISTFILE, il suffit de taper `history`. Alternativement, nous pouvons spécifier le nombre de commandes que nous voulons voir en ajoutant un argument (le nombre de commandes les plus récentes) à `history` comme dans `history 3`.

HOME

Cette variable contient le chemin absolu du répertoire personnel de l'utilisateur actuel et elle est définie lorsque l'utilisateur se connecte.

Ce bout de code — issu de `~/.profile` — est assez intuitif (il source "`$HOME/.bashrc`" s'il existe) :

```
# include .bashrc if it exists
if [ -f "$HOME/.bashrc" ]; then
. "$HOME/.bashrc"
fi
```

NOTE

Si vous ne comprenez pas bien l'instruction `if`, ne vous inquiétez pas : reportez-vous simplement aux leçons sur les scripts shell.

Rappelez-vous que `~` équivaut à `$HOME` :

```
$ echo ~; echo $HOME
/home/carol
/home/carol
```

NOTE

Les commandes peuvent être combinées avec un point-virgule (;).

Nous pouvons le prouver à l'aide d'une condition `if` :

```
$ if [ ~ == "$HOME" ]; then echo "true"; else echo "false"; fi
true
```

NOTE

Rappelez-vous : Le signe égal `=` est utilisé pour l'affectation de variables. `==` est utilisé pour tester l'égalité.

HOSTNAME

Cette variable stocke le nom TCP/IP de l'ordinateur hôte :

```
$ echo $HOSTNAME
debian
```

HOSTTYPE

Celle-ci stocke l'architecture du processeur de l'ordinateur hôte :

```
$ echo $HOSTTYPE
x86_64
```

LANG

Cette variable enregistre la localisation du système :


```
$ echo $LANG
en_UK.UTF-8
```

LD_LIBRARY_PATH

Cette variable consiste en une liste de répertoires, séparés par deux points, dans lesquels les programmes ont accès aux bibliothèques partagées :

```
$ echo $LD_LIBRARY_PATH
/usr/local/lib
```

MAIL

Cette variable indique le fichier dans lequel Bash va chercher le courrier électronique :

```
$ echo $MAIL
/var/mail/carol
```

Une autre valeur courante pour cette variable est `/var/spool/mail/$USER`.

MAILCHECK

Cette variable enregistre une valeur numérique qui indique en secondes la fréquence à laquelle Bash vérifie l'arrivée de nouveaux e-mails :

```
$ echo $MAILCHECK
60
```

PATH

Cette variable d'environnement contient la liste des répertoires dans lesquels Bash va chercher les fichiers exécutables lorsqu'on lui demande d'exécuter un programme. Dans notre machine de test, cette variable est définie dans le fichier système `/etc/profile` :

```
if [ "`id -u`" -eq 0 ]; then
    PATH="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"
else
    PATH="/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games"
fi
export PATH
```

L'instruction `if` vérifie l'identité de l'utilisateur et—en fonction du résultat (`root` ou autre)—nous obtenons l'un ou l'autre des `PATH`. Au final, le `PATH` sélectionné est propagé avec `export`.

Notez deux détails concernant la valeur de `PATH` :

- Les noms des répertoires sont écrits en utilisant des chemins absolus.
- Le deux-points est utilisé comme délimiteur.

Pour ajouter le dossier `/usr/local/sbin` dans le `PATH` des utilisateurs réguliers, il faudrait modifier la ligne de manière à ce qu'elle ressemble à ceci :

```
(...)
else
    PATH="/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games:/usr/local/sbin"
fi
export PATH
```

Nous pouvons voir à présent comment la valeur de la variable change lorsque nous nous connectons en tant qu'utilisateur normal :

```
# su - carol
$ echo $PATH
/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games:/usr/local/sbin
```

NOTE

Nous aurions également pu ajouter `/usr/local/sbin` en ligne de commande en tapant soit `PATH=/usr/local/sbin:$PATH` soit `PATH=$PATH:/usr/local/sbin`—le premier faisant de `/usr/local/sbin` le premier répertoire à rechercher pour les fichiers exécutables ; le second le dernier.

PS1

Cette variable enregistre la valeur de l'invite Bash. Dans le bout de code qui suit (également issu de `/etc/profile`), l'instruction `if` teste l'identité de l'utilisateur et lui attribue une invite très discrète (`#` pour `root` ou `$` pour les utilisateurs normaux) :

```
if [ "`id -u`" -eq 0 ]; then
    PS1='# '
else
    PS1='$ '
```

```
fi
```

NOTE

L'identifiant `id` de `root` est `0`. Devenez `root` et voyez par vous-même avec `id -u`.

Voici les autres variables d'invite :

PS2

Normalement réglé sur `>` et utilisé comme invite de continuation pour les commandes longues et multilignes.

PS3

Utilisé comme invite pour la commande `select`.

PS4

Normalement fixé à `+` et utilisé pour le débogage.

SHELL

Cette variable contient le chemin absolu du shell utilisé :

```
$ echo $SHELL
/bin/bash
```

USER

Enregistre le nom de l'utilisateur actuel :

```
$ echo $USER
carol
```

Exercices guidés

1. Regardez l'affectation de la variable dans la colonne “Commande(s)” et indiquez si la variable résultante est “Locale” ou “Globale” :

Commande(s)	Locale	Globale
<code>debian=mother</code>		
<code>ubuntu=deb-based</code>		
<code>mint=ubuntu-based;</code> <code>export mint</code>		
<code>export suse=rpm-based</code>		
<code>zorin=ubuntu-based</code>		

2. Examinez la “Commande” et le “Résultat” et expliquez leur “Signification” :

Commande	Résultat	Signification
<code>echo \$HISTCONTROL</code>	<code>ignoreboth</code>	
<code>echo ~</code>	<code>/home/carol</code>	
<code>echo \$DISPLAY</code>	<code>reptilium:0:2</code>	
<code>echo \$MAILCHECK</code>	<code>60</code>	
<code>echo \$HISTFILE</code>	<code>/home/carol/.bash_history</code>	

3. Les variables sont définies de manière incorrecte dans la colonne “Commande erronée”. Fournissez les informations manquantes dans les colonnes “Commande correcte” et “Référence de la variable” afin d’obtenir le “Résultat attendu” :

Commande erronée	Commande correcte	Référence de la variable	Résultat attendu
<code>lizard =chameleon</code>			<code>chameleon</code>
<code>cool</code> <code>lizard=chameleon</code>			<code>chameleon</code>
<code>lizard=cha{pipe}me{pipe}leon</code>			<code>cha{pipe}me{pipe}leon</code>

Commande erronée	Commande correcte	Référence de la variable	Résultat attendu
<pre>lizard=/* :pipe: :backtick: `* :pipe:   :backtick: ` chameleon * :pipe: :backtick: `* :pipe:   :backtick: `/</pre>			<pre>/* :pipe: :backtick: `* :pipe:   :backtick: ` chamelon * :pipe: :backtick: `* :pipe:   :backtick: `/</pre>
<pre>win_path=C:\path\to\dir\</pre>			<pre>C:\path\to\dir\</pre>

4. Réfléchissez à l' "Objectif" et écrivez la "Commande" appropriée :

Objectif	Commande
Définir la langue du shell actuel en espagnol UTF-8 (es_ES.UTF-8).	
Afficher le nom du répertoire courant.	
Référencer la variable d'environnement qui enregistre les informations sur les connexions ssh.	
Définir PATH pour inclure /home/carol/scripts en tant que dernier répertoire dans lequel il faut chercher les exécutable.	
Définir la valeur de my_path à PATH.	
Définir la valeur de my_path à celle de PATH.	

5. Créez une variable locale nommée `mammal` et affectez-lui la valeur `gnu` :

6. En utilisant la substitution de variables, créez une autre variable locale nommée `var_sub` avec la valeur appropriée de sorte que lorsqu'elle est référencée par `echo $var_sub` nous obtenons `The value of mammal is gnu` :

7. Transformez `mammal` en variable d'environnement :

8. Retrouvez-la avec `set` et `grep`:

9. Retrouvez-la avec `env` et `grep`:

10. En deux commandes consécutives, créez une variable d'environnement nommée `BIRD` dont la valeur est `penguin` :

11. En une seule commande, créez une variable d'environnement nommée `NEW_BIRD` dont la valeur est `yellow-eyed penguin` :

12. En supposant que vous êtes `user2`, utilisez `mkdir` pour créer un dossier nommé `bin` dans votre répertoire personnel :

13. Tapez la commande pour ajouter le dossier `~/bin` à votre `PATH` afin qu'il soit le premier répertoire dans lequel `bash` recherche des binaires :

14. Pour assurer que la valeur de `PATH` reste inchangée après un redémarrage, quel bout de code — sous forme d'une instruction `if` — devriez-vous ajouter dans `~/.profile` ?

Exercices d'approfondissement

1. `let`: plus que l'évaluation d'une expression arithmétique :

- Faites une recherche dans le manuel en ligne ou sur le web à propos de `let` et son fonctionnement pour la définition des variables et créez une nouvelle variable locale nommée `my_val` dont la valeur est 10 — en additionnant 5 et 5 :

- Ensuite, créez une autre variable nommée `your_val` et dont la valeur est 5 — en divisant par 2 la valeur de `my_val` :

2. Le résultat d'une commande dans une variable ? Bien sûr que c'est possible ; c'est ce qu'on appelle la *substitution de commande*. Essayez en étudiant la fonction suivante, nommée `music_info` :

```
music_info(){
latest_music=`ls -l1t ~/Music | head -n 6`
echo -e "Your latest 5 music files:\n$latest_music"
}
```

Le résultat de la commande `ls -l1t ~/Music | head -n 6` devient la valeur de la variable `latest_music`. Ensuite, la variable `latest_music` est référencée dans la commande `echo` (qui affiche le nombre total d'octets occupés par le dossier `Music` et les cinq derniers fichiers musicaux stockés dans le dossier `Music` — un par ligne).

Laquelle des commandes suivantes est un remplacement valide pour

```
latest_music=`ls -l1t ~/Music | head -n 6`
```

Option A :

```
latest_music=$(ls -l1t ~/Music | head -n 6)
```

Option B :

```
latest_music="(ls -l1t ~/Music | head -n 6)"
```

Option C :

```
latest_music=((ls -l1t ~/Music| head -n 6))
```


Résumé

Voici ce que nous avons vu dans cette leçon :

- Les variables sont une partie très importante de l'environnement du shell étant donné qu'elles sont utilisées par l'interpréteur de commandes lui-même ainsi que par d'autres programmes.
- Comment affecter et référencer des variables.
- Les différences entre les variables *locales* et les variables *globales* (ou variables *d'environnement*).
- Comment rendre les variables en lecture seule (*readonly*).
- Comment transformer une variable locale en variable d'environnement avec la commande `export`.
- Comment afficher la liste de toutes les variables d'environnement.
- Comment exécuter un programme dans un environnement modifié.
- Comment rendre les variables persistantes à l'aide des scripts de démarrage.
- Quelques variables d'environnement courantes : `DISPLAY`, `HISTCONTROL`, `HISTSIZE`, `HISTFILESIZE`, `HISTFILE`, `HOME`, `HOSTNAME`, `HOSTTYPE`, `LANG`, `LD_LIBRARY_PATH`, `MAIL`, `MAILCHECK`, `PATH`, `PS1` (et d'autres variables d'invite), `SHELL` et `USER`.
- La signification du tilde (`~`).
- Les fondamentaux des instructions `if`.

Commandes utilisées dans cette leçon :

`echo`

Référence une variable.

`ls`

Affiche le contenu d'un répertoire.

`readonly`

Rend les variables immuables. Affiche la liste de toutes les variables en lecture seule dans la session en cours.

`set`

Affiche toutes les variables et les fonctions de la session en cours.

grep

Affiche les lignes qui correspondent à un motif.

bash

Lance un nouveau shell.

unset

Réinitialise une variable.

export

Transforme une variable locale en variable d'environnement. Affiche les variables d'environnement.

env

Affiche les variables d'environnement. Lance un programme dans un environnement modifié.

printenv

Affiche les variables d'environnement. Référence une variable.

chmod

Modifie les permissions d'un fichier, par exemple pour le rendre exécutable.

history

Affiche les commandes précédentes.

su

Modifier l'ID de l'utilisateur ou devenir superutilisateur.

id

Affiche l'ID de l'utilisateur.

Réponses aux exercices guidés

1. Regardez l'affectation de la variable dans la colonne “Commande(s)” et indiquez si la variable résultante est “Locale” ou “Globale” :

Commande(s)	Locale	Globale
debian=mother	Oui	Non
ubuntu=deb-based	Oui	Non
mint=ubuntu-based; export mint	Non	Oui
export suse=rpm-based	Non	Oui
zorin=ubuntu-based	Oui	Non

2. Examinez la “Commande” et le “Résultat” et expliquez leur “Signification” :

Commande	Résultat	Signification
echo \$HISTCONTROL	ignoreboth	Les commandes redondantes et celles qui commencent par un espace ne seront pas enregistrées dans history.
echo ~	/home/carol	Le HOME de carol est /home/carol.
echo \$DISPLAY	reptilium:0:2	La machine reptilium a un serveur X en marche et nous utilisons le deuxième écran de l'affichage.
echo \$MAILCHECK	60	Le courrier est vérifié toutes les minutes.
echo \$HISTFILE	/home/carol/.bash_history	history sera enregistré dans /home/carol/.bash_history.

3. Les variables sont définies de manière incorrecte dans la colonne “Commande erronée”. Fournissez les informations manquantes dans les colonnes “Commande correcte” et “Référence de la variable” afin d’obtenir le “Résultat attendu” :

Commande erronée	Commande correcte	Référence de la variable	Résultat attendu
<code>lizard =chameleon</code>	<code>lizard=chameleon</code>	<code>echo \$lizard</code>	<code>chameleon</code>
<code>cool lizard=chameleon</code>	<code>cool_lizard=chameleon (par exemple)</code>	<code>echo \$cool_lizard</code>	<code>chameleon</code>
<code>lizard=cha{pipe}me{pipe}leon</code>	<code>lizard="cha{pipe}me{pipe}leon" ou lizard='cha{pipe}me{pipe}leon'</code>	<code>echo \$lizard</code>	<code>cha{pipe}me{pipe}leon</code>
<code>lizard=/* :pipe: :backtick: `* :pipe:   :backtick: ` chameleon * :pipe: :backtick: `* :pipe:   :backtick: `/ ou lizard='/* :pipe: :backtick: `* :pipe:   :backtick: ` chameleon * :pipe: :backtick: `* :pipe:   :backtick: `/'</code>	<code>lizard="/* :pipe: :backtick: `* :pipe:   :backtick: ` chameleon * :pipe: :backtick: `* :pipe:   :backtick: `/"</code> ou <code>lizard='/* :pipe: :backtick: `* :pipe:   :backtick: ` chameleon * :pipe: :backtick: `* :pipe:   :backtick: `/'</code>	<code>echo "\$lizard"</code>	<code>/* :pipe: :backtick: `* :pipe:   :backtick: ` chamelon * :pipe: :backtick: `* :pipe:   :backtick: `/'</code>
<code>win_path=C:\path\to\dir\</code>	<code>win_path=C:\\path\\to\\dir\\</code>	<code>echo \$win_path</code>	<code>C:\path\to\dir\</code>

4. Réfléchissez à l' "Objectif" et écrivez la "Commande" appropriée :

Objectif	Commande
Définir la langue du shell actuel en espagnol UTF-8 (<code>es_ES.UTF-8</code>).	<code>LANG=es_ES.UTF-8</code>
Afficher le nom du répertoire courant.	<code>echo \$PWD</code> ou <code>pwd</code>

Objectif	Commande
Référencer la variable d'environnement qui enregistre les informations sur les connexions ssh.	<code>echo \$SSH_CONNECTION</code>
Définir PATH pour inclure /home/carol/scripts en tant que dernier répertoire dans lequel il faut chercher les exécutable.	<code>PATH=\$PATH:/home/carol/scripts</code>
Définir la valeur de my_path à PATH.	<code>my_path=PATH</code>
Définir la valeur de my_path à celle de PATH.	<code>my_path=\$PATH</code>

5. Créez une variable locale nommée `mammal` et affectez-lui la valeur `gnu` :

```
mammal=gnu
```

6. En utilisant la substitution de variables, créez une autre variable locale nommée `var_sub` avec la valeur appropriée de sorte que lorsqu'elle est référencée par `echo $var_sub` nous obtenons `The value of mammal is gnu` :

```
var_sub="The value of mammal is $mammal"
```

7. Transformez `mammal` en variable d'environnement :

```
export mammal
```

8. Retrouvez-la avec `set` et `grep`:

```
set | grep mammal
```

9. Retrouvez-la avec `env` et `grep`:

```
env | grep mammal
```

10. En deux commandes consécutives, créez une variable d'environnement nommée `BIRD` dont la valeur est `penguin` :

```
BIRD=penguin; export BIRD
```

11. En une seule commande, créez une variable d'environnement nommée `NEW_BIRD` dont la valeur est `yellow-eyed penguin` :

```
export NEW_BIRD="yellow-eyed penguin"
```

ou bien

```
export NEW_BIRD='yellow-eyed penguin'
```

12. En supposant que vous êtes `user2`, utilisez `mkdir` pour créer un dossier nommé `bin` dans votre répertoire personnel :

```
mkdir ~/bin
```

ou bien

```
mkdir /home/user2/bin
```

ou alors

```
mkdir $HOME/bin
```

13. Tapez la commande pour ajouter le dossier `~/bin` à votre `PATH` afin qu'il soit le premier répertoire dans lequel `bash` recherche des binaires :

```
PATH="$HOME/bin:$PATH"
```

`PATH=~/bin:$PATH` ou `PATH=/home/user2/bin:$PATH` sont également valables.

14. Pour assurer que la valeur de `PATH` reste inchangée après un redémarrage, quel bout de code — sous forme d'une instruction `if` — devriez-vous ajouter dans `~/.profile` ?

```
if [ -d "$HOME/bin" ] ; then  
    PATH="$HOME/bin:$PATH"
```

```
fi
```

Réponses aux exercices d'approfondissement

1. `let` : plus que l'évaluation d'une expression arithmétique :

- Faites une recherche dans le manuel en ligne ou sur le web à propos de `let` et son fonctionnement pour la définition des variables et créez une nouvelle variable locale nommée `my_val` dont la valeur est 10 — en additionnant 5 et 5 :

```
let "my_val = 5 + 5"
```

ou bien

```
let 'my_val = 5 + 5'
```

- Ensuite, créez une autre variable nommée `your_val` et dont la valeur est 5 — en divisant par 2 la valeur de `my_val` :

```
let "your_val = $my_val / 2"
```

ou bien

```
let 'your_val = $my_val / 2'
```

2. Le résultat d'une commande dans une variable ? Bien sûr que c'est possible ; c'est ce qu'on appelle la *substitution de commande*. Essayez en étudiant la fonction suivante, nommée `music_info` :

```
music_info(){  
latest_music=`ls -l1t ~/Music | head -n 6`  
echo -e "Your latest 5 music files:\n$latest_music"  
}
```

Le résultat de la commande `ls -l1t ~/Music | head -n 6` devient la valeur de la variable `latest_music`. Ensuite, la variable `latest_music` est référencée dans la commande `echo` (qui affiche le nombre total d'octets occupés par le dossier `Music` et les cinq derniers fichiers musicaux stockés dans le dossier `Music` — un par ligne).

Laquelle des commandes suivantes est un remplacement valide pour


```
latest_music=`ls -l1t ~/Music | head -n 6`
```

C'est l'option A :

```
latest_music=$(ls -l1t ~/Music| head -n 6)
```



105.1 Leçon 3

Certification :	LPIC-1
Version :	5.0
Thème :	105 Shells et scripts shell
Objectif :	105.1 Personnaliser et utiliser l'environnement shell
Leçon :	3 sur 3

Introduction

Nous avons abordé les shells, les scripts de démarrage et les variables dans les leçons précédentes, et nous allons compléter la personnalisation du shell en regardant de près deux éléments importants du shell : les *alias* et les *fonctions*. En fait, c'est l'ensemble des variables, des alias et des fonctions — avec leurs interactions mutuelles — qui constitue l'environnement du shell.

Le point fort de ces deux éléments du shell, qui permettent de gagner en souplesse et en temps, est lié au concept d'encapsulation : ils offrent la possibilité de rassembler — sous une seule commande — une série de commandes répétitives ou récurrentes.

Créer des alias

Un alias est un nom de substitution pour une ou plusieurs autres commandes. Il peut être exécuté comme une commande normale, mais il exécute une autre commande en fonction de la définition de l'alias.

La syntaxe pour déclarer un alias est assez simple. Un alias est déclaré en écrivant le mot-clé

`alias` suivi de l'affectation de l'alias. Cette dernière se compose du *nom de l'alias*, d'un *signe égal* et d'une ou plusieurs *commandes* :

```
alias alias_name=command(s)
```

Par exemple :

```
$ alias oldshell=sh
```

Cet alias peu commode va lancer une instance du shell ancestral `sh` lorsque l'utilisateur tape `oldshell` dans le terminal :

```
$ oldshell
$
```

La puissance des alias réside dans le fait qu'ils nous permettent d'écrire des versions courtes de commandes longues à taper :

```
$ alias ls='ls --color=auto'
```

NOTE Pour en savoir plus sur `ls` et ses couleurs, tapez `man dir_colors` dans le terminal.

De même, nous pouvons créer un alias pour une série de commandes concaténées—le point-virgule (;) sera utilisé comme délimiteur. Nous pouvons, par exemple, avoir un alias qui nous donne des informations sur l'emplacement de l'exécutable `git` ainsi que sa version :

```
$ alias git_info='which git;git --version'
```

Pour invoquer un alias, il suffit de taper son nom dans le terminal :

```
$ git_info
/usr/bin/git
git version 2.7.4
```

La commande `alias` affiche la liste de tous les alias disponibles sur le système :

```
$ alias
```

```
alias git-info='which git;git --version'
alias ls='ls --color=auto'
alias oldshell='sh'
```

La commande `unalias` supprime un alias. Nous pouvons, par exemple, invoquer `unalias git_info` et le voir disparaître de la liste :

```
$ unalias git_info
$ alias
alias ls='ls --color=auto'
alias oldshell='sh'
```

Comme nous l'avons vu avec la commande `alias hi='echo We salute you.'` dans une leçon précédente, nous devons entourer les commandes de guillemets (simples ou doubles) lorsqu'elles contiennent des espaces en raison de la présence d'arguments ou de paramètres :

```
$ alias greet='echo Hello world!'
$ greet
Hello world!
```

Les commandes avec des espaces comprennent également celles avec des options :

```
$ alias ll='ls -al'
```

Maintenant `ll` va afficher tous les fichiers — y compris les fichiers cachés (`a`) — dans le format long (`l`).

Nous pouvons référencer des variables dans un alias :

```
$ reptile=uromastyx
$ alias greet='echo Hello $reptile!'
$ greet
Hello uromastyx!
```

La variable peut également être affectée à l'intérieur de l'alias :

```
$ alias greet='reptile=tortoise; echo Hello $reptile!'
$ greet
```

```
Hello tortoise!
```

Nous pouvons échapper un alias avec `\` :

```
$ alias where?='echo $PWD'
$ where?
/home/user2
$ \where?
-bash: where?: command not found
```

Échapper un alias est utile lorsqu'un alias porte le même nom qu'une commande normale. Dans ce cas, l'alias a la priorité sur la commande originale, qui reste toutefois accessible en échappant l'alias.

De même, nous pouvons placer un alias à l'intérieur d'un autre alias :

```
$ where?
/home/user2
$ alias my_home=where?
$ my_home
/home/user2
```

Par ailleurs, nous pouvons également placer une fonction à l'intérieur d'un alias, comme vous allez le voir ci-dessous.

Expansion et interprétation des guillemets dans les alias

Pour les variables d'environnement, les guillemets simples rendent l'expansion dynamique :

```
$ alias where?='echo $PWD'
$ where?
/home/user2
$ cd Music
$ where?
/home/user2/Music
```

En revanche, l'expansion s'effectue de manière statique avec les guillemets doubles :

```
$ alias where?="echo $PWD"
$ where?
```

```
/home/user2
$ cd Music
$ where?
/home/user2
```

Persistance des alias : scripts de démarrage

Tout comme pour les variables, les alias doivent être rangés dans des scripts d'initialisation sourcés au démarrage pour les rendre persistants. Comme nous l'avons vu, un fichier approprié pour ranger les alias des utilisateurs est `~/.bashrc`. Vous y trouverez probablement déjà quelques alias prédéfinis (la plupart d'entre eux sont commentés et prêts à l'emploi dès que vous supprimez le `# initial`) :

```
$ grep alias .bashrc
# enable color support of ls and also add handy aliases
  alias ls='ls --color=auto'
  #alias dir='dir --color='
  #alias vdir='vdir --color='
  #alias grep='grep --color='
  #alias fgrep='fgrep --color='
  #alias egrep='egrep --color='
# some more ls aliases
#ll='ls -al'
#alias la='ls -A'
#alias l='ls -CF'
# ~/.bash_aliases, instead of adding them here directly.
if [ -f ~/.bash_aliases ]; then
  . ~/.bash_aliases
```

Comme vous pouvez le voir dans les trois dernières lignes, on nous offre la possibilité d'avoir notre propre fichier dédié aux alias — `~/.bash_aliases` — et de le faire sourcer par `.bashrc` à chaque démarrage du système. Nous allons choisir cette option et éditer ce fichier :

```
#####
# .bash_aliases:
# a file to be populated by the user's personal aliases (and sourced by ~/.bashrc).
#####
alias git_info='which git;git --version'
alias greet='echo Hello world!'
alias ll='ls -al'
alias where?='echo $PWD'
```

Créer des fonctions

A la différence des alias, les fonctions sont plus programmatiques et plus flexibles, en particulier lorsqu'il s'agit d'exploiter tout le potentiel des variables spéciales de Bash et des paramètres positionnels. Elles sont également idéales pour travailler avec des structures de contrôle de flux telles que les boucles ou les tests conditionnels. Nous pouvons considérer une fonction comme une commande qui inclut de la logique à travers des blocs ou un ensemble d'autres commandes.

Deux syntaxes pour créer une fonction

Il existe deux syntaxes valables pour définir une fonction.

En utilisant le mot clé `function`

D'une part, on peut utiliser le mot-clé `function` suivi du nom de la fonction et des commandes entre accolades :

```
function function_name {  
  command #1  
  command #2  
  command #3  
  .  
  .  
  .  
  command #n  
}
```

En utilisant `()`

D'autre part, nous pouvons omettre le mot-clé `function` et le remplacer par deux parenthèses juste après le nom de la fonction :

```
function_name() {  
  command #1  
  command #2  
  command #3  
  .  
  .  
  .  
  command #n  
}
```

Il est courant de mettre les fonctions dans des fichiers ou des scripts. Elles peuvent également être écrites directement dans l'invite du shell, avec chaque commande sur une ligne différente — notez `PS2(>)` qui indique une nouvelle ligne après un retour à la ligne :

```
$ greet() {  
> greeting="Hello world!"  
> echo $greeting  
> }
```

Quoi qu'il en soit, et quelle que soit la syntaxe choisie, si l'on décide de ne pas faire de retour à la ligne et d'écrire une fonction en une seule ligne, les commandes doivent être séparées par des points-virgules (notez également le point-virgule après la dernière commande) :

```
$ greet() { greeting="Hello world!"; echo $greeting; }
```

`bash` ne s'est pas manifesté lorsque nous avons appuyé sur Entrée, notre fonction est donc prête à être invoquée. Pour invoquer une fonction, nous pouvons taper son nom dans le terminal :

```
$ greet  
Hello world!
```

Tout comme pour les variables et les alias, si nous voulons que les fonctions soient persistantes après un redémarrage du système, nous devons les placer dans des scripts d'initialisation du shell tels que `/etc/bash.bashrc` (global) ou `~/ .bashrc` (local).

WARNING

Une fois que vous avez ajouté des alias ou des fonctions à un script de démarrage, vous devez sourcer ces fichiers avec `.` ou `source` pour que les changements prennent effet si vous voulez éviter de vous déconnecter et de vous reconnecter ou de redémarrer le système.

Variables spéciales intégrées à Bash

Le shell Bourne comporte un ensemble de variables spéciales qui s'avèrent très utiles pour les fonctions et les scripts. Elles sont spéciales parce qu'elles peuvent seulement être référencées — mais pas affectées. Voici la liste des variables spéciales les plus importantes :

\$?

La référence de cette variable renvoie vers le résultat de la dernière commande exécutée. Une valeur de `0` équivaut à un succès :


```
$ ps aux | grep bash
user2      420  0.0  0.4  21156  5012 pts/0    Ss   17:10   0:00 -bash
user2      640  0.0  0.0  12784   936 pts/0    S+   18:04   0:00 grep bash
$ echo $?
0
```

Une valeur différente de 0 correspond à une erreur :

```
user1@debian:~$ ps aux |rep bash
-bash: rep: command not found
user1@debian:~$ echo $?
127
```

\$\$

Correspond au PID (identifiant du processus ou *process ID*) du shell :

```
$ ps aux | grep bash
user2      420  0.0  0.4  21156  5012 pts/0    Ss   17:10   0:00 -bash
user2      640  0.0  0.0  12784   936 pts/0    S+   18:04   0:00 grep bash
$ echo $$
420
```

\$!

Correspond au PID de la dernière tâche exécutée en arrière-plan :

```
$ ps aux | grep bash &
[1] 663
$ user2      420  0.0  0.4  21156  5012 pts/0    Ss+  17:10   0:00 -bash
user2      663  0.0  0.0  12784   972 pts/0    S    18:08   0:00 grep bash
^C
[1]+  Done                  ps aux | grep bash
$ echo $!
663
```

NOTE

Rappelez-vous que l'esperluette (&) est utilisée pour lancer des processus en arrière-plan.

Paramètres positionnels \$0 à \$9

Ils renvoient les paramètres et les arguments passés à la fonction (alias ou script)—\$0

correspond au nom du script ou du shell.

Nous allons créer une fonction pour faire une démonstration des paramètres positionnels — notez PS2 (>) qui indique les nouvelles lignes après un retour à la ligne :

```
$ special_vars() {  
> echo $0  
> echo $1  
> echo $2  
> echo $3  
}
```

Nous allons maintenant appeler la fonction (`special_vars`) en lui passant trois paramètres (`debian`, `ubuntu`, `zorin`) :

```
$ special_vars debian ubuntu zorin  
-bash  
debian  
ubuntu  
zorin
```

Ça a marché comme prévu.

WARNING

Même s'il est techniquement possible de passer des paramètres positionnels aux alias, ce n'est pas très pratique étant donné qu'avec les alias, les paramètres positionnels sont toujours passés à la fin :

```
$ alias great_editor='echo $1 is a great text editor'  
$ great_editor emacs  
is a great text editor emacs
```

Voici d'autres variables spéciales intégrées à Bash :

`$#`

Correspond au nombre d'arguments passés à la commande.

`$@`, `$*`

Correspond aux arguments passés à la commande.

\$_

Correspond au dernier paramètre ou au nom du script (entre autres choses ; voir `man bash` pour en savoir plus !):

Les variables dans les fonctions

Bien évidemment, les variables peuvent être utilisées à l'intérieur des fonctions.

Pour le prouver, nous allons créer un nouveau fichier appelé `funed` avec la fonction suivante :

```
editors() {
    editor=emacs

    echo "My editor is: $editor. $editor is a fun text editor."
}
```

Comme vous l'avez peut-être deviné, nous devons d'abord sourcer le fichier avant de pouvoir invoquer la fonction :

```
$ . funed
```

Et maintenant, nous pouvons le tester :

```
$ editors
My editor is emacs. emacs is a fun text editor.
```

Comme vous pouvez le deviner, la variable `editor` doit d'abord être définie pour que la fonction `editors` fonctionne correctement. La portée de cette variable est locale au shell en cours et nous pouvons la référencer tout au long de la session :

```
$ echo $editor
emacs
```

Parallèlement aux variables locales, nous pouvons également utiliser des variables d'environnement dans notre fonction :

```
editors() {
```

```
editor=emacs

echo "The text editor of $USER is: $editor."
}

editors
```

Notez que cette fois-ci, nous avons décidé d'appeler la fonction depuis le fichier lui-même (`editors` à la dernière ligne). Ainsi, lorsque nous sourçons le fichier, la fonction est appelée d'une traite :

```
$ . funed
The text editor of user2 is: emacs.
```

Les paramètres positionnels dans les fonctions

C'est un peu le même principe pour les paramètres positionnels.

Nous pouvons les passer à des fonctions à l'intérieur du fichier ou du script (notez la dernière ligne : `editors tortoise`):

```
editors() {

editor=emacs

echo "The text editor of $USER is: $editor."
echo "Bash is not a $1 shell."
}

editors tortoise
```

Nous sourçons le fichier et nous voyons qu'il fonctionne :

```
$ . funed
The text editor of user2 is: emacs.
Bash is not a tortoise shell.
```

Nous pouvons également passer des paramètres positionnels aux fonctions en ligne de commande. Pour preuve, nous supprimons la dernière ligne du fichier :

```
editors() {  
  
    editor=emacs  
  
    echo "The text editor of $USER is: $editor."  
    echo "Bash is not a $1 shell."  
}
```

Ensuite, nous devons sourcer le fichier :

```
$ . funed
```

Et enfin, nous invoquons la fonction avec `tortoise` comme paramètre positionnel `$1` en ligne de commande :

```
$ editors tortoise  
The text editor of user2 is: emacs.  
Bash is not a tortoise shell.
```

Les fonctions dans les scripts

Les fonctions figurent principalement dans les scripts Bash.

La conversion de notre fichier `funed` en un script (que nous appellerons `funed.sh`) est un vrai jeu d'enfant :

```
#!/bin/bash  
  
editors() {  
  
    editor=emacs  
  
    echo "The text editor of $USER is: $editor."  
    echo "Bash is not a $1 shell."  
}  
  
editors tortoise
```

Et c'est tout ! Nous avons juste ajouté deux lignes :

- La première ligne est le *shebang* et définit le programme qui est censé interpréter le script : `#!/bin/bash`. Vous ne serez pas étonnés d'apprendre que c'est `bash` lui-même.
- La dernière ligne correspond simplement à l'invocation de la fonction.

Il ne reste plus qu'une chose à faire : rendre le script exécutable :

```
$ chmod +x funed.sh
```

Et le voilà prêt à être exécuté :

```
$ ./funed.sh
The text editor of user2 is: emacs.
Bash is not a tortoise shell.
```

NOTE Vous apprendrez tout sur les scripts shell dans les leçons à venir.

Une fonction au sein d'un alias

Comme nous l'avons dit plus haut, nous pouvons définir une fonction à l'intérieur d'un alias :

```
$ alias great_editor='gr8_ed() { echo $1 is a great text editor; unset -f gr8_ed; }; gr8_ed'
```

Cet alias complexe mérite une explication. Voyons ce qu'il en est :

- Tout d'abord, il y a la fonction en elle-même : `gr8_ed() { echo $1 is a great text editor; unset -f gr8_ed; }`
- La dernière commande de la fonction — `unset -f gr8_ed` — désactive la fonction pour qu'elle ne subsiste pas dans la session `bash` actuelle après l'appel de l'alias.
- Enfin, pour que l'invocation de l'alias réussisse, nous devons d'abord invoquer la fonction en question : `gr8_ed`.

Nous allons invoquer l'alias et vérifier qu'il fonctionne :

```
$ great_editor emacs
emacs is a great text editor
```

Comme le montre `unset -f gr8_ed` ci-dessus, la commande `unset` n'est pas seulement utilisée pour désactiver des variables, mais aussi des fonctions. En fait, il existe des options spécifiques :

unset -v

pour les variables

unset -f

pour les fonctions

Si la commande est invoquée sans options, `unset` essaiera d'abord de désactiver une variable et — si l'opération échoue — elle essaiera dans un deuxième temps de désactiver une fonction.

Une fonction imbriquée dans une fonction

Imaginons maintenant que nous voulions communiquer deux choses à l'utilisatrice `user2` à chaque fois qu'elle se connecte au système :

- Dire bonjour et chanter les louanges d'un éditeur de texte.
- Étant donné qu'elle commence à enregistrer un nombre important de fichiers vidéo Matroska dans son dossier `$HOME/Video`, nous voulons également lui afficher un avertissement.

Pour ce faire, nous avons implémenté les deux fonctions suivantes dans `/home/user2/.bashrc` :

La première fonction (`check_vids`) vérifie les fichiers `.mkv` et affiche l'avertissement :

```
check_vids() {
    ls -l ~/Video/*.mkv > /dev/null 2>&1
    if [ "$?" = "0" ];then
        echo -e "Remember, you must not keep more than 5 video files in your Video
folder.\nThanks."
    else
        echo -e "You do not have any videos in the Video folder. You can keep up to 5.\nThanks."
    fi
}
```

La fonction `check_vids` fait trois choses :

- Elle établit la liste des fichiers `mkv` dans `~/Video` en envoyant le résultat — et toutes les erreurs éventuelles — au *paradis des octets* (`/dev/null`).
- Elle vérifie si la commande précédente a été exécutée avec succès.
- Elle affiche l'un ou l'autre message en fonction du résultat.

La seconde fonction est une version modifiée de notre fonction `editors` :

```
editors() {  
  
    editor=emacs  
  
    echo "Hi, $USER!"  
    echo "$editor is more than a text editor!"  
  
    check_vids  
}  
  
editors
```

Notons deux choses importantes ici :

- La dernière commande de `editors` invoque `check_vids` de sorte que les deux fonctions s'enchaînent : La salutation, l'éloge, la vérification et l'avertissement sont exécutés successivement.
- `editors` est lui-même le point d'entrée de la séquence de fonctions, il est donc invoqué dans la dernière ligne (`editors`).

Maintenant, nous pouvons nous connecter en tant que `user2` et voir si ça fonctionne :

```
# su - user2  
Hi, user2!  
emacs is more than a text editor!  
Remember, you must not keep more than 5 video files in your Video folder.  
Thanks.
```


Exercices guidés

1. Complétez le tableau par "Oui" ou "Non" en considérant les possibilités des alias et des fonctions :

Caractéristique	Alias ?	Fonctions ?
Les variables locales peuvent être utilisées		
Les variables d'environnement peuvent être utilisées		
Peuvent être échappés avec \		
Peuvent être récursifs		
Très efficaces en conjonction avec les paramètres positionnels		

2. Tapez la commande qui affiche la liste de tous les alias de votre système :

3. Écrivez un alias nommé `logg` qui affiche tous les fichiers `ogg` dans `~/Music` — un par ligne :

4. Invoquez l'alias pour montrer qu'il fonctionne :

5. Maintenant, modifiez l'alias de manière à afficher l'utilisateur de la session et un deux-points avant la liste :

6. Invoquez-le à nouveau pour montrer que cette nouvelle mouture fonctionne tout aussi bien :

7. Affichez à nouveau la liste de tous les alias et vérifiez que l'alias `logg` apparaît bien dans la liste :

8. Supprimez l'alias :

9. Examinez les colonnes “Nom de l’alias” et “Commande(s) aliasée(s)” et définissez correctement les alias en fonction des valeurs qui leur sont attribuées :

Nom de l’alias	Commande(s) aliasée(s)	Définition de l’alias
b	bash	
bash_info	which bash + echo "\$BASH_VERSION"	
kernel_info	uname -r	
greet	echo Hi, \$USER!	
computer	pc=slimbook + echo My computer is a \$pc	

10. En tant que `root`, inscrivez une fonction appelée `my_fun` dans `/etc/bash.bashrc`. La fonction doit saluer l’utilisateur et lui afficher son `PATH`. Invoquez-la pour que l’utilisateur voie apparaître les deux messages à chaque fois qu’il se connecte :

11. Connectez-vous en tant que `user2` pour vérifier si ça fonctionne :

12. Écrivez la même fonction en une seule ligne :

13. Invoquez la fonction :

14. Désactivez la fonction :

15. Voici une version modifiée de la fonction `special_vars` :

```
$ special_vars2() {
> echo $#
> echo $_
> echo $1
> echo $4
> echo $6
> echo $7
```

```
> echo $_
> echo $@
> echo $?
> }
```

Et voici la commande que nous utilisons pour l'appeler :

```
$ special_vars2 crying cockles and mussels alive alive oh
```

Devinez les résultats :

Référence	Valeur
echo \$#	
echo \$_	
echo \$1	
echo \$4	
echo \$6	
echo \$7	
echo \$_	
echo \$@	
echo \$?	

16. En vous basant sur la fonction donnée en exemple (`check_vids`) dans la section “Une fonction imbriquée dans une fonction”, écrivez une fonction nommée `check_music` à inclure dans un script de démarrage `bash` qui accepte les paramètres positionnels de manière à ce que nous puissions modifier facilement :

- le type de fichier à vérifier : `ogg`
- le répertoire où les fichiers sont enregistrés : `~/Music`
- le type de fichier conservé : `music`
- le nombre de fichiers enregistrés : `7`

Exercices d'approfondissement

1. Les fonctions en lecture seule (readonly) sont celles dont on ne peut pas modifier le contenu. Faites une recherche sur les *fonctions en lecture seule* et complétez le tableau suivant :

Nom de la fonction	Basculer en lecture seule	Afficher toutes les fonctions en lecture seule
my_fun		

2. Cherchez sur le web comment modifier PS1 et tout ce qui peut être utile pour écrire une fonction appelée `fyi` (à placer dans un script de démarrage) qui donne les informations suivantes à l'utilisateur :

- le nom de l'utilisateur
- son répertoire personnel
- le nom d'hôte
- le type de système d'exploitation
- le PATH pour les exécutables
- l'emplacement des e-mails
- la fréquence de vérification des e-mails
- le niveau d'imbrication du shell en cours
- l'invite (modifiez-la de manière à ce qu'elle affiche `<utilisateur>@<hôte-date>`)

Résumé

Voici ce que nous avons vu dans cette leçon :

- Les alias et les fonctions sont des fonctionnalités importantes du shell qui nous permettent d'encapsuler des blocs de code récurrents.
- Les alias sont pratiques pour avoir une version plus courte d'une commande longue ou compliquée.
- Les fonctions sont des procédures qui implémentent la logique et qui nous permettent d'automatiser des tâches, en particulier lorsqu'elles sont utilisées dans des scripts.
- La syntaxe pour définir des alias et des fonctions.
- Concaténer plusieurs commandes au moyen du point-virgule (;).
- Le bon usage des guillemets avec les alias.
- Comment rendre les alias et les fonctions persistants.
- Les variables spéciales intégrées à Bash : `$?`, `$$`, `$!`, les paramètres positionnels (`$0-$9`), `$#`, `$@`, `$*` et `$_`.
- Comment utiliser les variables et les paramètres positionnels avec les fonctions.
- Comment utiliser les fonctions dans les scripts.
- Comment invoquer une fonction à partir d'un alias.
- Comment invoquer une fonction à partir d'une autre fonction.
- Les bases pour créer un script `bash`.

Commandes et mots-clés utilisés dans cette leçon :

alias

Créer un alias.

unalias

Supprimer un alias.

cd

Changer de répertoire.

grep

Afficher les lignes qui correspondent à un motif.

function

Mot-clé du shell pour créer une fonction.

.

Sourcer un fichier.

source

Sourcer un fichier.

ps

Afficher un aperçu des processus en cours.

echo

Afficher une ligne de texte.

chmod

Modifier les permissions d'un fichier, pour le rendre exécutable par exemple.

unset

Réinitialiser une variable ou une fonction.

su

Changer d'utilisateur ou devenir superutilisateur.

Réponses aux exercices guidés

1. Complétez le tableau par "Oui" ou "Non" en considérant les possibilités des alias et des fonctions :

Caractéristique	Alias ?	Fonctions ?
Les variables locales peuvent être utilisées	Oui	Oui
Les variables d'environnement peuvent être utilisées	Oui	Oui
Peuvent être échappés avec \	Oui	Non
Peuvent être récursifs	Oui	Oui
Très efficaces en conjonction avec les paramètres positionnels	Non	Oui

2. Tapez la commande qui affiche la liste de tous les alias de votre système :

```
alias
```

3. Écrivez un alias nommé `logg` qui affiche tous les fichiers `ogg` dans `~/Music` — un par ligne :

```
alias logg='ls -1 ~/Music/*ogg'
```

4. Invoquez l'alias pour montrer qu'il fonctionne :

```
logg
```

5. Maintenant, modifiez l'alias de manière à afficher l'utilisateur de la session et un deux-points avant la liste :

```
alias logg='echo $USER;; ls -1 ~/Music/*ogg'
```

6. Invoquez-le à nouveau pour montrer que cette nouvelle mouture fonctionne tout aussi bien :

```
logg
```

7. Affichez à nouveau la liste de tous les alias et vérifiez que l'alias `logg` apparaît bien dans la liste :

```
alias
```

8. Supprimez l'alias :

```
unalias logg
```

9. Examinez les colonnes “Nom de l'alias” et “Commande(s) aliasée(s)” et définissez correctement les alias en fonction des valeurs qui leur sont attribuées :

Nom de l'alias	Commande(s) aliasée(s)	Définition de l'alias
<code>b</code>	<code>bash</code>	<code>alias b=bash</code>
<code>bash_info</code>	<code>which bash + echo "\$BASH_VERSION"</code>	<code>alias bash_info='which bash; echo "\$BASH_VERSION" '</code>
<code>kernel_info</code>	<code>uname -r</code>	<code>alias kernel_info='uname -r'</code>
<code>greet</code>	<code>echo Hi, \$USER!</code>	<code>alias greet='echo Hi, \$USER'</code>
<code>computer</code>	<code>pc=slimbook + echo My computer is a \$pc</code>	<code>alias computer='pc=slimbook; echo My computer is a \$pc'</code>

NOTE Les guillemets simples peuvent être remplacés par des guillemets doubles.

10. En tant que `root`, inscrivez une fonction appelée `my_fun` dans `/etc/bash.bashrc`. La fonction doit saluer l'utilisateur et lui afficher son `PATH`. Invoquez-la pour que l'utilisateur voie apparaître les deux messages à chaque fois qu'il se connecte :

Option A :

```
my_fun() {
```



```
echo Hello, $USER!  
echo Your path is: $PATH  
}  
my_fun
```

Option B :

```
function my_fun {  
echo Hello, $USER!  
echo Your path is: $PATH  
}  
my_fun
```

11. Connectez-vous en tant que `user2` pour vérifier si ça fonctionne :

```
su - user2
```

12. Écrivez la même fonction en une seule ligne :

Option A :

```
my_fun() { echo "Hello, $USER!"; echo "Your path is: $PATH"; }
```

Option B :

```
function my_fun { echo "Hello, $USER!"; echo "Your path is: $PATH"; }
```

13. Invoquez la fonction :

```
my_fun
```

14. Désactivez la fonction :

```
unset -f my_fun
```

15. Voici une version modifiée de la fonction `special_vars` :

```
$ special_vars2() {  
> echo $#  
> echo $_  
> echo $1  
> echo $4  
> echo $6  
> echo $7  
> echo $_  
> echo $@  
> echo $?  
> }
```

Et voici la commande que nous utilisons pour l'appeler :

```
$ special_vars2 crying cockles and mussels alive alive oh
```

Devinez les résultats :

Référence	Valeur
echo \$#	7
echo \$_	7
echo \$1	crying
echo \$4	mussels
echo \$6	alive
echo \$7	oh
echo \$_	oh
echo \$@	crying cockles and mussels alive alive oh
echo \$?	0

16. En vous basant sur la fonction donnée en exemple (`check_vids`) dans la section “Une fonction imbriquée dans une fonction”, écrivez une fonction nommée `check_music` à inclure dans un script de démarrage `bash` qui accepte les paramètres positionnels de manière à ce que nous puissions modifier facilement :
- le type de fichier à vérifier : `ogg`
 - le répertoire où les fichiers sont enregistrés : `~/Music`

- le type de fichier conservé : `music`
- le nombre de fichiers enregistrés : `7`

```
check_music() {  
    ls -1 ~/$1/*. $2 > ~/.mkv.log 2>&1  
    if [ "$?" = "0" ];then  
        echo -e "Remember, you must not keep more than $3 $4 files in your $1  
folder.\nThanks."  
    else  
        echo -e "You do not have any $4 files in the $1 folder. You can keep up to  
$3.\nThanks."  
    fi  
}  
  
check_music Music ogg 7 music
```

Réponses aux exercices d'approfondissement

1. Les fonctions en lecture seule (readonly) sont celles dont on ne peut pas modifier le contenu. Faites une recherche sur les *fonctions en lecture seule* et complétez le tableau suivant :

Nom de la fonction	Basculer en lecture seule	Afficher toutes les fonctions en lecture seule
my_fun	readonly -f my_fun	readonly -f

2. Cherchez sur le web comment modifier PS1 et tout ce qui peut être utile pour écrire une fonction appelée `fyi` (à placer dans un script de démarrage) qui donne les informations suivantes à l'utilisateur :

- le nom de l'utilisateur
- son répertoire personnel
- le nom d'hôte
- le type de système d'exploitation
- le PATH pour les exécutables
- l'emplacement des e-mails
- la fréquence de vérification des e-mails
- le niveau d'imbrication du shell en cours
- l'invite (modifiez-la de manière à ce qu'elle affiche `<utilisateur>@<hôte-date>`)

```
fyi() {
    echo -e "For your Information:\n
    Username: $USER
    Home directory: $HOME
    Host: $HOSTNAME
    Operating System: $OSTYPE
    Path for executable files: $PATH
    Your mail directory is $MAIL and is searched every $MAILCHECK seconds.
    The current level of your shell is: $SHLVL"
    PS1="\u@\h-\d "
}

fyi
```



Linux
Professional
Institute

105.2 Personnalisation ou écriture de scripts simples

Référence aux objectifs de LPI

LPIC-1 5.0, Exam 102, Objective 105.2

Valeur

4

Domaines de connaissance les plus importants

- Utilisation de la syntaxe standard du shell sh (boucles, tests).
- Utilisation de la substitution de commandes.
- Test de la valeur de retour d'une fonction indiquant la réussite, l'échec ou d'autres informations.
- Exécution de commandes chaînées.
- Envoi conditionnel de courriels au superutilisateur.
- Sélection correcte de l'interpréteur de commandes à utiliser dans l'entête du script (#!).
- Gestion de l'emplacement, des propriétés, des droits d'exécution et les droits spéciaux (suid) des scripts.

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

- `for`
- `while`
- `test`
- `if`
- `read`
- `seq`

- `exec`
- `||`
- `&&`



105.2 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	105 Shells et scripts shell
Objectif :	105.2 Personnaliser ou écrire des scripts simples
Leçon :	1 sur 2

Introduction

L'environnement shell de Linux permet d'utiliser des fichiers — appelés *scripts* — qui contiennent des commandes de n'importe quel programme disponible dans le système en combinaison avec les commandes intégrées au shell afin d'automatiser les tâches personnalisées de l'utilisateur et/ou du système. En effet, bon nombre des tâches de maintenance du système d'exploitation sont effectuées par des scripts constitués de séquences de commandes, de structures décisionnelles et de boucles conditionnelles. Bien que les scripts soient généralement destinés à des tâches liées au système d'exploitation lui-même, ils sont également utiles pour des tâches orientées utilisateur comme le renommage en masse de fichiers, la collecte et l'analyse de données ou toute autre activité répétitive en ligne de commande.

Les scripts ne sont rien d'autre que des fichiers texte qui se comportent comme des programmes. Un vrai programme — l'interpréteur — lit et exécute les instructions qui figurent dans le script. L'interpréteur peut également lancer une session interactive dans laquelle les commandes — y compris les scripts — sont lues et exécutées au fur et à mesure qu'elles sont saisies, comme c'est le cas dans les sessions du shell Linux. Les scripts permettent de regrouper ces instructions et ces commandes lorsqu'elles deviennent trop complexes pour être implémentées sous forme d'alias ou

de fonctions shell personnalisées. En outre, les scripts peuvent être maintenus comme des programmes conventionnels et, dans la mesure où il ne s'agit que de simples fichiers texte, ils peuvent être créés et modifiés à l'aide de n'importe quel éditeur de texte courant.

Structure et exécution d'un script

En principe, un script est une séquence ordonnée de commandes qui doivent être exécutées par l'interpréteur de commandes correspondant. La façon dont un interpréteur lit un script peut varier et il y a plusieurs façons de le faire dans une session du shell Bash, mais l'interpréteur par défaut pour un script sera celui indiqué dans la première ligne du script, juste après les caractères `#!` (connus sous le nom de *shebang*). Dans un script qui contient des instructions pour l'interpréteur de commandes Bash, la première ligne doit être `#!/bin/bash`. En indiquant cette ligne, l'interpréteur pour toutes les instructions du fichier sera `/bin/bash`. À l'exception de la première ligne, toutes les autres lignes qui commencent par le caractère dièse `#` seront ignorées, de sorte qu'elles peuvent être utilisées pour insérer des remarques et des commentaires. Les lignes vides sont également ignorées. Un script shell très basique peut donc s'écrire comme ceci :

```
#!/bin/bash

# A very simple script

echo "Cheers from the script file! Current time is: "

date +%H:%M
```

Ce script n'a que deux instructions pour l'interpréteur `/bin/bash` : la primitive du shell `echo` et la commande `date`. La façon la plus simple d'exécuter un script consiste à exécuter l'interpréteur avec le chemin du script en argument. Ainsi, en supposant que l'exemple précédent ait été sauvegardé dans un fichier script nommé `script.sh` dans le répertoire courant, il sera lu et interprété par Bash avec la commande suivante :

```
$ bash script.sh
Cheers from the script file! Current time is:
10:57
```

La commande `echo` ajoutera automatiquement un retour à la ligne après avoir affiché le contenu, mais l'option `-n` supprimera ce comportement. Ainsi, l'utilisation de `echo -n` dans le script fera apparaître le résultat des deux commandes sur la même ligne :


```
$ bash script.sh
```

```
Cheers from the script file! Current time is: 10:57
```

Bien que ce ne soit pas obligatoire, le suffixe `.sh` permet d'identifier les scripts shell lors de l'affichage et de la recherche de fichiers.

TIP

Bash appellera la commande indiquée après le `#!` en tant qu'interpréteur du fichier script. Il peut être utile, par exemple, d'utiliser le *shebang* pour d'autres langages scriptés comme *Python* (`#!/usr/bin/python`), *Perl* (`#!/usr/bin/perl`) ou *awk* (`#!/usr/bin/awk`).

Si le script est censé être exécuté par d'autres utilisateurs du système, il est important de vérifier que les permissions en lecture sont correctes. La commande `chmod o+r script.sh` accordera la permission en lecture à tous les utilisateurs du système, ce qui leur permettra d'exécuter `script.sh` en indiquant le chemin vers le script comme argument de la commande `bash`. Alternativement, le script peut avoir les permissions d'exécution définies de façon à ce que le fichier puisse être exécuté comme une commande conventionnelle. Les droits d'exécution sur le script sont activés avec la commande `chmod +x` :

```
$ chmod +x script.sh
```

Si les droits en exécution sont définis, le script nommé `script.sh` dans le répertoire courant peut être exécuté directement avec la commande `./script.sh`. Par ailleurs, les scripts rangés dans un des répertoires figurant dans la variable d'environnement `PATH` seront accessibles sans leur chemin d'accès complet.

WARNING

Un script qui exécute des actions restreintes peut avoir son autorisation SUID activée, de sorte que les utilisateurs ordinaires peuvent également exécuter le script avec les privilèges de l'administrateur. Dans ce cas, il est très important de s'assurer qu'aucun utilisateur autre que `root` n'a le droit d'écrire dans le fichier. Autrement, un utilisateur ordinaire pourrait modifier le fichier pour effectuer des opérations arbitraires et potentiellement dangereuses.

L'emplacement et l'indentation des commandes dans les scripts ne sont pas trop rigides. Chaque ligne d'un script shell sera exécutée comme une commande shell classique, dans l'ordre dans lequel la ligne apparaît dans le script, et les règles qui valent pour l'invite shell s'appliquent également à chaque ligne individuelle du script. On peut écrire plusieurs commandes sur la même ligne, séparées par des points-virgules :

```
echo "Cheers from the script file! Current time is:" ; date +%H:%M
```

Même si ce format peut s'avérer pratique dans certains cas, son utilisation est facultative, étant donné que les commandes séquentielles peuvent être écrites à raison d'une commande par ligne et qu'elles seront exécutées comme si elles avaient été séparées par des points-virgules. En d'autres termes, le point-virgule peut être remplacé par un retour à la ligne dans les scripts Bash.

Lorsqu'un script est exécuté, les commandes qu'il contient ne sont pas exécutées directement dans la session en cours, mais par un nouveau processus Bash, appelé *sous-shell* (*sub-shell*). Cette technique permet d'éviter que le script n'écrase les variables d'environnement de la session en cours et qu'il ne laisse des modifications en suspens dans cette session. Si le but consiste à exécuter le script dans la session en cours, il doit être exécuté avec `source script.sh` ou `. script.sh` (notez qu'il y a un espace entre le point et le nom du script).

Comme pour l'exécution de n'importe quelle autre commande, l'invite du shell ne réapparaîtra qu'à la fin de l'exécution du script et son code d'état de sortie sera disponible dans la variable `$?` . Pour changer ce comportement de manière à ce que le shell actuel se termine également avec le script, ce dernier — ou toute autre commande — peut être précédé de la commande `exec`. Cette commande remplacera également le code d'état de sortie de la session en cours du shell par le sien.

Les variables

Les variables dans les scripts shell se comportent de la même manière que dans les sessions interactives, étant donné que l'interpréteur est le même. Par exemple, la syntaxe `SOLUTION=42` (sans espace autour du signe égal) affectera la valeur 42 à la variable nommée `SOLUTION`. Par convention, les noms de variables sont en majuscules, mais ce n'est pas obligatoire. En revanche, les noms de variables ne peuvent pas commencer par des caractères autres que ceux de l'alphabet.

En dehors des variables classiques créées par l'utilisateur, les scripts Bash disposent également d'un ensemble de variables spéciales appelées *paramètres*. Contrairement aux variables habituelles, les noms des paramètres commencent par un caractère non alphabétique qui désigne leur fonction. Les arguments passés à un script et d'autres informations utiles sont stockés dans des paramètres comme `$0`, `$*`, `$?`, etc. où le caractère qui suit le signe du dollar indique l'information à récupérer :

`$*`

Tous les arguments passés au script.

\$@

Tous les arguments passés au script. Avec des guillemets doubles comme dans "\$@", chaque argument sera entouré de guillemets doubles.

\$#

Le nombre d'arguments.

\$0

Le nom du script.

\$!

Le PID du dernier programme exécuté.

\$\$

Le PID du shell en cours.

\$?

Code numérique de l'état de sortie de la dernière commande terminée. Pour les processus POSIX standard, une valeur numérique de 0 signifie que la dernière commande a été exécutée avec succès, ce qui s'applique également aux scripts shell.

Un *paramètre positionnel* est un paramètre désigné par un ou plusieurs chiffres, autres que le chiffre unique 0. Par exemple, la variable \$1 correspond au premier argument transmis au script (paramètre positionnel 1), \$2 correspond au second argument, et ainsi de suite. Si la position d'un paramètre est supérieure à neuf, il doit être référencé avec des accolades, comme dans \${10}, \${11}, etc.

Les variables ordinaires, en revanche, sont destinées à stocker des valeurs insérées manuellement ou les résultats générés par d'autres commandes. La commande `read`, par exemple, peut être utilisée par le script pour demander à l'utilisateur de saisir des données pendant l'exécution du script :

```
echo "Do you want to continue (y/n)?"
read ANSWER
```

La valeur retournée sera stockée dans la variable `ANSWER`. Si le nom de la variable n'est pas fourni, la variable `REPLY` sera utilisée par défaut. Il est également possible d'utiliser la commande `read` pour lire plusieurs variables en même temps :

```
echo "Type your first name and last name:"
```

```
read NAME SURNAME
```

Dans ce cas, chaque terme séparé par un espace sera assigné respectivement aux variables `NAME` et `SURNAME`. Si le nombre de termes fournis est supérieur au nombre de variables, les termes en excès seront stockés dans la dernière variable. `read` lui-même peut afficher le message à l'utilisateur avec l'option `-p`, ce qui rend la commande `echo` redondante dans ce cas :

```
read -p "Type your first name and last name:" NAME SURNAME
```

Les scripts qui exécutent des tâches système ont souvent besoin d'informations fournies par d'autres programmes. La notation avec guillemet inverse (*backtick notation*) peut être utilisée pour stocker la sortie d'une commande dans une variable :

```
$ OS=`uname -o`
```

Dans l'exemple, le résultat de la commande `uname -o` sera stocké dans la variable `OS`. On obtiendra le même résultat avec `$()` :

```
$ OS=$(uname -o)
```

La longueur d'une variable, c'est-à-dire la quantité de caractères qu'elle contient, est retournée en faisant précéder le nom de la variable d'un dièse `#`. Cette fonctionnalité nécessite toutefois l'utilisation de la syntaxe entre accolades pour indiquer la variable :

```
$ OS=$(uname -o)
$ echo $OS
GNU/Linux
$ echo ${#OS}
9
```

Bash utilise également des variables de type tableau unidimensionnel, de sorte qu'un ensemble d'éléments apparentés peut être stocké dans une seule variable. Chaque élément d'un tableau possède un index numérique, qui doit être utilisé pour écrire et lire les valeurs de l'élément correspondant. Contrairement aux variables classiques, les tableaux doivent être déclarés à l'aide de la commande Bash `declare`. Par exemple, pour déclarer une variable nommée `SIZES` comme un tableau :

```
$ declare -a SIZES
```

Les tableaux peuvent également être déclarés implicitement lorsqu'ils sont constitués à partir d'une liste prédéfinie d'éléments, en utilisant la notation entre parenthèses :

```
$ SIZES=( 1048576 1073741824 )
```

Dans l'exemple, les deux entiers de taille importante ont été stockés dans le tableau `SIZES`. Les éléments d'un tableau doivent être référencés à l'aide d'accolades et de crochets, faute de quoi Bash ne pourra pas modifier ou afficher l'élément correctement. Étant donné que les index des tableaux commencent à 0, le contenu du premier élément est dans `${SIZES[0]}`, le deuxième élément est dans `${SIZES[1]}` et ainsi de suite :

```
$ echo ${SIZES[0]}
1048576
$ echo ${SIZES[1]}
1073741824
```

Contrairement à la lecture, la modification du contenu de l'élément d'un tableau s'effectue sans les accolades (par exemple, `SIZES[0]=1048576`). Comme pour les variables classiques, la longueur d'un élément d'un tableau est retournée avec le caractère dièse (par exemple, `${#SIZES[0]}` pour la longueur du premier élément du tableau `SIZES`). Le nombre total d'éléments d'un tableau est retourné si `@` ou `*` sont utilisés comme index :

```
$ echo ${#SIZES[@]}
2
$ echo ${#SIZES[*]}
2
```

Les tableaux peuvent également être déclarés en utilisant le résultat d'une commande comme les éléments initiaux par le biais de la substitution de commande. L'exemple suivant montre comment créer un tableau Bash composé des systèmes de fichiers pris en charge par le système actuel :

```
$ FS=( $(cut -f 2 < /proc/filesystems) )
```

La commande `cut -f 2 < /proc/filesystems` affiche tous les systèmes de fichiers actuellement pris en charge par le noyau en cours d'exécution (tels qu'ils sont listés dans la

deuxième colonne du fichier `/proc/filesystems`), de sorte que le tableau `FS` contient maintenant un élément pour chaque système de fichiers pris en charge. Tout contenu texte peut être utilisé pour initialiser un tableau étant donné que, par défaut, n'importe quel terme délimité par des caractères espace, tabulation ou retour à la ligne constituera un élément du tableau.

TIP

Bash traite chaque caractère de la variable d'environnement `$IFS` (*Input Field Separator* ou séparateur de champs d'entrée) comme un délimiteur. Pour modifier le délimiteur de champ en utilisant uniquement des caractères de retour à la ligne, par exemple, la variable `IFS` doit être réinitialisée avec la commande `IFS=$'\n'`.

Les expressions arithmétiques

Bash fournit une méthode pratique pour effectuer des opérations arithmétiques sur les nombres entiers avec la commande interne `expr`. Deux variables numériques, `$VAL1` et `$VAL2` par exemple, peuvent être additionnées avec la commande suivante :

```
$ SUM=`expr $VAL1 + $VAL2`
```

La valeur résultante de l'exemple sera disponible dans la variable `$SUM`. La commande `expr` peut être remplacée par `$(( ))`, de sorte que l'exemple précédent peut être réécrit comme `SUM=$(( $VAL1 + $VAL2 ))`. Les puissances sont également autorisées avec l'opérateur double astérisque, de sorte que la déclaration de tableau précédente `SIZES=( 1048576 1073741824 )` peut être réécrite comme `SIZES=( $(1024**2) $(1024**3) )`.

La substitution de commande peut également être utilisée dans les expressions arithmétiques. Par exemple, le fichier `/proc/meminfo` contient des informations détaillées sur la mémoire du système, y compris le nombre d'octets libres dans la RAM :

```
$ FREE=$(( 1000 * `sed -nre '2s/[^[[:digit:]]//gp' < /proc/meminfo` ))
```

L'exemple montre comment la commande `sed` peut être utilisée pour analyser le contenu de `/proc/meminfo` à l'intérieur de l'expression arithmétique. La deuxième ligne du fichier `/proc/meminfo` indique la quantité de mémoire libre en milliers d'octets, l'expression arithmétique la multiplie donc par 1000 pour obtenir le nombre d'octets libres dans la RAM.

L'exécution conditionnelle

Certains scripts ne sont généralement pas destinés à exécuter la totalité des commandes du script, mais uniquement celles qui répondent à des critères prédéfinis. Par exemple, un script de

maintenance peut envoyer un message d'avertissement à l'adresse e-mail de l'administrateur seulement si l'exécution d'une commande échoue. Bash fournit des méthodes spécifiques pour évaluer le succès de l'exécution d'une commande et des structures conditionnelles générales, semblables à celles que l'on peut trouver dans les langages de programmation classiques.

Lorsqu'on sépare les commandes par `&&`, la commande de droite ne sera exécutée que si la commande de gauche n'a pas rencontré d'erreur, c'est-à-dire si son état de sortie était égal à `0` :

```
COMMAND A && COMMAND B && COMMAND C
```

Le comportement inverse se produit si les commandes sont séparées par `||`. Dans ce cas, la commande suivante ne sera exécutée que si la commande précédente a rencontré une erreur, c'est-à-dire si son code d'état de retour est différent de `0`.

La possibilité d'exécuter des commandes en fonction de conditions préalablement définies constitue l'une des principales fonctionnalités de tous les langages de programmation. La manière la plus simple d'exécuter des commandes de manière conditionnelle consiste à utiliser la commande interne `if` de Bash, qui exécute une ou plusieurs commandes uniquement si la commande fournie en argument renvoie un `0` (succès). Une autre commande, `test`, peut être utilisée pour évaluer toute une série de critères spéciaux, elle est donc souvent utilisée de pair avec `if`. Dans l'exemple suivant, le message `Confirmed: /bin/bash is executable.` sera affiché si le fichier `/bin/bash` existe et s'il est exécutable :

```
if test -x /bin/bash ; then
    echo "Confirmed: /bin/bash is executable."
fi
```

L'option `-x` fait en sorte que la commande `test` retourne un code d'état `0` seulement si le chemin spécifié est un fichier exécutable. L'exemple suivant montre une autre façon d'obtenir exactement le même résultat, puisque les crochets peuvent remplacer `test` :

```
if [ -x /bin/bash ] ; then
    echo "Confirmed: /bin/bash is executable."
fi
```

L'instruction `else` est une option de la structure `if` et peut, si elle est utilisée, définir une commande ou une séquence de commandes à exécuter si l'expression conditionnelle n'est pas vérifiée :

```
if [ -x /bin/bash ] ; then
    echo "Confirmed: /bin/bash is executable."
else
    echo "No, /bin/bash is not executable."
fi
```

Une structure `if` doit toujours se terminer par `fi`, de sorte que l'interpréteur Bash sache où se trouve la fin d'une commande conditionnelle.

Afficher les résultats d'un script

Même si le but d'un script ne concerne que des opérations sur des fichiers, il est important d'afficher les messages relatifs à la progression dans la sortie standard, de manière à ce que l'utilisateur soit au courant des éventuels problèmes et qu'il puisse éventuellement utiliser ces messages pour générer des rapports d'opération.

La commande interne `echo` de Bash est normalement utilisée pour afficher de simples chaînes de texte, mais elle offre également des fonctionnalités étendues. Avec l'option `-e`, la commande `echo` est capable d'afficher des caractères spéciaux en utilisant des séquences échappées (une séquence d'antislash désignant un caractère spécial). Par exemple :

```
#!/bin/bash

# Get the operating system's generic name
OS=$(uname -o)

# Get the amount of free memory in bytes
FREE=$(( 1000 * `sed -nre '2s/^[[:digit:]]//gp' < /proc/meminfo` ))

echo -e "Operating system:\t$OS"
echo -e "Unallocated RAM:\t$( $FREE / 1024**2 ) MB"
```

L'utilisation des guillemets est facultative lorsqu'on utilise `echo` sans option, mais elle devient nécessaire avec l'option `-e`, autrement les caractères spéciaux risquent de ne pas s'afficher correctement. Dans le script ci-dessus, les deux commandes `echo` utilisent le caractère de tabulation `\t` pour aligner le texte, ce qui donne le résultat suivant :

```
Operating system:      GNU/Linux
Unallocated RAM:      1491 MB
```


Le caractère de retour à la ligne `\n` peut être utilisé pour séparer les lignes du résultat, et l'on obtient la même chose en combinant les deux commandes `echo` en une seule :

```
echo -e "Operating system:\t$OS\nUnallocated RAM:\t$(( $FREE / 1024**2 )) MB"
```

Même si elle permet d'afficher la plupart des messages texte, la commande `echo` n'est pas forcément adaptée à l'affichage de motifs de texte plus complexes. La commande `printf` intégrée à Bash offre plus de choix sur la façon d'afficher les variables. La commande `printf` utilise le premier argument comme format de sortie, et les caractères de remplissage seront remplacés par les arguments successifs dans l'ordre où ils apparaissent dans la ligne de commande. Par exemple, le message de l'exemple précédent pourrait être généré avec la commande `printf` suivante :

```
printf "Operating system:\t%s\nUnallocated RAM:\t%d MB\n" $OS $(( $FREE / 1024**2 ))
```

Le marqueur `%s` est destiné au contenu textuel (il sera remplacé par la variable `$OS`) et le marqueur `%d` est destiné aux nombres entiers (il sera remplacé par le nombre de mégaoctets libres dans la RAM). `printf` n'ajoute pas de caractère de retour à la ligne à la fin du texte, le caractère correspondant `\n` doit donc être placé à la fin du motif si nécessaire. Le motif entier sera interprété comme un seul argument et doit donc être placé entre guillemets.

TIP

Le format de substitution des espaces réservés effectué par `printf` peut être personnalisé en utilisant la même structure que celle utilisée par la fonction `printf` du langage de programmation C. La référence complète de la fonction `printf` peut être trouvée dans sa page de manuel, accessible avec la commande `man 3 printf`.

Avec `printf`, les variables sont placées à l'extérieur du motif de texte, ce qui permet de stocker le motif dans une variable séparée :

```
MSG='Operating system:\t%s\nUnallocated RAM:\t%d MB\n'
printf "$MSG" $OS $(( $FREE / 1024**2 ))
```

Cette méthode est particulièrement utile pour afficher des formats de sortie distincts, en fonction des besoins de l'utilisateur. Elle facilite, par exemple, l'écriture d'un script qui utilise un modèle de texte distinct si l'utilisateur souhaite une liste CSV (*Comma Separated Values*) plutôt qu'un message de sortie par défaut.

Exercices guidés

1. L'option `-s` de la commande `read` est utile pour saisir les mots de passe, étant donné qu'elle n'affiche pas le contenu tapé à l'écran. Comment pourrait-on utiliser la commande `read` pour stocker la saisie de l'utilisateur dans la variable `PASSWORD` tout en masquant le contenu de la frappe ?

2. Le seul rôle de la commande `whoami` consiste à afficher l'identifiant de l'utilisateur qui l'a appelée. C'est pourquoi elle est principalement utilisée dans les scripts pour identifier l'utilisateur qui l'exécute. Dans un script Bash, comment pourrait-on stocker le résultat de la commande `whoami` dans une variable nommée `WHO` ?

3. Quel opérateur Bash doit se trouver entre les commandes `apt-get dist-upgrade` et `systemctl reboot` si l'utilisateur root veut exécuter `systemctl reboot` seulement si `apt-get dist-upgrade` s'est terminé avec succès ?

Exercices d'approfondissement

1. Un utilisateur essaie d'exécuter un script Bash nouvellement créé, mais il est confronté au message d'erreur suivant :

```
bash: ./script.sh: Permission denied
```

Sachant que le fichier `./script.sh` a été créé par le même utilisateur, quelle pourrait être la cause probable de cette erreur ?

2. Admettons qu'un fichier script nommé `do.sh` soit exécutable et que le lien symbolique nommé `undo.sh` pointe vers lui. Depuis le script, comment pourriez-vous déterminer si le fichier qui l'appelle est `do.sh` ou `undo.sh` ?

3. Dans un système avec un service de messagerie correctement configuré, la commande `mail -s "Maintenance Error" root <<<"Scheduled task error"` envoie un message de notification à l'utilisateur `root`. Une telle commande pourrait être utilisée dans des tâches automatiques comme les *cronjobs* pour informer l'administrateur du système d'un problème inattendu. Écrivez une fonction *if* qui exécutera la commande `mail` mentionnée ci-dessus si le statut de sortie de la commande précédente — quelle qu'elle soit — est un échec.

Résumé

Cette leçon aborde les concepts de base qui permettent de comprendre et d'écrire des scripts shell Bash. Les scripts shell sont un élément essentiel de toute distribution Linux, étant donné qu'ils offrent un moyen très souple d'automatiser les tâches de l'utilisateur et du système effectuées dans l'environnement shell. La leçon passe par les étapes suivantes :

- Structure des scripts shell et permissions correctes pour les fichiers de scripts
- Paramètres des scripts
- Utiliser des variables pour lire les informations saisies par l'utilisateur et pour stocker les résultats des commandes
- Les tableaux unidimensionnels de Bash
- Tests simples et exécution conditionnelle
- Mise en forme de l'affichage des résultats

Voici les commandes et les procédures abordées :

- Notation intégrée à Bash pour la substitution de commandes, l'expansion de tableaux et les expressions arithmétiques
- Exécution conditionnelle des commandes avec les opérateurs `||` et `&&`
- `echo`
- `chmod`
- `exec`
- `read`
- `declare`
- `test`
- `if`
- `printf`

Réponses aux exercices guidés

1. L'option `-s` de la commande `read` est utile pour saisir les mots de passe, étant donné qu'elle n'affiche pas le contenu tapé à l'écran. Comment pourrait-on utiliser la commande `read` pour stocker la saisie de l'utilisateur dans la variable `PASSWORD` tout en masquant le contenu de la frappe ?

```
read -s PASSWORD
```

2. Le seul rôle de la commande `whoami` consiste à afficher l'identifiant de l'utilisateur qui l'a appelée. C'est pourquoi elle est principalement utilisée dans les scripts pour identifier l'utilisateur qui l'exécute. Dans un script Bash, comment pourrait-on stocker le résultat de la commande `whoami` dans une variable nommée `WHO` ?

```
WHO=`whoami` ou WHO=$(whoami)
```

3. Quel opérateur Bash doit se trouver entre les commandes `apt-get dist-upgrade` et `systemctl reboot` si l'utilisateur root veut exécuter `systemctl reboot` seulement si `apt-get dist-upgrade` s'est terminé avec succès ?

L'opérateur `&&`, comme dans `apt-get dist-upgrade && systemctl reboot`.

Réponses aux exercices d'approfondissement

1. Un utilisateur essaie d'exécuter un script Bash nouvellement créé, mais il est confronté au message d'erreur suivant :

```
bash: ./script.sh: Permission denied
```

Sachant que le fichier `./script.sh` a été créé par le même utilisateur, quelle pourrait être la cause probable de cette erreur ?

Le script `./script.sh` ne possède pas les droits d'exécution.

2. Admettons qu'un fichier script nommé `do.sh` soit exécutable et que le lien symbolique nommé `undo.sh` pointe vers lui. Depuis le script, comment pourriez-vous déterminer si le fichier qui l'appelle est `do.sh` ou `undo.sh` ?

La variable spéciale `$0` contient le nom de fichier utilisé pour appeler le script.

3. Dans un système avec un service de messagerie correctement configuré, la commande `mail -s "Maintenance Error" root <<<"Scheduled task error"` envoie un message de notification à l'utilisateur `root`. Une telle commande pourrait être utilisée dans des tâches automatiques comme les *cronjobs* pour informer l'administrateur du système d'un problème inattendu. Écrivez une fonction *if* qui exécutera la commande `mail` mentionnée ci-dessus si le statut de sortie de la commande précédente — quelle qu'elle soit — est un échec.

```
if [ "$?" -ne 0 ]; then mail -s "Maintenance Error" root <<<"Scheduled task error"; fi
```



105.2 Leçon 2

Certification :	LPIC-1
Version :	5.0
Thème :	105 Shells et scripts shell
Objectif :	105.2 Personnaliser ou écrire des scripts simples
Leçon :	2 sur 2

Introduction

En règle générale, les scripts shell servent à automatiser les opérations liées aux fichiers et aux répertoires, les mêmes opérations que celles qui peuvent s'effectuer manuellement en ligne de commande. Cependant, la portée des scripts shell ne se limite pas aux documents d'un utilisateur, étant donné que la configuration et l'interaction avec certains éléments d'un système d'exploitation Linux s'effectuent également à l'aide d'une série de scripts.

Le shell Bash comporte toute une série de commandes intégrées (*builtins* ou primitives du shell) fort pratiques pour écrire des scripts, mais toute la puissance des scripts repose sur la combinaison des commandes intégrées de Bash avec les nombreux outils en ligne de commande disponibles sur un système Linux.

Tests étendus

En tant que langage de script, Bash sert principalement à manipuler des fichiers. C'est pourquoi la commande interne `test` de Bash dispose de toute une série d'options qui permettent d'évaluer les propriétés de tout ce qui constitue le système de fichiers (notamment les fichiers et les

répertoires). Ces tests permettent, par exemple, de vérifier si les fichiers et les répertoires nécessaires à l'exécution d'une tâche particulière sont présents et peuvent être lus. Ensuite, en combinaison avec une structure conditionnelle `if`, les actions appropriées sont exécutées en fonction du résultat du test.

La commande `test` peut évaluer des expressions en utilisant deux syntaxes différentes : les expressions de test peuvent être fournies en argument à la commande `test` ou elles peuvent être placées entre crochets, avec la commande `test` donnée implicitement. Ainsi, le test pour évaluer si `/etc` est un répertoire valide peut s'écrire `test -d /etc` ou `[ -d /etc ]` :

```
$ test -d /etc
$ echo $?
0
$ [ -d /etc ]
$ echo $?
0
```

Comme le confirme le code de sortie de la variable spéciale `$?` — une valeur de 0 signifie que le test a réussi — les deux formes ont évalué `/etc` en tant que répertoire valide. En admettant que le chemin vers un fichier ou un répertoire est enregistré dans la variable `$VAR`, les expressions suivantes peuvent être utilisées comme arguments de `test` ou entre crochets :

-a "\$VAR"

Vérifie si `VAR` existe dans le système de fichiers et est un fichier.

-b "\$VAR"

Vérifie si `VAR` existe et est un fichier spécial en mode bloc.

-c "\$VAR"

Vérifie si `VAR` existe et est un fichier spécial en mode caractère.

-d "\$VAR"

Vérifie si `VAR` existe et est un répertoire

-e "\$VAR"

Vérifie si `VAR` existe dans le système de fichiers.

-f "\$VAR"

Vérifie si `VAR` existe et est un fichier ordinaire.

-g "\$VAR"

Vérifie si VAR existe et a son bit SGID positionné.

-h "\$VAR"

Vérifie si VAR existe et est un lien symbolique.

-L "\$VAR"

Vérifie si VAR existe et est un lien symbolique (comme -h).

-k "\$VAR"

Vérifie si VAR existe et son bit collant (*sticky*) est positionné.

-p "\$VAR"

Vérifie si VAR existe et est un tube nommé.

-r "\$VAR"

Vérifie si VAR existe et est lisible pour l'utilisateur en cours.

-s "\$VAR"

Vérifie si VAR existe et n'est pas vide.

-S "\$VAR"

Vérifie si VAR existe et est un fichier socket.

-t "\$VAR"

Vérifie si VAR est ouvert dans un terminal.

-u "\$VAR"

Vérifie si VAR existe et son bit SUID positionné.

-w "\$VAR"

Vérifie si VAR existe et est accessible en écriture pour l'utilisateur en cours.

-x "\$VAR"

Vérifie si VAR existe et est exécutable pour l'utilisateur en cours.

-O "\$VAR"

Vérifie si VAR existe et appartient à l'utilisateur en cours.

-G "\$VAR"

Vérifie si VAR existe et appartient au groupe effectif de l'utilisateur en cours.

-N "\$VAR"

Vérifie si VAR existe et a été modifié depuis le dernier accès.

"\$VAR1" -nt "\$VAR2"

Vérifie si VAR1 est plus récent (*newer than*) que VAR2 en fonction de la date de dernière modification.

"\$VAR1" -ot "\$VAR2"

Vérifie si VAR1 est plus ancien (*older than*) que VAR2.

"\$VAR1" -ef "\$VAR2"

Vérifie si VAR1 est un lien symbolique vers \$VAR2.

Il est préférable d'utiliser les guillemets doubles autour d'une variable testée car, si la variable est vide, elle peut provoquer une erreur de syntaxe pour la commande `test`. Les options de test requièrent un argument d'opérande et une variable vide sans guillemets provoquerait une erreur due à l'absence de cet argument. Il existe également des tests pour des variables texte arbitraires, tels que ceux décrits ci-dessous :

-z "\$TXT"

Vérifie si la variable TXT est vide (taille zéro).

-n "\$TXT" ou test "\$TXT"

Vérifie si la variable TXT n'est pas vide.

"\$TXT1" = "\$TXT2" ou "\$TXT1" == "\$TXT2"

Vérifie si TXT1 est égal à TXT2.

"\$TXT1" != "\$TXT2"

Vérifie si TXT1 n'est pas égal à TXT2.

"\$TXT1" < "\$TXT2"

Vérifie si TXT1 vient avant TXT2 par ordre alphabétique.

"\$TXT1" > "\$TXT2"

Vérifie si TXT1 vient après TXT2 par ordre alphabétique.

Chaque langage peut avoir des règles différentes en ce qui concerne l'ordre alphabétique. Pour obtenir des résultats cohérents, quels que soient les paramètres de localisation du système sur lequel le script est exécuté, il est préférable de définir la variable d'environnement LANG à C, comme dans `LANG=C`, avant d'effectuer des opérations qui impliquent l'ordre alphabétique. Cette

définition va également afficher les messages du système dans la langue d'origine, elle ne doit donc être utilisée que dans le contexte du script.

Les comparaisons numériques ont leur propre jeu de tests :

\$NUM1 -lt \$NUM2

Vérifie si NUM1 est plus petit que (*less than*) NUM2.

\$NUM1 -gt \$NUM2

Vérifie si NUM1 est plus grand que (*greater than*) NUM2.

\$NUM1 -le \$NUM2

Vérifie si NUM1 est plus petit ou égal à (*less or equal*) NUM2.

\$NUM1 -ge \$NUM2

Vérifie si NUM1 est plus grand ou égal à (*greater or equal*) NUM2.

\$NUM1 -eq \$NUM2

Vérifie si NUM1 est plus égal à (*equal*) NUM2.

\$NUM1 -ne \$NUM2

Vérifie si NUM1 n'est pas égal à (*not equal*) NUM2.

Tous les tests peuvent recevoir les modificateurs suivants :

! EXPR

Vérifie si l'expression EXPR est fausse.

EXPR1 -a EXPR2

Vérifie si les deux expressions EXPR1 et EXPR2 sont vraies.

EXPR1 -o EXPR2

Vérifie si au moins l'une des deux expressions est vraie.

Une autre structure conditionnelle, `case`, peut être considérée comme une variante de la structure `if`. L'instruction `case` va exécuter une liste de commandes données si un élément spécifié — le contenu d'une variable, par exemple — se trouve dans une liste d'éléments séparés par des *pipes* (la barre verticale `|`) et finissant par `)`. L'exemple de script ci-dessous montre comment la structure `case` peut être utilisée pour indiquer le format de paquet logiciel correspondant à une distribution Linux donnée :

```
#!/bin/bash

DISTRO=$1

echo -n "Distribution $DISTRO uses "
case "$DISTRO" in
    debian | ubuntu | mint)
        echo -n "the DEB"
        ;;
    centos | fedora | opensuse )
        echo -n "the RPM"
        ;;
    *)
        echo -n "an unknown"
        ;;
esac
echo " package format."
```

Chaque liste de motifs et de commandes associées doit être terminée par `;;`, `&`, ou `&`. Le dernier motif, un astérisque, correspondra si aucun autre motif n'a été trouvé auparavant. L'instruction `esac` (*case* à l'envers) met fin à la structure `case`. En admettant que l'exemple de script précédent ait été nommé `script.sh` et qu'il soit exécuté avec `opensuse` comme premier argument, le résultat suivant sera généré :

```
$ ./script.sh opensuse
Distribution opensuse uses the RPM package format.
```

TIP

Bash a une option nommée `nocasematch` qui active le filtrage insensible à la casse pour la construction `case` et d'autres commandes conditionnelles. La commande intégrée `shopt` bascule les valeurs des paramètres qui gèrent le comportement des options du shell : `shopt -s` va activer (*set*) l'option en question et `shopt -u` va désactiver (*unset*) l'option donnée. Ainsi, le fait de placer `shopt -s nocasematch` avant la structure `case` va activer le filtrage insensible à la casse. Les options modifiées par `shopt` n'affecteront que la session courante, donc les options modifiées dans les scripts s'exécutant dans un sous-shell — ce qui est la manière standard d'exécuter un script — n'affectent pas les options de la session parent.

L'élément recherché et les motifs sont soumis à l'expansion du tilde, à l'expansion des paramètres, à la substitution de commandes et à l'expansion arithmétique. Si l'élément recherché est spécifié avec des guillemets, ces derniers seront supprimés avant que la correspondance ne soit établie.

Les structures de boucles

Les scripts sont souvent utilisés pour automatiser des tâches répétitives, en exécutant le même jeu de commandes jusqu'à ce qu'un critère d'arrêt soit satisfait. Bash possède trois instructions de boucles — `for`, `until` et `while` — conçues pour des structures de boucles légèrement distinctes.

La structure `for` parcourt une liste donnée d'éléments—généralement une liste de mots ou d'autres segments de texte séparés par des espaces—en exécutant le même ensemble de commandes sur chacun de ces éléments. Avant chaque itération, l'instruction `for` assigne l'élément courant à une variable, qui peut alors être utilisée par les commandes correspondantes. Le processus est répété jusqu'à ce qu'il n'y ait plus d'éléments. Voici la syntaxe de la structure `for` :

```
for VARNAME in LIST
do
    COMMANDS
done
```

`VARNAME` est un nom de variable shell aléatoire et `LIST` représente n'importe quelle séquence de termes séparés. Les caractères de délimitation valides pour séparer les éléments de la liste sont définis par la variable d'environnement `IFS` et sont par défaut les caractères *espace*, *tabulation* et *retour chariot*. La liste des commandes à exécuter est délimitée par les instructions `do` et `done` de sorte que les commandes peuvent occuper autant de lignes que nécessaire.

Dans l'exemple suivant, la commande `for` récupère chaque élément de la liste fournie—une séquence de nombres—et l'affecte à la variable `NUM`, un élément à la fois :

```
#!/bin/bash

for NUM in 1 1 2 3 5 8 13
do
    echo -n "$NUM is "
    if [ $(( $NUM % 2 )) -ne 0 ]
    then
        echo "odd."
    else
        echo "even."
    fi
done
```

Dans l'exemple, une structure `if` imbriquée est utilisée en conjonction avec une expression

arithmétique pour évaluer si le nombre dans la variable `NUM` est pair ou impair. En admettant que l'exemple de script ci-dessus ait été nommé `script.sh` et qu'il se trouve dans le répertoire courant, le résultat suivant sera généré :

```
$ ./script.sh
1 is odd.
1 is odd.
2 is even.
3 is odd.
5 is odd.
8 is even.
13 is odd.
```

Bash gère également un format alternatif pour les structures `for`, avec la notation en double parenthèse. Cette notation ressemble à la syntaxe de l'instruction `for` du langage de programmation C et se révèle particulièrement utile pour travailler avec des tableaux :

```
#!/bin/bash

SEQ=( 1 1 2 3 5 8 13 )

for (( IDX = 0; IDX < ${#SEQ[*]}; IDX++ ))
do
    echo -n "${SEQ[$IDX]} is "
    if [ $(( ${SEQ[$IDX]} % 2 )) -ne 0 ]
    then
        echo "odd."
    else
        echo "even."
    fi
done
```

Cet échantillon de script génère exactement le même résultat que l'exemple ci-dessus. En revanche, au lieu d'utiliser la variable `NUM` pour stocker un élément à la fois, c'est la variable `IDX` qui est utilisée pour suivre l'index actuel du tableau en ordre croissant, en commençant par 0 et en l'incrémentant continuellement tant qu'il est inférieur au nombre d'éléments dans le tableau `SEQ`. L'élément actuel est récupéré à partir de sa position dans le tableau avec `${SEQ[$IDX]}`.

De la même manière, la structure `until` exécute une séquence de commandes jusqu'à ce qu'une commande de test — comme la commande `test` elle-même — se termine avec l'état 0 (succès). Par exemple, la même structure de boucle de l'exemple précédent peut être implémentée avec `until`

comme ceci :

```
#!/bin/bash

SEQ=( 1 1 2 3 5 8 13 )

IDX=0

until [ $IDX -eq ${#SEQ[*]} ]
do
    echo -n "${SEQ[$IDX]} is "
    if [ $(( ${SEQ[$IDX]} % 2 )) -ne 0 ]
    then
        echo "odd."
    else
        echo "even."
    fi
    IDX=$(( $IDX + 1 ))
done
```

Les structures `until` requièrent certes plus d'instructions que les structures `for`, mais elles sont plus adaptées aux critères d'arrêt non numériques fournis par les expressions `test` ou toute autre commande. Il convient d'inclure des actions qui garantissent un critère d'arrêt valide, comme l'incrémentement d'une variable compteur, faute de quoi la boucle risque de s'exécuter indéfiniment.

L'instruction `while` ressemble à l'instruction `until`, sauf que `while` continue à réitérer l'ensemble des commandes si la commande de test se termine avec l'état 0 (succès). Par conséquent, l'instruction `until [ $IDX -eq ${#SEQ[*]} ]` de l'exemple précédent équivaut à `while [ $IDX -lt ${#SEQ[*]} ]`, étant donné que la boucle est censée se répéter tant que l'index du tableau est *inférieur* au nombre total d'éléments dans le tableau.

Un exemple plus élaboré

Imaginons qu'un utilisateur souhaite synchroniser régulièrement ses fichiers et répertoires avec un autre périphérique de stockage, monté sur un point de montage quelconque du système de fichiers, et qu'un système de sauvegarde classique soit considéré comme excessif. Comme il s'agit d'une activité destinée à être exécutée périodiquement, c'est un bon cas de figure pour automatiser une application à l'aide d'un script shell.

La mission est simple : synchroniser tous les fichiers et répertoires contenus dans une liste, depuis

un répertoire d'origine renseigné comme premier argument du script vers un répertoire de destination renseigné comme second argument du script. Pour faciliter l'ajout ou la suppression d'éléments de la liste, celle-ci sera conservée dans un fichier séparé, `~/sync.list`, avec un élément par ligne :

```
$ cat ~/.sync.list
Documents
To do
Work
Family Album
.config
.ssh
.bash_profile
.vimrc
```

Le fichier contient un assortiment de fichiers et de répertoires, dont certains avec des espaces dans le nom. C'est un scénario approprié pour la commande interne `mapfile` de Bash, qui va analyser n'importe quel contenu textuel donné et créer une variable de type tableau à partir de celui-ci, en plaçant chaque ligne comme un élément de tableau distinct. Le fichier sera nommé `sync.sh`, et contiendra le script suivant :

```
#!/bin/bash

set -ef

# List of items to sync
FILE=~/.sync.list

# Origin directory
FROM=$1

# Destination directory
TO=$2

# Check if both directories are valid
if [ ! -d "$FROM" -o ! -d "$TO" ]
then
    echo Usage:
    echo "$0 <SOURCEDIR> <DESTDIR>"
    exit 1
fi
```



```
# Create array from file
mapfile -t LIST < $FILE

# Sync items
for (( IDX = 0; IDX < ${#LIST[*]}; IDX++ ))
do
    echo -e "$FROM/${LIST[$IDX]} \u2192 $TO/${LIST[$IDX]}";
    rsync -qa --delete "$FROM/${LIST[$IDX]}" "$TO";
done
```

La première action du script consiste à redéfinir deux options du shell avec la commande `set` : l'option `-e` terminera l'exécution immédiatement si une commande se termine avec un code d'état non nul et l'option `-f` désactive les métacaractères. Les deux options peuvent être abrégées en `-ef`. Cette étape n'est pas obligatoire, mais elle permet de diminuer les risques de comportements inattendus.

Les instructions du script qui concernent l'application à proprement parler peuvent être organisées en trois parties :

1. Récupérer et vérifier les paramètres du script

La variable `FILE` correspond au chemin du fichier qui contient la liste des éléments à copier : `~/sync.list`. Les variables `FROM` et `TO` correspondent respectivement aux chemins d'origine et de destination. Puisque ces deux derniers paramètres sont fournis par l'utilisateur, ils sont soumis à un simple test de validation effectué par la structure `if` : si l'un des deux n'est pas un répertoire valide—évalué par le test `[ ! -d "$FROM" -o ! -d "$TO" ]`—le script affichera un message d'aide succinct et se terminera avec un état de sortie de 1.

2. Charger la liste des fichiers et des répertoires

Une fois que tous les paramètres sont définis, un tableau avec les éléments à copier est créé avec la commande `mapfile -t LIST < $FILE`. L'option `-t` de `mapfile` supprime le caractère de fin de ligne de chaque ligne avant de l'inclure dans la variable tableau nommée `LIST`. Le contenu du fichier renseigné par la variable `FILE`—`~/sync.list`—est lu via une redirection d'entrée.

3. Effectuer la copie et informer l'utilisateur

Une boucle `for` avec double parenthèse parcourt la grille d'éléments, avec la variable `IDX` qui garde la trace de l'incrément de l'index. La commande `echo` informe l'utilisateur de chaque élément copié. Le caractère unicode échappé—`\u2192`—pour le caractère *flèche droite* est présent dans le message de sortie, et l'option `-e` de la commande `echo` doit donc être

utilisée. La commande `rsync` ne copie sélectivement que les parties modifiées du fichier d'origine, son utilisation est donc recommandée pour de telles tâches. Les options `-q` et `-a`, condensées en `-qa`, neutralisent les messages `rsync` et activent le mode *archive*, qui préserve toutes les propriétés du fichier. L'option `--delete` va faire en sorte que `rsync` supprime un élément dans la destination qui n'existe plus dans l'origine, elle doit donc être utilisée avec précaution.

En admettant que tous les éléments de la liste existent dans le répertoire personnel de l'utilisateur `carol`, `/home/carol`, et que le répertoire de destination `/media/carol/backup` pointe vers un périphérique de stockage externe monté, la commande `sync.sh /home/carol /media/carol/backup` va générer le résultat suivant :

```
$ sync.sh /home/carol /media/carol/backup
/home/carol/Documents → /media/carol/backup/Documents
/home/carol/"To do" → /media/carol/backup/"To do"
/home/carol/Work → /media/carol/backup/Work
/home/carol/"Family Album" → /media/carol/backup/"Family Album"
/home/carol/.config → /media/carol/backup/.config
/home/carol/.ssh → /media/carol/backup/.ssh
/home/carol/.bash_profile → /media/carol/backup/.bash_profile
/home/carol/.vimrc → /media/carol/backup/.vimrc
```

L'exemple suppose également que le script est exécuté par `root` ou par l'utilisateur `carol`, étant donné que la plupart des fichiers seraient illisibles pour d'autres utilisateurs. Si `sync.sh` n'est pas dans un répertoire figurant dans la variable d'environnement `PATH`, il doit être invoqué avec son chemin complet.

Exercices guidés

1. Comment peut-on utiliser la commande `test` pour vérifier si le chemin d'un fichier stocké dans la variable `FROM` est plus récent que celui d'un fichier dont le chemin est stocké dans la variable `TO` ?

2. Le script suivant est censé afficher une séquence de chiffres de 0 à 9, mais au lieu de cela, il affiche 0 indéfiniment. Que doit-on faire pour obtenir le résultat escompté ?

```
#!/bin/bash

COUNTER=0

while [ $COUNTER -lt 10 ]
do
    echo $COUNTER
done
```

3. Admettons qu'un utilisateur écrive un script qui nécessite une liste ordonnée de noms d'utilisateurs. La liste classée qui en résulte est présentée comme ceci sur son ordinateur :

```
carol
Dave
emma
Frank
Grace
henry
```

Or, cette même liste est classée de la manière suivante sur l'ordinateur de son collègue :

```
Dave
Frank
Grace
carol
emma
henry
```

Comment expliquer les différences de classement entre les deux listes ?

Exercices d'approfondissement

1. Comment peut-on utiliser tous les arguments en ligne de commande du script pour initialiser un tableau Bash ?

2. Comment se fait-il que, contre toute attente, la commande `test 1 > 2` soit considérée comme vraie ?

3. Comment un utilisateur pourrait-il modifier temporairement le séparateur de champ par défaut en le remplaçant uniquement par le caractère de retour à la ligne, tout en ayant la possibilité de revenir à son contenu d'origine ?

Résumé

Cette leçon aborde en détail les tests disponibles pour la commande `test` et d'autres structures conditionnelles et de boucles, nécessaires pour écrire des scripts shell plus élaborés. Un script simple de synchronisation de fichiers est donné comme exemple d'application pratique d'un script shell. La leçon comprend les points suivants :

- Tests étendus pour les structures conditionnelles `if` et `case`.
- Structures de boucles du shell : `for`, `until` et `while`.
- Itération dans les tableaux et les paramètres.

Voici les commandes et les procédures abordées :

test

Effectue une comparaison entre les éléments fournis à la commande.

if

Une structure logique utilisée dans les scripts pour évaluer quelque chose comme étant vrai ou faux, puis pour brancher l'exécution de la commande en fonction des résultats.

case

Évaluer plusieurs valeurs par rapport à une seule variable. L'exécution de la commande de script est alors effectuée en fonction du résultat de la commande `case`.

for

Répète l'exécution d'une commande en fonction d'un critère donné.

until

Répète l'exécution d'une commande jusqu'à ce qu'une expression soit évaluée comme fausse.

while

Répète l'exécution d'une commande pendant qu'une expression donnée est évaluée comme vraie.

Réponses aux exercices guidés

1. Comment peut-on utiliser la commande `test` pour vérifier si le chemin d'un fichier stocké dans la variable `FROM` est plus récent que celui d'un fichier dont le chemin est stocké dans la variable `TO` ?

La commande `test "$FROM" -nt "$TO"` renverra un code d'état de 0 si le fichier dans la variable `FROM` est plus récent que le fichier dans la variable `TO`.

2. Le script suivant est censé afficher une séquence de chiffres de 0 à 9, mais au lieu de cela, il affiche 0 indéfiniment. Que doit-on faire pour obtenir le résultat escompté ?

```
#!/bin/bash

COUNTER=0

while [ $COUNTER -lt 10 ]
do
    echo $COUNTER
done
```

La variable `COUNTER` doit être incrémentée, ce qui peut se faire avec l'expression arithmétique `COUNTER=$(( $COUNTER + 1 ))` qui permettra d'atteindre le critère d'arrêt pour terminer la boucle.

3. Admettons qu'un utilisateur écrive un script qui nécessite une liste ordonnée de noms d'utilisateurs. La liste classée qui en résulte est présentée comme ceci sur son ordinateur :

```
carol
Dave
emma
Frank
Grace
henry
```

Or, cette même liste est classée de la manière suivante sur l'ordinateur de son collègue :

```
Dave
Frank
Grace
carol
```

```
emma  
henry
```

Comment expliquer les différences de classement entre les deux listes ?

Le tri se base sur les paramètres linguistiques du système actuel. Pour éviter les incohérences, les tâches de classement doivent être effectuées avec la variable d'environnement `LANG` fixée à `C`.

Réponses aux exercices d'approfondissement

1. Comment peut-on utiliser tous les arguments en ligne de commande du script pour initialiser un tableau Bash ?

Les commandes `PARAMS=( $* )` ou `PARAMS=( "$@" )` créent un tableau nommé `PARAMS` avec tous les arguments.

2. Comment se fait-il que, contre toute attente, la commande `test 1 > 2` soit considérée comme vraie ?

L'opérateur `>` est conçu pour les tests de chaînes de caractères, pas pour les tests numériques.

3. Comment un utilisateur pourrait-il modifier temporairement le séparateur de champ par défaut en le remplaçant uniquement par le caractère de retour à la ligne, tout en ayant la possibilité de revenir à son contenu d'origine ?

Une copie de la variable `IFS` peut être stockée dans une autre variable : `OLDIFS=$IFS`. Le nouveau séparateur de ligne est alors défini avec `IFS=$'\n'` et la variable `IFS` peut être rétablie avec `IFS=$OLDIFS`.



Thème 106: Interfaces et bureaux utilisateur



106.1 Installation et configuration de X11

Référence aux objectifs de LPI

LPIC-1 version 5.0, Exam 102, Objective 106.1

Valeur

2

Domaines de connaissance les plus importants

- Compréhension de l'architecture X11.
- Compréhension et connaissances de base du fichier de configuration de X Window.
- Remplacement d'aspects spécifiques de la configuration, comme l'agencement du clavier.
- Compréhension des composants des environnements de bureau, comme les gestionnaires d'affichage et les gestionnaires de fenêtres.
- Gestion de l'accès au serveur X et affichage d'applications sur des serveurs X distants.
- Connaissance de base de Wayland.

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

- `/etc/X11/xorg.conf`
- `/etc/X11/xorg.conf.d/`
- `~/.xsession-errors`
- `xhost`
- `xauth`
- `DISPLAY`
- `X`



106.1 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	106 Interfaces utilisateur et environnements de bureau
Objectif :	106.1 Installer et configurer X11
Leçon :	1 sur 1

Introduction

Le système X Window est une pile logicielle utilisée pour afficher du texte et des graphismes à l'écran. L'aspect général et la conception d'un client X ne sont pas déterminés par le système X Window. C'est chaque client X qui les gère : soit un *gestionnaire de fenêtres* (comme Window Maker ou Tab Window Manager par exemple), soit un *environnement de bureau* complet comme KDE, GNOME ou Xfce. Les environnements de bureau seront abordés dans une leçon ultérieure. Cette leçon se concentre sur l'architecture sous-jacente et les outils classiques du système X Window qu'un administrateur peut utiliser pour configurer X.

Le système X Window est multiplateforme et fonctionne sur différents systèmes d'exploitation comme Linux, les BSD, Solaris et d'autres systèmes de type Unix. Il existe également des implémentations pour macOS d'Apple et Microsoft Windows.

La principale version du protocole X utilisée dans les distributions Linux modernes est la version 11 de X.org, communément appelée *X11*. Le protocole X est le mécanisme de communication entre le client X et le serveur X. Les différences entre le client X et le serveur X seront abordées ci-dessous.

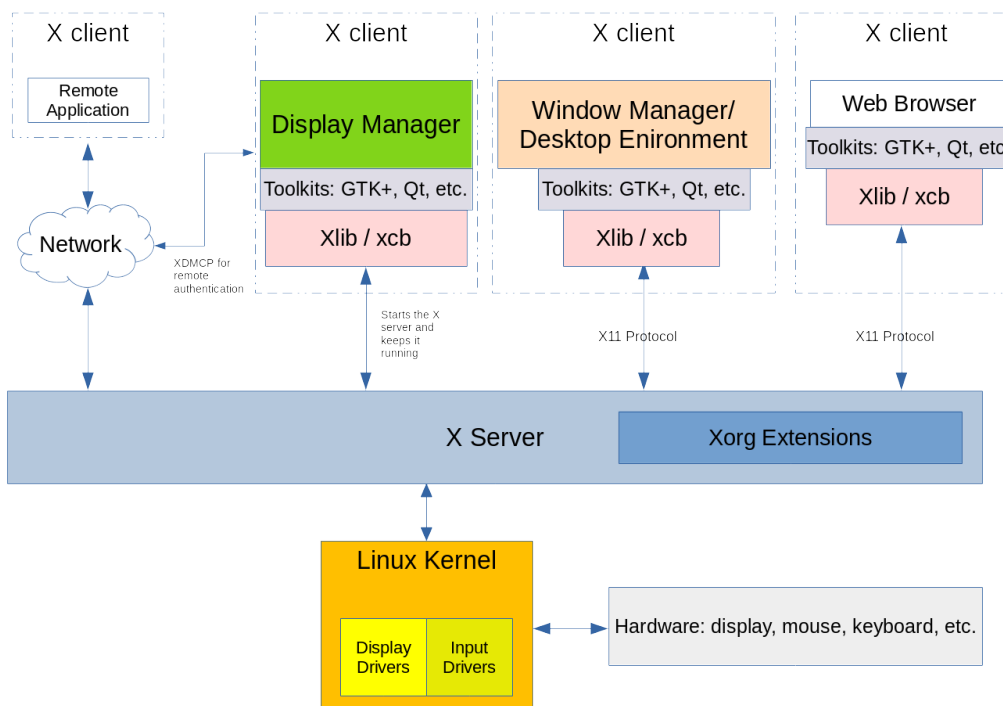
NOTE

Le prédécesseur du système X Window était un système de fenêtres appelé *W*, développé conjointement par IBM, DEC et le MIT. Ce logiciel est issu du *Projet Athena* en 1984. Lorsque les développeurs ont commencé à travailler sur un nouveau serveur d'affichage, ils ont choisi la prochaine lettre de l'alphabet : "X". L'évolution du système X Window est actuellement contrôlée par le *MIT X Consortium*.

Architecture du système X Window

Le système X Window fournit les mécanismes qui permettent de dessiner des objets 2D de base (et des objets 3D grâce à des extensions) sur un écran. Il est constitué d'un client et d'un serveur, et dans la plupart des installations où un environnement graphique est requis, ces deux composants se trouvent sur le même ordinateur. La partie client prend la forme d'une application comme un émulateur de terminal, un jeu ou un navigateur web. Chaque application client informe le serveur X de l'emplacement et de la taille de sa fenêtre sur l'écran de l'ordinateur. Le client gère également le contenu de cette fenêtre, et le serveur X affiche le dessin souhaité à l'écran. Le système X Window gère également les entrées en provenance des périphériques comme les souris, les claviers, les trackpads, etc.

Structure de base d'un système X Window



Le système X Window est capable de fonctionner en réseau et plusieurs clients X sur différents ordinateurs d'un réseau peuvent adresser des requêtes graphiques à un seul serveur X distant. Le raisonnement à la base de ce système est qu'un administrateur ou un utilisateur peut avoir accès à une application graphique sur un système distant qui n'est pas forcément disponible sur son système local.

L'une des principales caractéristiques du système X Window est sa modularité. Au cours de l'existence du système X Window, de nouvelles fonctionnalités ont été développées et ajoutées à son infrastructure. Ces nouveaux composants ont été ajoutés sous forme d'extensions au serveur X, en laissant le protocole X11 intact. Ces extensions sont contenues dans les fichiers de bibliothèques *Xorg*. Voici quelques exemples de bibliothèques Xorg : `libXrandr`, `libXcursor`, `libX11`, `libxkbfile` et beaucoup d'autres, chacune fournissant des fonctionnalités étendues au serveur X.

Un *gestionnaire d'affichage* fournit une connexion graphique à un système. Ce système peut être un ordinateur local ou un ordinateur sur le réseau. Le gestionnaire d'affichage est lancé après le démarrage de l'ordinateur et lance une session de serveur X pour l'utilisateur authentifié. Le gestionnaire d'affichage est également chargé de maintenir le serveur X en état de marche. Parmi les exemples de gestionnaires d'affichage, on peut citer GDM, SDDM et LightDM.

Chaque instance d'un serveur X en cours d'exécution possède un *nom d'affichage* (*display name*) pour l'identifier. Le nom d'affichage contient les éléments suivants :

```
hostname:displaynumber.screennumber
```

Le nom d'affichage indique également à une application graphique où elle doit être affichée et sur quel hôte (en cas de connexion X à distance).

La partie `hostname` fait référence au nom du système censé afficher l'application. Si le nom d'hôte n'est pas indiqué dans le nom d'affichage, c'est l'hôte local qui est présumé.

La partie `displaynumber` fait référence à l'ensemble des *écrans* (*screens*) utilisés, qu'il s'agisse d'un seul écran d'ordinateur portable ou de plusieurs écrans sur une station de travail. Chaque session de serveur X en cours d'exécution se voit attribuer un numéro d'affichage en commençant par 0.

Le `screennumber` par défaut est 0. C'est le cas lorsqu'un seul écran physique ou plusieurs écrans physiques sont configurés de manière à fonctionner comme un seul écran. Lorsque tous les écrans d'une configuration multi-moniteurs sont regroupés en un seul écran logique, les fenêtres des applications peuvent être déplacées librement d'un écran à l'autre. En revanche, si les écrans sont configurés de manière à fonctionner indépendamment l'un de l'autre, chaque écran va

contenir les fenêtres des applications qui s’y ouvrent, et ces fenêtres ne pourront pas être déplacées d’un écran à l’autre. Un numéro est attribué à chaque écran indépendant. Si un seul écran logique est utilisé, le point et le numéro d’écran sont omis.

Le nom d’affichage d’une session X en cours est enregistré dans la variable d’environnement `DISPLAY` :

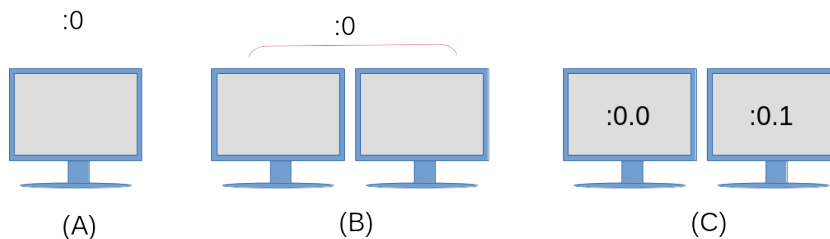
```
$ echo $DISPLAY :0
```

Le résultat fournit les éléments suivants :

1. Le serveur X utilisé réside sur le système local, ce qui explique que rien ne s’affiche à gauche des deux points.
2. La session en cours du serveur X est la première, comme l’indique le `0` qui suit immédiatement les deux points.
3. Il n’y a qu’un seul écran logique en cours d’utilisation, de sorte qu’un numéro d’écran n’est pas affiché.

Pour mieux illustrer ce concept, voici un schéma :

Exemples de configurations d’affichage



(A)

Un seul moniteur, avec une configuration d’affichage simple et un seul écran.

(B)

Configuré comme un seul affichage, avec deux moniteurs physiques configurés comme un seul écran. Les fenêtres d’application peuvent être déplacées librement entre les deux moniteurs.

(C)

Une configuration d’affichage simple (comme le montre le `:0`), mais chaque moniteur est un écran indépendant. Les deux écrans partagent les mêmes périphériques d’entrée comme le clavier et la souris, mais une application lancée sur l’écran `:0.0` ne pourra pas être déplacée

sur l'écran `:0.1` et vice-versa.

Pour démarrer une application sur un écran spécifique, affectez le numéro d'écran (screennumber) à la variable d'environnement `DISPLAY` avant de lancer l'application :

```
$ DISPLAY=:0.1 firefox &
```

Cette commande lance le navigateur web Firefox sur l'écran de droite dans le schéma ci-dessus. Certaines boîtes à outils fournissent également des options en ligne de commande pour demander à une application de s'exécuter sur un écran spécifique. Voir `--screen` et `--display` dans la page de manuel `gtk-options(7)` pour avoir un exemple.

Configuration du serveur X

Traditionnellement, le fichier de configuration principal utilisé pour configurer un serveur X est le fichier `/etc/X11/xorg.conf`. Sur les distributions Linux modernes, le serveur X se configure lui-même au moment de l'exécution lorsqu'il est démarré, et il se peut donc que le fichier `xorg.conf` n'existe pas.

Le fichier `xorg.conf` est structuré en stances distinctes appelées *sections*. Chaque section commence par le terme `Section` suivi du *nom de la section* qui fait référence à la configuration d'un composant. Chaque `Section` est terminée par un `EndSection` correspondant. Le fichier `xorg.conf` typique contient les sections suivantes :

InputDevice

Permet de configurer un modèle spécifique de clavier ou de souris.

InputClass

Dans les distributions Linux modernes, cette section se trouve typiquement dans un fichier de configuration séparé rangé dans `/etc/X11/xorg.conf.d/`. L'option `InputClass` est utilisée pour configurer une *classe* de périphériques matériels comme les claviers et les souris plutôt qu'un matériel spécifique. Voici un exemple de fichier `/etc/X11/xorg.conf.d/00-keyboard.conf` :

```
Section "InputClass"
    Identifier "system-keyboard"
    MatchIsKeyboard "on"
    Option "XkbLayout" "us"
    Option "XkbModel" "pc105"
EndSection
```


L'option `XkbLayout` définit la disposition des touches sur un clavier, comme Dvorak, gaucher ou droitier, QWERTY et la langue. L'option `XkbModel` permet de définir le type de clavier utilisé. Un tableau avec les modèles, les dispositions et leurs descriptions peut être consulté dans `xkeyboard-config(7)`. Les fichiers associés aux dispositions de clavier se trouvent dans `/usr/share/X11/xkb`. Voici un exemple de disposition de clavier grec polytonique sur un ordinateur Chromebook :

```
Section "InputClass"
    Identifier "system-keyboard"
    MatchIsKeyboard "on"
    Option "XkbLayout" "gr(polytonic)"
    Option "XkbModel" "chromebook"
EndSection
```

Alternativement, la disposition du clavier peut être modifiée pendant une session X en cours d'exécution avec la commande `setxkbmap`. Voici un exemple de cette commande qui configure la disposition grecque polytonique sur un ordinateur Chromebook :

```
$ setxkbmap -model chromebook -layout "gr(polytonic)"
```

Cette configuration ne sera valable que pour la durée de la session X en cours d'utilisation. Pour rendre ces modifications persistantes, modifiez le fichier `/etc/X11/xorg.conf.d/00-keyboard.conf` pour y inclure les paramètres requis.

NOTE

La commande `setxkbmap` utilise la *X Keyboard Extension* (XKB). Voilà un exemple de la fonctionnalité supplémentaire du système X Window grâce à l'utilisation d'extensions.

Les distributions Linux modernes fournissent la commande `localectl` via `systemd` qui permet également de modifier la disposition du clavier en créant automatiquement le fichier de configuration `/etc/X11/xorg.conf.d/00-keyboard.conf`. Encore une fois, voici un exemple de configuration d'un clavier grec polytonique sur un Chromebook, cette fois avec la commande `localectl` :

```
$ localectl --no-convert set-x11-keymap "gr(polytonic)" chromebook
```

L'option `--no-convert` est utilisée ici pour empêcher `localectl` de modifier la disposition du clavier de la console de l'hôte.

Monitor

La section `Monitor` décrit le moniteur physique utilisé et l'endroit où il est connecté. Voici un exemple de configuration avec un écran matériel connecté au deuxième port d'affichage et utilisé comme écran principal.

```
Section "Monitor"
    Identifier "DP2"
    Option     "Primary" "true"
EndSection
```

Device

La section `Device` décrit la carte vidéo matérielle utilisée. Cette section indique également le module du noyau utilisé comme pilote pour la carte vidéo, ainsi que son emplacement physique sur la carte mère.

```
Section "Device"
    Identifier "Device0"
    Driver     "i915"
    BusID      "PCI:0:2:0"
EndSection
```

Screen

La section `Screen` associe les sections `Monitor` et `Device`. Une section `Screen` pourrait ressembler à ceci :

```
Section "Screen"
    Identifier "Screen0"
    Device     "Device0"
    Monitor    "DP2"
EndSection
```

ServerLayout

La section `ServerLayout` regroupe toutes les sections comme la souris, le clavier et les écrans en une seule interface du système X Window.

```
Section "ServerLayout"
    Identifier "Layout-1"
    Screen     "Screen0" 0 0
    InputDevice "mouse1" "CorePointer"
```

```
InputDevice "system-keyboard" "CoreKeyboard"
EndSection
```

NOTE

Toutes les sections ne sont pas forcément présentes dans un fichier de configuration. Dans les cas de figure où une section est manquante, l'instance de serveur X en cours d'exécution fournit des valeurs par défaut.

Les fichiers de configuration définis par l'utilisateur résident également dans `/etc/X11/xorg.conf.d/`. Les fichiers de configuration fournis par la distribution sont disponibles dans `/usr/share/X11/xorg.conf.d/`. Les fichiers de configuration situés dans `/etc/X11/xorg.conf.d/` sont pris en compte avant le fichier `/etc/X11/xorg.conf` s'il existe sur le système.

La commande `xdpinfo` est utilisée sur un ordinateur pour afficher des informations sur une instance de serveur X en cours d'exécution. Voici un exemple de retour d'information de la commande :

```
$ xdpinfo
name of display:      :0
version number:      11.0
vendor string:       The X.Org Foundation
vendor release number: 12004000
X.Org version: 1.20.4
maximum request size: 16777212 bytes
motion buffer size:  256
bitmap unit, bit order, padding:  32, LSBFirst, 32
image byte order:     LSBFirst
number of supported pixmap formats:  7
supported pixmap formats:
    depth 1, bits_per_pixel 1, scanline_pad 32
    depth 4, bits_per_pixel 8, scanline_pad 32
    depth 8, bits_per_pixel 8, scanline_pad 32
    depth 15, bits_per_pixel 16, scanline_pad 32
    depth 16, bits_per_pixel 16, scanline_pad 32
    depth 24, bits_per_pixel 32, scanline_pad 32
    depth 32, bits_per_pixel 32, scanline_pad 32
keycode range:       minimum 8, maximum 255
focus: None
number of extensions:  25
    BIG-REQUESTS
    Composite
    DAMAGE
    DOUBLE-BUFFER
```

```

DRI3
GLX
Generic Event Extension
MIT-SCREEN-SAVER
MIT-SHM
Present
RANDR
RECORD
RENDER
SECURITY
SHAPE
SYNC
X-Resource
XC-MISC
XFIXES
XFree86-VidModeExtension
XINERAMA
XInputExtension
XKEYBOARD
XTEST
XVideo
default screen number: 0
number of screens: 1

screen #0:
dimensions: 3840x1080 pixels (1016x286 millimeters)
resolution: 96x96 dots per inch
depths (7): 24, 1, 4, 8, 15, 16, 32
root window id: 0x39e
depth of root window: 24 planes
number of colormaps: minimum 1, maximum 1
default colormap: 0x25
default number of colormap cells: 256
preallocated pixels: black 0, white 16777215
options: backing-store WHEN MAPPED, save-unders NO
largest cursor: 3840x1080
current input event mask: 0xda0033
    KeyPressMask          KeyReleaseMask          EnterWindowMask
    LeaveWindowMask       StructureNotifyMask    SubstructureNotifyMask
    SubstructureRedirectMask PropertyChangeMask    ColormapChangeMask
number of visuals: 270
...

```

Les parties les plus pertinentes du résultat sont indiquées en gras, comme le nom de l’affichage

(qui est le même que le contenu de la variable d'environnement `DISPLAY`), les informations sur la version du serveur X utilisé, le nombre et la liste des extensions Xorg utilisées, et d'autres informations sur l'écran lui-même.

Créer un fichier de configuration Xorg de base

Même si X crée sa configuration après le démarrage du système sur les installations Linux modernes, un fichier `xorg.conf` pourra toujours être utilisé. Pour générer un fichier `/etc/X11/xorg.conf` permanent, exécutez la commande suivante :

```
$ sudo Xorg -configure
```

NOTE

Au cas où une session X est déjà en cours d'exécution, vous devez spécifier un `DISPLAY` différent pour la commande, par exemple :

```
$ sudo Xorg :1 -configure
```

Sur certaines distributions Linux, la commande `X` peut être utilisée à la place de `Xorg`, étant donné que `X` est un lien symbolique vers `Xorg`.

Un fichier `xorg.conf.new` sera créé dans votre répertoire de travail actuel. Le contenu de ce fichier est dérivé de ce que le serveur X a pu trouver comme matériel et pilotes disponibles sur le système local. Pour utiliser ce fichier, il devra être placé dans le répertoire `/etc/X11/` et renommé en `xorg.conf` :

```
$ sudo mv xorg.conf.new /etc/X11/xorg.conf
```

NOTE

Les pages de manuel suivantes fournissent des informations supplémentaires sur les composants du système X Window : `xorg.conf(5)`, `Xserver(1)`, `X(1)` et `Xorg(1)`.

Wayland

Wayland est le nouveau protocole d'affichage conçu pour remplacer le système X Window. Bon nombre de distributions Linux modernes l'utilisent comme serveur d'affichage par défaut. Il est censé être plus léger en termes de ressources système avec une empreinte d'installation plus réduite que X. Le projet a débuté en 2010 et connaît un développement actif, notamment grâce au travail de développeurs X.org anciens et en activité.

Contrairement au système X Window, il n'y a pas d'instance de serveur qui s'exécute entre le client et le noyau. Au lieu de cela, une fenêtre client travaille avec son propre code ou celui d'une boîte à outils (comme Gtk+ ou Qt) pour assurer le rendu. Pour effectuer le rendu, une demande est adressée au noyau Linux via le protocole Wayland. Le noyau transmet la demande via le protocole Wayland au *compositeur* Wayland, qui s'occupe de l'entrée des périphériques, de la gestion des fenêtres et de la composition. Le compositeur est la partie du système qui combine les éléments rendus en une sortie visuelle sur l'écran.

La plupart des boîtes à outils modernes comme Gtk+ 3 et Qt 5 ont été mises à jour pour permettre le rendu sur un système X Window ainsi que sur un ordinateur sous Wayland. Toutes les applications autonomes n'ont pas encore été adaptées au rendu sous Wayland. Pour les applications et les frameworks qui visent toujours le système X Window, elles pourront être exécutées dans *XWayland*. Le système XWayland est un serveur X à part entière qui s'exécute au sein d'un client Wayland et rend ainsi le contenu d'une fenêtre client dans une instance autonome de serveur X.

Tout comme le système X Window utilise une variable d'environnement `DISPLAY` pour déterminer les écrans utilisés, le protocole Wayland utilise une variable d'environnement `WAYLAND_DISPLAY`. Vous trouverez ci-dessous un exemple de sortie d'un système qui utilise un affichage Wayland :

```
$ echo $WAYLAND_DISPLAY
wayland-0
```

Cette variable d'environnement n'est pas disponible sur les systèmes qui utilisent X.

Exercices guidés

1. Quelle commande utiliseriez-vous pour déterminer les extensions Xorg disponibles sur un système ?

2. Vous venez d'acquérir une nouvelle souris à dix boutons pour votre ordinateur, mais elle nécessite une configuration supplémentaire pour que tous les boutons fonctionnent correctement. Sans modifier le reste de la configuration du serveur X, quel répertoire utiliseriez-vous pour créer un nouveau fichier de configuration pour cette souris et quelle section de configuration précise serait utilisée dans ce fichier ?

3. Quel composant d'une installation Linux est responsable du fonctionnement d'un serveur X ?

4. Quelle option en ligne de commande est utilisée avec la commande X pour générer un nouveau fichier de configuration `xorg.conf` ?

Exercices d'approfondissement

1. Quel serait le contenu de la variable d'environnement `DISPLAY` sur un système nommé `lab01` avec une configuration à un seul écran ? Admettons que la variable d'environnement `DISPLAY` soit affichée dans un émulateur de terminal sur le troisième écran indépendant.

2. Quelle commande permet de créer un fichier de configuration du clavier pour le système X Window ?

3. Sur un système Linux classique, un utilisateur peut basculer vers une console virtuelle en appuyant sur les touches `Ctrl` + `Alt` + `F1-F6` du clavier. On vous demande de mettre en place un système kiosque avec une interface graphique et vous devez désactiver cette fonctionnalité pour empêcher toute manipulation illicite du système. Vous décidez de créer un fichier de configuration `/etc/X11/xorg.conf.d/10-kiosk.conf`. En utilisant une section `ServerFlags` (utilisée pour définir les options globales de Xorg sur le serveur), quelle option faut-il spécifier ? Consultez la page de manuel `xorg.conf(5)` pour trouver cette option.

Résumé

Cette leçon est consacrée au système X Window tel qu'il est utilisé sous Linux. Le système X Window est utilisé pour dessiner des images et du texte sur les écrans, tels qu'ils sont définis dans une série de fichiers de configuration. Le système X Window est souvent utilisé pour configurer les périphériques d'entrée tels que les souris et les claviers. Cette leçon aborde les points suivants :

- L'architecture du système X Window à un niveau élevé.
- Les fichiers de configuration utilisés pour configurer un système X Window et leur emplacement au sein du système de fichiers.
- L'utilisation de la variable d'environnement `DISPLAY` sur un système qui tourne sous X.
- Une introduction rapide au protocole d'affichage Wayland.

Voici les commandes et les fichiers de configuration abordés :

- Modifier la disposition du clavier dans une installation Xorg avec `setxkbmap` et `localectl`.
- La commande `Xorg` pour créer un nouveau fichier de configuration `/etc/X11/xorg.conf`.
- Le contenu des fichiers de configuration de Xorg situés dans : `/etc/X11/xorg.conf`, `/etc/X11/xorg.conf.d/` et `/usr/share/X11/xorg.conf.d/`.
- La commande `xdpyinfo` pour afficher des informations générales sur une session de serveur X en cours d'exécution.

Réponses aux exercices guidés

1. Quelle commande utiliseriez-vous pour déterminer les extensions Xorg disponibles sur un système ?

```
$ xdpinfo
```

2. Vous venez d'acquérir une nouvelle souris à dix boutons pour votre ordinateur, mais elle nécessite une configuration supplémentaire pour que tous les boutons fonctionnent correctement. Sans modifier le reste de la configuration du serveur X, quel répertoire utiliseriez-vous pour créer un nouveau fichier de configuration pour cette souris et quelle section de configuration précise serait utilisée dans ce fichier ?

Les configurations définies par l'utilisateur sont situées dans `/etc/X11/xorg.conf.d/` et la section dédiée à la configuration de la souris est `InputDevice`.

3. Quel composant d'une installation Linux est responsable du fonctionnement d'un serveur X ?

Le gestionnaire d'affichage.

4. Quelle option en ligne de commande est utilisée avec la commande X pour générer un nouveau fichier de configuration `xorg.conf` ?

```
-configure
```

Rappelez-vous que la commande `X` est un lien symbolique vers la commande `Xorg`.

Réponses aux exercices d'approfondissement

1. Quel serait le contenu de la variable d'environnement `DISPLAY` sur un système nommé `lab01` avec une configuration à un seul écran ? Admettons que la variable d'environnement `DISPLAY` soit affichée dans un émulateur de terminal sur le troisième écran indépendant.

```
$ echo $DISPLAY
lab01:0.2
```

2. Quelle commande permet de créer un fichier de configuration du clavier pour le système X Window ?

```
$ localectl
```

3. Sur un système Linux classique, un utilisateur peut basculer vers une console virtuelle en appuyant sur les touches `Ctrl` + `Alt` + `F1-F6` du clavier. On vous demande de mettre en place un système kiosque avec une interface graphique et vous devez désactiver cette fonctionnalité pour empêcher toute manipulation illicite du système. Vous décidez de créer un fichier de configuration `/etc/X11/xorg.conf.d/10-kiosk.conf`. En utilisant une section `ServerFlags` (utilisée pour définir les options globales de Xorg sur le serveur), quelle option faut-il spécifier ? Consultez la page de manuel `xorg.conf(5)` pour trouver cette option.

```
Section "ServerFlags"
    Option "DontVTSwitch" "True"
EndSection
```



106.2 Bureaux graphiques

Référence aux objectifs de LPI

LPIC-1 version 5.0, Exam 102, Objective 106.2

Valeur

1

Domaines de connaissance les plus importants

- Connaissance des bureaux graphiques les plus importants.
- Connaissance des protocoles utilisés pour accéder aux sessions graphiques distantes.

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

- KDE
- Gnome
- Xfce
- X11
- XDMCP
- VNC
- Spice
- RDP



106.2 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	106 Interfaces utilisateur et bureaux
Objectif :	106.2 Bureaux graphiques
Leçon :	1 sur 1

Introduction

Les systèmes d'exploitation Linux sont connus pour leur interface en ligne de commande sophistiquée, mais celle-ci peut s'avérer intimidante pour les utilisateurs sans compétences techniques. Afin de rendre l'utilisation de l'ordinateur plus intuitive, la combinaison d'écrans à haute résolution et de dispositifs de pointage a donné naissance à des interfaces utilisateur basées sur des éléments graphiques. Alors que l'interface en ligne de commande nécessite une connaissance préalable des noms des programmes et de leurs options de configuration, une interface graphique (GUI *Graphical User Interface*) permet de piloter les fonctionnalités d'un programme en pointant sur des éléments visuels familiers, ce qui rend la courbe d'apprentissage moins raide. De plus, l'interface graphique est plus adaptée au multimédia et à d'autres activités visuelles.

En effet, l'interface graphique est pratiquement synonyme d'interface informatique et la plupart des distributions Linux sont livrées avec une interface graphique installée par défaut. Il n'existe cependant pas un seul programme monolithique chargé d'assurer l'ensemble des fonctions des bureaux graphiques disponibles sous Linux. Au lieu de cela, chaque bureau graphique est en fait une vaste collection de programmes et de leurs dépendances, qui varient selon les choix de la distribution et les goûts personnels de l'utilisateur.

Le système X Window

Sous Linux et les autres systèmes d'exploitation de type Unix où il est utilisé, c'est le *système X Window* (également connu sous le nom de *X11* ou simplement *X*) qui fournit les ressources de bas niveau liées au rendu de l'interface graphique et à l'interaction de l'utilisateur avec celle-ci, comme par exemple :

- La gestion des entrées, comme les mouvements de la souris ou les saisies au clavier.
- La possibilité de couper, copier et coller du contenu texte entre des applications distinctes.
- L'interface de programmation qui permet à d'autres programmes de dessiner les éléments graphiques.

Même si le système X Window est chargé de gérer l'affichage graphique (le pilote vidéo lui-même fait partie de X), il n'est pas conçu pour dessiner lui-même des éléments visuels complexes. Les formes, les couleurs, les nuances et tout autre effet visuel sont générés par l'application qui tourne au-dessus de X. Cette approche laisse aux applications une grande marge de manœuvre pour créer des interfaces personnalisées, mais elle peut également entraîner des contraintes liées au développement qui dépassent le cadre de l'application, ainsi que certaines incohérences au niveau de l'apparence et du comportement par rapport à d'autres interfaces logicielles.

Du point de vue du développeur, l'avènement de l'environnement de bureau facilite la programmation de l'interface graphique liée au développement de l'application sous-jacente, tandis que du point de vue de l'utilisateur, elle offre une expérience cohérente entre les différentes applications. Les environnements de bureau rassemblent des interfaces de programmation, des bibliothèques et des programmes de référence qui coopèrent pour offrir des concepts de design traditionnels, mais en évolution constante.

L'environnement de bureau

L'interface graphique traditionnelle d'un ordinateur de bureau est constituée de diverses fenêtres—le terme "fenêtre" est utilisé ici pour désigner toute zone autonome de l'écran—associées à des processus en cours d'exécution. Étant donné que le système X Window n'offre que des fonctions interactives de base, l'expérience utilisateur globale repose sur les composants fournis par l'environnement de bureau.

Le gestionnaire de fenêtres, qui constitue sans doute le composant le plus important d'un environnement de bureau, permet de gérer l'emplacement et la décoration des fenêtres. C'est lui qui ajoute la barre de titre à la fenêtre, les boutons de contrôle—généralement associés aux actions de minimisation, d'agrandissement et de fermeture—et qui gère le passage d'une fenêtre ouverte à l'autre.

NOTE

Les concepts de base des interfaces graphiques des ordinateurs de bureau sont issus des espaces de travail des bureaux. Métaphoriquement, l'écran de l'ordinateur est le bureau, là où sont placés les objets tels que les documents et les dossiers. Une fenêtre d'application avec le contenu d'un document imite les actes physiques comme remplir un formulaire ou peindre une image. En tant que bureaux à proprement parler, les bureaux d'ordinateur comportent également des accessoires logiciels tels que des blocs-notes, des horloges, des calendriers, etc.

Tous les environnements de bureau fournissent un gestionnaire de fenêtres qui reflète l'aspect et la convivialité de sa boîte à outils de *widgets*. Les *widgets* sont des éléments visuels informatifs ou interactifs, comme les boutons ou les champs de saisie de texte, répartis à l'intérieur de la fenêtre de l'application. Les composants standard du bureau — comme le lanceur d'applications, la barre des tâches, etc. — et le gestionnaire de fenêtres lui-même s'appuient sur ces boîtes à outils de *widgets* pour assembler leurs interfaces.

Les bibliothèques logicielles, notamment *GTK+* et *Qt*, fournissent des *widgets* que les développeurs peuvent utiliser pour créer des interfaces graphiques sophistiquées pour leurs applications. Historiquement, les applications développées avec *GTK+* ne ressemblaient pas aux applications créées avec *Qt* et vice-versa, mais la meilleure prise en charge des thèmes dans les environnements de bureau actuels rend la distinction moins évidente.

Dans l'ensemble, *GTK+* et *Qt* offrent les mêmes fonctionnalités en termes de *widgets*. Les éléments interactifs simples peuvent être indiscernables, tandis que les *widgets* composites — comme par exemple la boîte de dialogue utilisée par les applications pour ouvrir ou enregistrer des fichiers — peuvent présenter un aspect très différent. Quoi qu'il en soit, les applications construites avec des boîtes à outils spécifiques peuvent fonctionner simultanément, indépendamment de la boîte à outils de *widgets* utilisée par les autres composants du bureau.

Au-delà des composants de base du bureau, que l'on pourrait considérer comme des programmes individuels, les environnements de bureau reprennent la métaphore du bureau en fournissant une panoplie minimale d'applications accessoires développées selon les mêmes directives de conception. Des variantes des applications suivantes sont couramment fournies par les principaux environnements de bureau :

Applications liées au système

Émulateur de terminal, gestionnaire de fichiers, gestionnaire de paquets, outils de configuration du système.

Communication et Internet

Gestionnaire de contacts, client de courrier électronique, navigateur web.

Applications de bureau

Calendrier, calculatrice, éditeur de texte.

Les environnements de bureau peuvent inclure beaucoup d'autres services et applications : écran d'accueil de connexion, gestionnaire de session, communication inter-processus, gestionnaire de trousseaux de clés, etc. Ils intègrent également les fonctionnalités fournies par des services système tiers comme *PulseAudio* pour le son ou *CUPS* pour l'impression. Ces services n'ont pas besoin de l'environnement graphique pour fonctionner, mais l'environnement de bureau fournit des interfaces graphiques pour faciliter leur configuration et leur exploitation.

Environnements de bureau populaires

La plupart des systèmes d'exploitation propriétaires ne prennent en charge qu'un seul environnement de bureau officiel, qui est lié à leur mouture particulière et qui n'est pas censé être modifié. À l'inverse, les systèmes d'exploitation Linux prennent en charge toute une série de choix d'environnements de bureau qui peuvent être utilisés en conjonction avec X. Chaque environnement de bureau possède ses propres caractéristiques, mais ils partagent généralement des concepts de design communs :

- Un lanceur d'applications qui répertorie les applications intégrées et tierces disponibles dans le système.
- Un ensemble de règles qui définissent les applications par défaut associées aux types de fichiers et aux protocoles.
- Des outils de configuration pour personnaliser l'apparence et le comportement de l'environnement de bureau.

Gnome est l'un des environnements de bureau les plus populaires et constitue le premier choix des distributions telles que Fedora, Debian, Ubuntu, SUSE Linux Enterprise, Red Hat Enterprise Linux, CentOS, etc. Dans sa version 3, Gnome a apporté des changements majeurs dans son apparence et sa structure, en s'éloignant de la métaphore du bureau et en introduisant le *Gnome Shell* comme nouvelle interface.

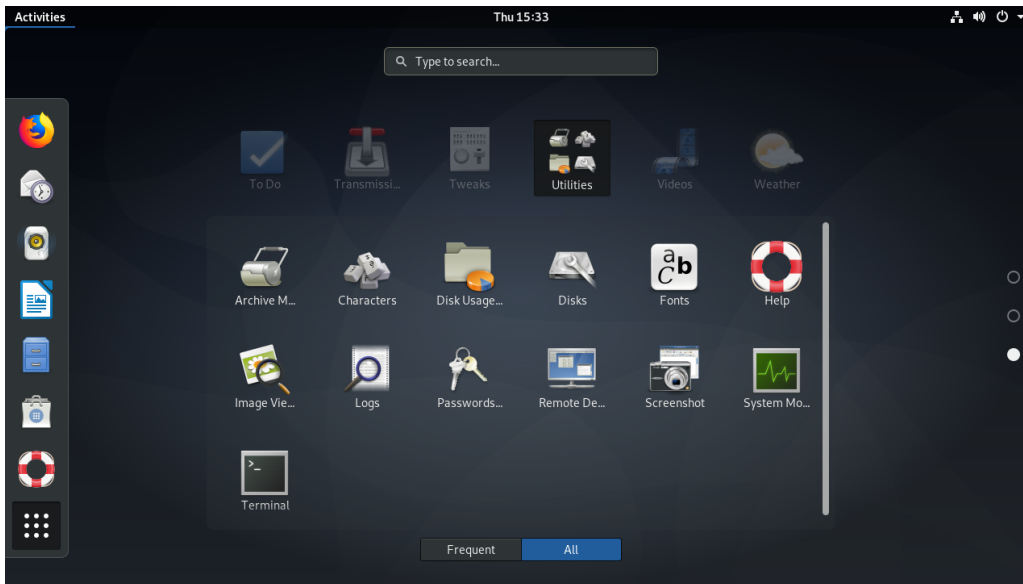


Figure 1. Gnome Shell Activities

Le lanceur plein écran *Gnome Shell Activities* a remplacé le lanceur d'applications et la barre des tâches traditionnels. En revanche, on peut toujours utiliser Gnome 3 avec l'ancienne apparence en choisissant l'option *Gnome Classic* dans le gestionnaire de connexion.

KDE est un vaste écosystème d'applications et une plateforme de développement. Sa dernière version d'environnement de bureau, *KDE Plasma*, est utilisée par défaut dans openSUSE, Mageia, Kubuntu, etc. L'utilisation de la bibliothèque Qt est la caractéristique principale de KDE, ce qui lui donne son apparence unique et une multitude d'applications originales. KDE propose même un outil de configuration pour assurer la cohérence visuelle avec les applications GTK+.

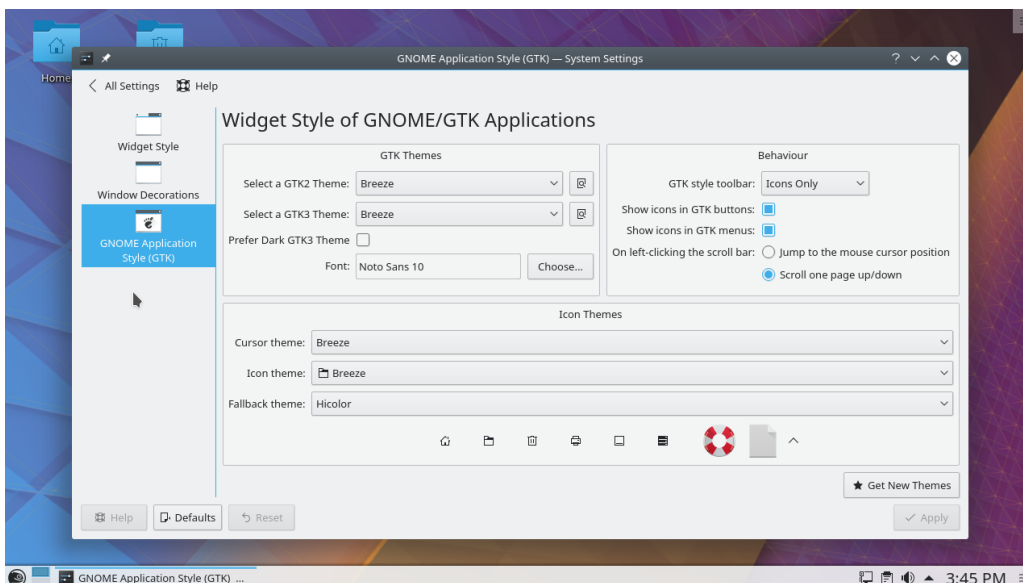


Figure 2. Réglages GTK de KDE

Xfce est un environnement de bureau qui se veut esthétique sans pour autant être gourmand en ressources machine. Sa structure est hautement modulaire, ce qui permet à l'utilisateur d'activer et de désactiver des composants en fonction de ses besoins et de ses préférences.

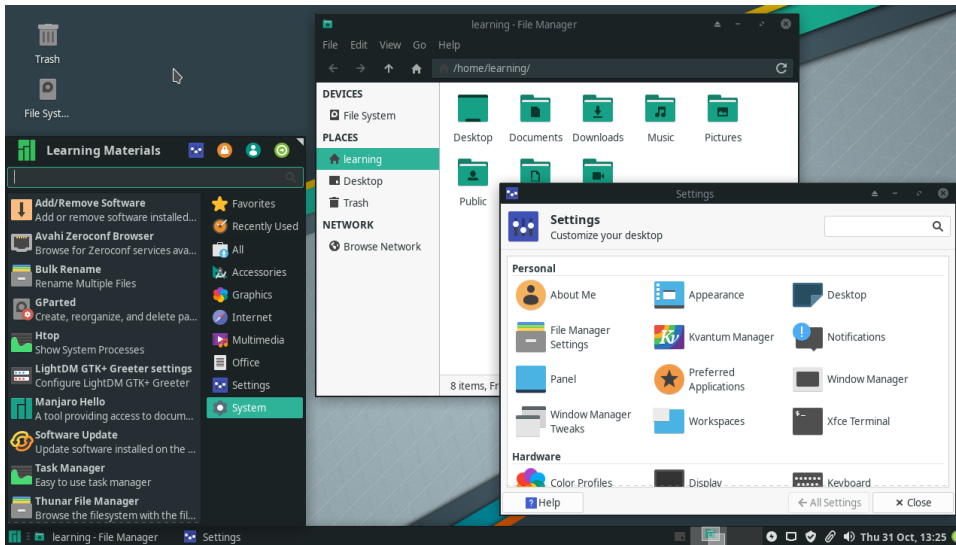


Figure 3. Le bureau Xfce

Il y a beaucoup d'autres environnements de bureau pour Linux, fournis en règle générale par les distributions alternatives. La distribution Linux Mint, par exemple, propose deux environnements de bureau originaux : *Cinnamon* (un *fork* de Gnome 3) et *MATE* (un *fork* de Gnome 2). *LXDE* est un environnement de bureau conçu pour consommer peu de ressources, ce qui en fait un bon choix pour une installation sur des équipements plus anciens ou des ordinateurs monocartes. Même s'il n'offre pas toutes les fonctionnalités des environnements de bureau plus élaborés, *LXDE* propose toutes les fonctionnalités de base que l'on attend d'une interface utilisateur graphique moderne.

TIP

Les raccourcis clavier jouent un rôle important dans l'interaction avec l'environnement de bureau. Certains raccourcis clavier — comme **Alt** + **Tab** pour passer d'une fenêtre à l'autre ou **Ctrl** + **C** pour copier du texte — sont communs à tous les environnements de bureau, mais chaque environnement de bureau dispose également de ses propres raccourcis clavier. Les raccourcis clavier se trouvent dans l'outil de configuration du clavier fourni par l'environnement de bureau, où ils peuvent être ajoutés ou modifiés.

Interopérabilité des bureaux

La grande variété d'environnements de bureau dans les systèmes d'exploitation Linux pose un dilemme : comment les faire fonctionner correctement avec des applications graphiques ou des services système tiers sans avoir à implémenter un support spécifique pour chacun d'entre eux. Des méthodes et des spécifications communes à tous les environnements de bureau améliorent

considérablement l'expérience de l'utilisateur et règlent bon nombre de problèmes de développement, étant donné que les applications graphiques doivent interagir avec l'environnement de bureau actuel, quel que soit l'environnement de bureau pour lequel elles ont été conçues à l'origine. En dehors de cela, il est important de conserver les paramètres généraux du bureau si l'utilisateur modifie éventuellement son choix d'environnement de bureau.

L'organisation *freedesktop.org* maintient un corpus important de spécifications pour l'interopérabilité des ordinateurs de bureau. L'adoption de l'ensemble des spécifications n'est pas obligatoire, mais bon nombre d'entre elles sont largement utilisées :

Emplacement des répertoires

L'endroit où se trouvent les paramètres personnels et d'autres fichiers spécifiques à l'utilisateur.

Entrées de menu

Les applications en ligne de commande peuvent être exécutées dans l'environnement de bureau via n'importe quel émulateur de terminal, mais il serait trop déroutant de les rendre toutes disponibles dans le lanceur d'applications. Les entrées de bureau sont des fichiers texte qui se terminent par `.desktop` et que l'environnement de bureau utilise pour rassembler des informations sur les applications de bureau disponibles et la manière de les utiliser.

Démarrage automatique

Configuration qui spécifie l'application censée démarrer automatiquement une fois que l'utilisateur s'est connecté.

Glisser-déposer

Comment les applications sont censées gérer les événements de type "glisser-déposer".

Corbeille

L'emplacement habituel des fichiers supprimés par le gestionnaire de fichiers, ainsi que les procédures de stockage et de suppression des fichiers à partir de cet emplacement.

Thèmes d'icônes

Le format commun pour les bibliothèques d'icônes interchangeables.

La facilité d'utilisation des environnements de bureau présente un inconvénient par rapport aux interfaces texte comme le shell : la possibilité de fournir un accès à distance. En effet, si l'on peut facilement accéder à un environnement shell en ligne de commande sur une machine distante à l'aide d'outils tels que `ssh`, l'accès à distance à un environnement graphique nécessite des méthodes différentes et peut s'avérer moins performant avec une connexion plus lente.

Accès non local

Le système X Window adopte une conception basée sur des *affichages (displays)* autonomes, où le même *Gestionnaire d'affichage X (X display manager)* peut gérer plus d'une session de bureau graphique en même temps. En fait, un affichage (*display*) est comparable à un terminal texte : les deux font référence à une machine ou à une application logicielle utilisée comme point d'entrée pour établir une session indépendante du système d'exploitation. Même si la configuration la plus courante implique une seule session graphique qui s'exécute sur la machine locale, d'autres configurations moins conventionnelles sont également possibles :

- Passer d'une session de bureau graphique active à une autre sur la même machine.
- Plus d'un jeu de périphériques d'affichage (écran, clavier, souris) connectés à la même machine, chacun contrôlant sa propre session de bureau graphique.
- Sessions de bureau graphique à distance, où l'interface graphique est envoyée via le réseau vers un écran distant.

Les sessions de bureau à distance sont prises en charge de manière native par X, qui utilise le *X Display Manager Control Protocol (XDMCP)* pour communiquer avec les écrans distants. En raison de ses besoins élevés en termes de bande passante, le protocole XDMCP est rarement utilisé sur Internet ou dans les réseaux locaux à débit modeste. Par ailleurs, XDMCP pose des problèmes de sécurité : l'affichage local communique avec un gestionnaire d'affichage X distant privilégié pour exécuter des tâches à distance, de sorte qu'une éventuelle vulnérabilité pourrait permettre d'exécuter des commandes arbitraires privilégiées sur la machine distante.

Qui plus est, XDMCP nécessite l'exécution d'instances X des deux côtés de la connexion, ce qui peut le rendre irréalisable si le système X Window n'est pas disponible sur toutes les machines concernées. Dans la pratique, d'autres méthodes plus efficaces et moins contraignantes sont utilisées pour établir des sessions de bureau graphique à distance.

Le *Virtual Network Computing (VNC)* est un outil autonome permettant de visualiser et de contrôler des environnements de bureau à distance via le protocole *Remote Frame Buffer (RFB)*. Grâce à ce protocole, les signaux émis par le clavier et la souris en local sont transmis au bureau distant, qui renvoie à son tour toutes les données actualisées de l'écran à afficher localement. On peut faire tourner plusieurs serveurs VNC sur la même machine, mais chacun de ces serveurs aura besoin d'un port TCP propre dans l'interface réseau pour accepter les demandes de session entrantes. Par convention, le premier serveur VNC devra utiliser le port TCP 5900, le second le port 5901, et ainsi de suite.

Le serveur VNC n'a pas besoin de privilèges particuliers pour fonctionner. Un utilisateur lambda peut très bien se connecter à son compte distant et démarrer son propre serveur VNC à partir de

là. Ensuite, sur la machine locale, n'importe quelle application client VNC peut être utilisée pour accéder au bureau distant (en supposant que les ports réseau correspondants soient accessibles). Le fichier `~/.vnc/xstartup` est un script shell exécuté par le serveur VNC lorsqu'il démarre et peut être utilisé pour définir l'environnement de bureau que le serveur VNC mettra à la disposition du client VNC. Il est important de noter que VNC ne fournit pas de méthodes modernes de chiffrement et d'authentification de manière native, il devra donc être utilisé en combinaison avec une application tierce qui fournit ces fonctionnalités. Les méthodes impliquant des tunnels VPN et SSH sont souvent utilisées pour sécuriser les connexions VNC.

Le *Remote Desktop Protocol* (RDP) est principalement utilisé pour accéder à distance au bureau d'un système d'exploitation *Microsoft Windows* via le port réseau TCP 3389. Même s'il utilise le protocole RDP propriétaire de Microsoft, l'implémentation client utilisée dans les systèmes Linux consiste en une série de logiciels libres sous licence *GNU General Public License* (GPL) et ne fait l'objet d'aucune restriction légale d'utilisation.

Le protocole Spice (*Simple Protocol for Independent Computing Environments*) comprend une suite d'outils qui permettent d'accéder à l'environnement de bureau de systèmes *virtualisés*, que ce soit sur la machine locale ou sur un site distant. En dehors de cela, le protocole Spice offre des fonctionnalités natives pour intégrer les systèmes locaux et distants, comme la possibilité d'accéder aux périphériques locaux (haut-parleurs et périphériques USB connectés par exemple) à partir de la machine distante et le partage de fichiers entre les deux systèmes.

Il existe des commandes client spécifiques pour se connecter à chacun de ces protocoles de bureau à distance, mais le client de bureau à distance *Remmina* fournit une interface graphique intégrée qui facilite le processus de connexion, en conservant éventuellement les paramètres de connexion pour une utilisation ultérieure. Remmina dispose de plugins pour chaque protocole spécifique et il existe des plugins pour XDMCP, VNC, RDP et Spice. Le choix de l'outil approprié se fait selon les systèmes d'exploitation concernés, la qualité de la connexion réseau et les fonctionnalités de l'environnement de bureau à distance qui doivent être disponibles.

Exercices guidés

1. Quel type d'application offre des sessions shell en mode fenêtré dans l'environnement de bureau ?

2. En raison de la variété des environnements de bureau Linux, la même application peut avoir plus d'une mouture, chacune d'entre elles étant mieux adaptée à un *toolkit* particulier. Par exemple, le client bittorrent *Transmission* dispose de deux versions : *transmission-gtk* et *transmission-qt*. Laquelle des deux doit être installée pour assurer une intégration parfaite avec KDE ?

3. Quels sont les environnements de bureau Linux recommandés pour les ordinateurs monocartes bas de gamme dotés d'une faible puissance de calcul ?

Exercices d'approfondissement

1. Il existe deux manières de copier et de coller du texte dans le système X Window : en utilisant les touches traditionnelles `ctrl` + `c` et `ctrl` + `v` (également disponibles dans le menu de la fenêtre) ou en utilisant le clic du milieu de la souris pour coller le texte sélectionné. Quelle est la méthode appropriée pour copier et coller du texte depuis l'émulateur de terminal ?

2. La plupart des environnements de bureau associent le raccourci clavier `Alt` + `F2` à la fenêtre *Exécuter un programme*, où un programme peut être exécuté en ligne de commande. Dans KDE, quelle commande permet d'exécuter l'émulateur de terminal par défaut ?

3. Quel est le protocole le mieux adapté pour accéder à un bureau Windows distant depuis un environnement de bureau Linux ?

Résumé

Cette leçon présente une vue d'ensemble des bureaux graphiques disponibles pour les systèmes Linux. Le système X Window ne fournit à lui seul que des fonctions d'interface simples, de sorte que les environnements de bureau améliorent l'expérience de l'utilisateur dans l'interface graphique fenêtrée. La leçon aborde les sujets suivants :

- Concepts relatifs à l'interface graphique et au système X Window.
- Environnements de bureau disponibles pour Linux.
- Points communs et différences entre les environnements de bureau.
- Comment accéder à un environnement de bureau à distance.

Les notions et les programmes suivants ont été abordés :

- Système X Window.
- Environnements de bureau populaires : KDE, Gnome, Xfce.
- Protocoles d'accès à distance : XDMCP, VNC, RDP, Spice.

Réponses aux exercices guidés

1. Quel type d'application offre des sessions shell en mode fenêtré dans l'environnement de bureau ?

N'importe quel émulateur de terminal comme Konsole, Gnome terminal, xterm, etc., permet d'accéder à une session interactive locale du shell.

2. En raison de la variété des environnements de bureau Linux, la même application peut avoir plus d'une mouture, chacune d'entre elles étant mieux adaptée à un *toolkit* particulier. Par exemple, le client bittorrent *Transmission* dispose de deux versions : *transmission-gtk* et *transmission-qt*. Laquelle des deux doit être installée pour assurer une intégration parfaite avec KDE ?

KDE est basé sur la bibliothèque Qt, c'est donc la version Qt — *transmission-qt* — qui devra être installée.

3. Quels sont les environnements de bureau Linux recommandés pour les ordinateurs monocartes bas de gamme dotés d'une faible puissance de calcul ?

Les environnements de bureau basiques qui n'utilisent pas trop d'effets visuels, comme Xfce et LXDE.

Réponses aux exercices d'approfondissement

1. Il existe deux manières de copier et de coller du texte dans le système X Window : en utilisant les touches traditionnelles `Ctrl` + `c` et `Ctrl` + `v` (également disponibles dans le menu de la fenêtre) ou en utilisant le clic du milieu de la souris pour coller le texte sélectionné. Quelle est la méthode appropriée pour copier et coller du texte depuis l'émulateur de terminal ?

Les sessions interactives de l'interpréteur de commandes attribuent la touche `Ctrl` + `c` pour arrêter l'exécution du programme, c'est pourquoi la méthode du bouton central est recommandée.

2. La plupart des environnements de bureau associent le raccourci clavier `Alt` + `F2` à la fenêtre *Exécuter un programme*, où un programme peut être exécuté en ligne de commande. Dans KDE, quelle commande permet d'exécuter l'émulateur de terminal par défaut ?

La commande `konsole` exécute l'émulateur de terminal de KDE, mais un terme générique comme *terminal* fonctionne également.

3. Quel est le protocole le mieux adapté pour accéder à un bureau Windows distant depuis un environnement de bureau Linux ?

Le protocole de bureau à distance (RDP), étant donné qu'il est pris en charge de manière native par Windows et Linux.



**Linux
Professional
Institute**

106.3 Accessibilité

Référence aux objectifs de LPI

LPIC-1 version 5.0, Exam 102, Objective 106.3

Valeur

1

Domaines de connaissance les plus importants

- Connaissance de base des paramètres d’affichage et des thèmes.
- Connaissance de base des outils d’accessibilité.

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

- Thèmes du bureau à fort contraste ou grandes polices.
- Lecteur d’écran.
- Lecteur braille.
- Loupe d’écran.
- Clavier virtuel.
- Touches Difficiles/Répétition.
- Touches lentes/rebond/inverser.
- Émulation de la souris au clavier.
- Gestuelles.
- Reconnaissance vocale.



106.3 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	106 Interfaces utilisateur et bureaux
Objectif :	106.3 Accessibilité
Leçon :	1 sur 1

Introduction

L'environnement de bureau Linux dispose d'un certain nombre de paramètres et d'outils qui permettent d'adapter l'interface utilisateur aux personnes handicapées. Les périphériques habituels de l'interface humaine — écran, clavier et souris/pavé tactile — peuvent être reconfigurés individuellement pour pallier les déficiences visuelles ou la mobilité réduite.

À titre d'exemple, on peut ajuster la palette de couleurs du bureau pour mieux répondre aux besoins des daltoniens. De même, les personnes souffrant de microtraumatismes répétés pourront utiliser des méthodes alternatives de saisie et de pointage.

Parmi ces éléments d'accessibilité, certains sont fournis par l'environnement de bureau lui-même, comme Gnome ou KDE, alors que d'autres sont assurés par des applications externes. Dans ce dernier cas, il est important de choisir l'outil qui s'intègre le mieux à l'environnement de bureau afin que l'aide soit de meilleure qualité.

Paramètres d'accessibilité

Toutes les grandes distributions Linux présentent grosso modo les mêmes éléments d'accessibilité,

qui peuvent être personnalisés à l'aide d'un module de configuration qui se trouve dans le gestionnaire de paramètres fourni avec l'environnement de bureau. Le module de configuration de l'accessibilité s'appelle *Accès universel* dans le bureau Gnome, tandis que sous KDE, il se trouve sous *Paramètres du système, Personnalisation, Accessibilité*. D'autres environnements de bureau, comme *Xfce*, l'appellent également *Accessibilité* dans leur gestionnaire de paramètres graphique. En revanche, ils offrent un ensemble réduit de fonctionnalités par rapport à Gnome et KDE.

Gnome, par exemple, peut être configuré pour afficher en permanence le menu *Accès universel* dans le coin supérieur droit de l'écran, où les options d'accessibilité pourront être activées rapidement. Ce menu pourra notamment être utilisé pour remplacer l'alerte sonore par une alerte visuelle, de manière à ce que les utilisateurs malentendants puissent percevoir plus facilement les alertes du système. Même si KDE ne dispose pas d'un menu d'accès rapide analogue, la fonction d'alerte visuelle est également disponible, mais elle s'appelle *visual bell* (cloche visuelle).

Assistance pour le clavier et la souris

Le comportement par défaut du clavier et de la souris peut être modifié pour surmonter les difficultés de mobilité spécifiques. Les combinaisons de touches, le taux de répétition automatique des touches et les frappes involontaires peuvent constituer des obstacles importants pour les utilisateurs ayant une mobilité réduite de la main. Ces défauts de frappe sont pris en compte par trois fonctions d'accessibilité liées au clavier : *Touches rémanentes*, *Rebonds de touches* et *Touches lentes*.

La fonction *Touches rémanentes* qui se trouve dans la section *Saisie* de la configuration *Accès universel* de Gnome, permet à l'utilisateur de taper les raccourcis clavier une touche à la fois. Lorsqu'elle est activée, les combinaisons de touches telles que **Ctrl** + **C** n'ont pas besoin d'être enfoncées simultanément. L'utilisateur peut d'abord appuyer sur la touche **Ctrl**, la relâcher puis appuyer sur la touche **C**. Sous KDE, cette option se trouve dans l'onglet *Touches de modifications* de la fenêtre de paramétrage de l'accessibilité. KDE propose également l'option *Touches avec auto-maintien* : si elle est activée, les touches **Alt**, **Ctrl** et **Majuscule** resteront "en bas" si l'utilisateur appuie deux fois dessus, un peu comme la touche de verrouillage des majuscules. Là aussi, l'utilisateur devra appuyer à nouveau sur la touche correspondante pour la relâcher.

La fonction *Rebonds de touches* tente d'éviter les frappes involontaires en instaurant un délai entre elles, c'est-à-dire qu'une nouvelle pression sur une touche ne sera acceptée qu'après un certain laps de temps depuis la dernière pression sur une touche. Les utilisateurs souffrant de tremblements de la main peuvent trouver cette fonction utile pour éviter d'appuyer plusieurs fois sur une touche alors qu'ils n'ont l'intention de le faire qu'une seule fois. Sous Gnome, cette fonctionnalité ne concerne que les répétitions de la même touche, tandis que sous KDE, elle concerne également toute autre pression de touche et se trouve dans l'onglet *Filtres de clavier*.

La fonction *Touches lentes* permet également d'éviter les frappes accidentelles. Lorsqu'elle est activée, les touches lentes obligent l'utilisateur à maintenir la touche enfoncée pendant une durée déterminée avant qu'elle ne soit acceptée. Selon les besoins de l'utilisateur, il peut également être utile de régler la répétition automatique lorsqu'une touche est maintenue enfoncée, dans les paramètres du clavier.

Les fonctions d'accessibilité *Touches rémanentes* et *Touches lentes* peuvent être activées et désactivées par des *Gestes d'activation* effectués sur le clavier. Sous KDE, l'option *Utiliser les gestes pour activer les touches rémanentes et les touches lentes* doit être cochée pour activer les gestes d'activation, tandis que sous Gnome, cette fonctionnalité est appelée *Activer par le clavier* dans la fenêtre de configuration de l'*Assistance à la frappe*. Une fois les gestes d'activation validés, la fonction *Touches rémanentes* sera activée après avoir appuyé cinq fois de suite sur la touche `Majuscule`. Pour activer la fonction *Touches lentes*, la touche `Majuscule` devra être maintenue enfoncée pendant huit secondes consécutives.

Les utilisateurs qui trouvent plus confortable d'utiliser le clavier plutôt que la souris ou le pavé tactile peuvent se servir des raccourcis clavier pour naviguer dans l'environnement de bureau. Par ailleurs, une option appelée *Navigation au clavier* permet à l'utilisateur de contrôler le pointeur de la souris lui-même à l'aide du pavé numérique, que l'on trouve sur les claviers de bureau de taille normale et sur les ordinateurs portables de plus grande taille.

Le pavé numérique est organisé en grille carrée, de sorte que chaque chiffre correspond à une direction : `2` déplace le curseur vers le bas, `4` déplace le curseur vers la gauche, `7` déplace le curseur vers le nord-ouest, etc. Par défaut, la touche `5` correspond au clic gauche de la souris.

Sous Gnome, il n'y a qu'un seul bouton pour activer l'option *Touches de la souris* dans la fenêtre des paramètres *Accès universel*. Sous KDE, les paramètres des touches de la souris se trouvent dans *Paramètres du système*, *Souris*, *Navigation au clavier*, et les options telles que la vitesse et l'accélération peuvent être personnalisées.

TIP

Les touches lentes, les touches rémanentes, les rebonds de touches et la navigation au clavier sont des fonctions d'accessibilité fournies par AccessX, une ressource de l'extension clavier X du système X Window. Les paramètres d'AccessX peuvent également être modifiés à partir de la ligne de commande, avec la commande `xkbset`.

La souris ou le pavé tactile peuvent être utilisés pour générer des saisies au clavier lorsque l'utilisation du clavier est trop inconfortable voire impossible. Lorsque le bouton *Clavier à l'écran* des paramètres *Accès universel* de Gnome est activé, un clavier apparaît à l'écran chaque fois que le curseur se trouve dans un champ de texte et le nouveau texte est saisi en cliquant sur les touches à l'aide de la souris ou de l'écran tactile, un peu comme le fait le clavier virtuel des

smartphones.

KDE et d'autres environnements de bureau peuvent ne pas fournir le clavier à l'écran par défaut, mais le paquet *onboard* peut être installé manuellement pour fournir un simple clavier à l'écran qui pourra être utilisé dans n'importe quel environnement de bureau. Après l'installation, il sera disponible comme une application classique dans le lanceur d'applications.

Le comportement du pointeur peut également être modifié si le fait de cliquer et de faire glisser la souris est douloureux ou impraticable pour toute autre raison. Si l'utilisateur n'est pas en mesure de cliquer sur le bouton de la souris assez rapidement pour déclencher un double-clic, par exemple, le laps de temps nécessaire pour appuyer une deuxième fois sur le bouton de la souris afin de double-cliquer pourra être augmenté dans les *Préférences de la souris* de la fenêtre de configuration du système.

Si l'utilisateur n'est pas en mesure d'appuyer sur l'un des boutons de la souris ou sur aucun bouton, il est possible de simuler des clics de souris à l'aide de différentes techniques. Dans la section *Assistant de clic* du menu *Accès universel*, l'option *Simulation du clic secondaire* génère un clic du bouton droit si l'utilisateur appuie sur le bouton gauche de la souris et le maintient enfoncé. Si l'option *Clic par survol* est activée, un événement de clic sera déclenché lorsque l'utilisateur maintiendra la souris immobile. Sous KDE, l'application *KMouseTool* offre les mêmes fonctionnalités pour assister aux actions de la souris.

Déficiences visuelles

Les utilisateurs dont la vue est réduite peuvent toujours utiliser l'écran du moniteur pour interagir avec l'ordinateur. En fonction des besoins de l'utilisateur, de nombreux ajustements visuels peuvent être effectués pour améliorer les détails du bureau graphique standard qui seraient autrement difficiles à voir.

La section *Vision* de Gnome, dans les paramètres *Accès universel*, propose des options qui peuvent aider les personnes souffrant de troubles de la vue :

Contraste élevé

les fenêtres et les boutons sont plus faciles à voir grâce à des couleurs plus vives.

Grand texte

agrandit la taille de la police standard de l'écran.

Taille du curseur

permet de choisir un curseur de souris plus grand, ce qui le rend plus facile à localiser sur l'écran.

Certains de ces ajustements ne sont pas strictement liés aux fonctions d'accessibilité et peuvent donc être trouvés dans la section *Apparence* de l'utilitaire de configuration fourni par d'autres environnements de bureau. Un utilisateur qui éprouve des difficultés à discerner les éléments visuels pourra opter pour un thème à contraste élevé afin de faciliter l'identification des boutons, des fenêtres qui se chevauchent, etc.

Si les ajustements d'apparence ne suffisent pas à améliorer la lisibilité sur l'écran, il est possible d'utiliser un programme d'agrandissement pour zoomer sur certaines parties de l'écran. Cette fonction est appelée *Zoom* dans les paramètres *Accès universel* de Gnome, où des options telles que le taux d'agrandissement, la position de la loupe et les ajustements de couleur peuvent être personnalisées.

Sous KDE, le programme *KMagnifier* offre les mêmes fonctionnalités, mais il est disponible en tant qu'application classique via le lanceur d'applications. D'autres environnements de bureau peuvent fournir leurs propres loupes d'écran. Xfce, par exemple, permet d'agrandir ou de réduire l'écran en faisant tourner la molette de la souris lorsque la touche  est enfoncée.

Enfin, les utilisateurs pour lesquels l'interface graphique n'est pas envisageable peuvent utiliser un *lecteur d'écran* pour interagir avec l'ordinateur. Quel que soit l'environnement de bureau choisi, le lecteur d'écran le plus populaire pour les systèmes Linux est *Orca*, qui est généralement installé par défaut dans la plupart des distributions. Orca génère une voix synthétisée pour signaler les événements à l'écran et pour lire le texte sous le curseur de la souris. Orca fonctionne également avec les *afficheurs braille*, des appareils spéciaux qui affichent des caractères braille en soulevant de petits picots que l'on peut sentir du bout des doigts. Toutes les applications de bureau ne sont pas entièrement adaptées aux lecteurs d'écran et tous les utilisateurs ne trouveront pas facile de les utiliser, c'est pourquoi il est important de fournir autant de stratégies de lecture d'écran que possible pour que les utilisateurs puissent choisir.

Exercices guidés

1. Quelle option d'accessibilité pourrait aider un utilisateur à basculer entre les fenêtres ouvertes à l'aide du clavier, sachant qu'il n'est pas capable d'appuyer en même temps sur les touches **Alt** et **Tab** ?

2. Comment l'option d'accessibilité *Rebonds de touches* pourrait-elle aider les utilisateurs dont les tremblements involontaires de la main perturbent la frappe ?

3. Quel est le geste d'activation le plus courant pour l'option d'accessibilité *Touches rémanentes* ?

Exercices d'approfondissement

1. Les éléments d'accessibilité ne sont pas forcément fournis par une seule application et peuvent varier d'un environnement de bureau à l'autre. Sous KDE, quelle est l'application qui aide les personnes souffrant de microtraumatismes répétés en cliquant sur la souris chaque fois que le curseur s'arrête brièvement ?

2. Quels aspects de l'environnement graphique peuvent être modifiés pour faciliter la lecture du texte à l'écran ?

3. De quelle manière l'application *Orca* peut-elle aider les utilisateurs malvoyants à interagir avec l'environnement de bureau ?

Résumé

Cette leçon présente les options d'accessibilité généralement disponibles dans les systèmes Linux. Tous les principaux environnements de bureau, notamment Gnome et KDE, proposent une série d'applications intégrées et tierces pour aider les personnes souffrant de déficiences visuelles ou à mobilité réduite. La leçon aborde les points suivants :

- Comment modifier les paramètres d'accessibilité.
- Différentes façons d'utiliser le clavier et la souris.
- Améliorations du bureau pour les malvoyants.

Voici les commandes et les procédures abordées :

- Paramètres d'accessibilité du clavier : Touches rémanentes, touches lentes, rebonds de touches.
- Génération artificielle d'événements liés à la souris.
- Clavier à l'écran.
- Paramètres visuels pour améliorer la lisibilité.
- Thèmes de bureau à fort contraste/grande taille de caractères.
- Loupes d'écran.
- Lecteur d'écran Orca.

Réponses aux exercices guidés

1. Quelle option d'accessibilité pourrait aider un utilisateur à basculer entre les fenêtres ouvertes à l'aide du clavier, sachant qu'il n'est pas capable d'appuyer en même temps sur les touches **Alt** et **Tab** ?

La fonction *Touches rémanentes*, qui permet à l'utilisateur de taper des raccourcis clavier une touche à la fois.

2. Comment l'option d'accessibilité *Rebonds de touches* pourrait-elle aider les utilisateurs dont les tremblements involontaires de la main perturbent la frappe ?

Lorsque l'option *Rebonds de touches* est activée, une nouvelle pression sur une touche n'est acceptée qu'après un laps de temps déterminé depuis la dernière pression sur une touche.

3. Quel est le geste d'activation le plus courant pour l'option d'accessibilité *Touches rémanentes* ?

Si les gestes d'activation sont utilisés, la fonction *Touches rémanentes* sera activée après avoir appuyé cinq fois de suite sur la touche **Majuscule**.

Réponses aux exercices d'approfondissement

1. Les éléments d'accessibilité ne sont pas forcément fournis par une seule application et peuvent varier d'un environnement de bureau à l'autre. Sous KDE, quelle est l'application qui aide les personnes souffrant de microtraumatismes répétés en cliquant sur la souris chaque fois que le curseur s'arrête brièvement ?

L'application *KMouseTool*.

2. Quels aspects de l'environnement graphique peuvent être modifiés pour faciliter la lecture du texte à l'écran ?

La définition d'une taille de police élevée dans la configuration du bureau facilitera la lecture de tous les textes affichés à l'écran.

3. De quelle manière l'application *Orca* peut-elle aider les utilisateurs malvoyants à interagir avec l'environnement de bureau ?

Orca est un lecteur d'écran qui génère une voix synthétisée pour signaler les événements à l'écran et pour lire le texte sous le curseur de la souris. Il fonctionne également avec des appareils appelés *afficheurs braille*, afin que l'utilisateur puisse identifier le texte à l'aide de motifs tactiles.



Thème 107 : Tâches d'administration



107.1 Gestion des comptes utilisateurs et des groupes ainsi que des fichiers systèmes concernés

Référence aux objectifs de LPI

LPIC-1 version 5.0, Exam 102, Objective 107.1

Valeur

5

Domaines de connaissance les plus importants

- Ajout, modification et suppression d'utilisateurs et de groupes.
- Gestion des informations associées aux utilisateurs et aux groupes dans les fichiers de bases de données système.
- Création et gestion de comptes pour des usages spécifiques et limités.

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

- `/etc/passwd`
- `/etc/shadow`
- `/etc/group`
- `/etc/skel/`
- `chage`
- `getent`
- `groupadd`
- `groupdel`
- `groupmod`
- `passwd`

- `useradd`
- `userdel`
- `usermod`



107.1 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	107 Tâches administratives
Objectif :	107.1 Gérer les comptes utilisateurs et les groupes ainsi que les fichiers système correspondants
Leçon :	1 sur 2

Introduction

La gestion des utilisateurs et des groupes est une partie très importante du travail de tout administrateur système. Les distributions Linux modernes mettent en œuvre des interfaces graphiques qui vous permettent de gérer rapidement et facilement toutes les activités liées à cet aspect fondamental. Ces interfaces peuvent varier en termes de présentation graphique, mais les fonctionnalités restent les mêmes. Ces outils vous permettent de visualiser, de modifier, d'ajouter et de supprimer des utilisateurs et des groupes locaux. Cependant, pour une gestion plus avancée, vous devez utiliser la ligne de commande.

Ajouter des comptes utilisateurs

Sous Linux, vous pouvez ajouter un nouveau compte utilisateur avec la commande `useradd`. Par exemple, en tant que root, vous pouvez créer un nouveau compte utilisateur nommé `michael` avec une configuration par défaut, en utilisant la commande suivante :

```
# useradd michael
```

Lorsque vous utilisez la commande `useradd`, les informations sur les utilisateurs et les groupes stockées dans les bases de données de mots de passe et de groupes sont mises à jour pour le compte utilisateur nouvellement créé et, si cela est spécifié, le répertoire personnel du nouvel utilisateur est également créé. Un groupe portant le même nom que le nouveau compte utilisateur est également créé.

Une fois que vous avez créé le nouvel utilisateur, vous pouvez définir son mot de passe en utilisant la commande `passwd`. Vous pouvez consulter son identifiant utilisateur (UID), son identifiant de groupe (GID) et les groupes auxquels il appartient avec les commandes `id` et `groups`.

```
# passwd michael
Changing password for user michael.
New UNIX password:
Retype new UNIX password:
passwd: all authentication tokens updated successfully.
# id michael
uid=1000(michael) gid=100(michael) groups=100(michael)
# groups michael
michael : michael
```

NOTE

Gardez à l'esprit que n'importe quel utilisateur peut consulter son UID, son GID et les groupes auxquels il appartient en utilisant simplement les commandes `id` et `groups` sans arguments, et que n'importe quel utilisateur peut changer son mot de passe en utilisant la commande `passwd`. En revanche, seuls les utilisateurs avec les privilèges de root peuvent modifier le mot de passe de *n'importe quel* utilisateur.

Voici les options les plus importantes qui s'appliquent à la commande `useradd` :

-c

Créer un nouveau compte utilisateur avec un commentaire personnalisé (par exemple le nom complet de l'utilisateur).

-d

Créez un nouveau compte utilisateur avec un répertoire personnel spécifique.

-e

Créer un nouveau compte utilisateur en précisant la date à laquelle il sera désactivé.

-f

Créer un nouveau compte utilisateur en définissant le nombre de jours après l'expiration du mot de passe pendant lesquels l'utilisateur devra mettre à jour son mot de passe (faute de quoi le compte sera désactivé).

-g

Créer un nouveau compte utilisateur avec un GID spécifique.

-G

Créer un nouveau compte utilisateur en l'ajoutant à plusieurs groupes secondaires.

-k

Créer un nouveau compte utilisateur en copiant les fichiers squelette depuis un répertoire donné (cette option n'est valide que si l'option `-m` ou `--create-home` est spécifiée).

-m

Créer un nouveau compte utilisateur avec son répertoire personnel (s'il n'existe pas).

-M

Créer un nouveau compte utilisateur sans le répertoire personnel.

-s

Créer un nouveau compte utilisateur avec un shell de connexion spécifique.

-u

Créer un nouveau compte utilisateur avec un UID spécifique.

Consultez la page de manuel de la commande `useradd` pour la liste complète des options.

Modifier les comptes utilisateurs

Parfois, il vous faut changer un paramètre d'un compte utilisateur existant, comme le nom d'utilisateur (*login name*), le shell de connexion, la date d'expiration du mot de passe et ainsi de suite. Dans ce cas, vous pouvez utiliser la commande `usermod`.

```
# usermod -s /bin/tcsh michael
# usermod -c "Michael User Account" michael
```

Tout comme la commande `useradd`, la commande `usermod` requiert les privilèges de root.

Dans les exemples ci-dessus, le shell de connexion de `michael` est modifié dans un premier temps, puis une description succincte est ajoutée à ce compte d'utilisateur. Notez que vous pouvez modifier plusieurs paramètres à la fois, en les spécifiant dans une seule commande.

Voici les options les plus importantes qui s'appliquent à la commande `usermod` :

-c

Ajouter un commentaire succinct au compte utilisateur spécifié.

-d

Modifier le répertoire personnel du compte utilisateur spécifié. Si cette option est utilisée avec l'option `-m`, le contenu du répertoire personnel en cours sera déplacé vers le nouveau répertoire personnel, qui sera créé s'il n'existe pas déjà.

-e

Définir la date d'expiration du compte utilisateur spécifié.

-f

Définir le nombre de jours après l'expiration du mot de passe pendant lesquels l'utilisateur devra mettre à jour son mot de passe (faute de quoi le compte sera désactivé).

-g

Modifier le groupe primaire du compte utilisateur spécifié (le groupe doit exister).

-G

Ajouter des groupes secondaires au compte utilisateur spécifié. Chaque groupe doit exister et doit être séparé du suivant par une virgule, sans espace blanc intermédiaire. Si elle est utilisée seule, cette option supprime tous les groupes existants auxquels l'utilisateur appartient, tandis que si elle est utilisée avec l'option `-a`, elle ajoute simplement les nouveaux groupes secondaires aux groupes existants.

-l

Modifier nom d'utilisateur (*login*) du compte spécifié.

-L

Verrouiller le compte utilisateur spécifié. Cette opération ajoute un point d'exclamation devant le mot de passe chiffré dans le fichier `/etc/shadow`, ce qui désactive l'accès avec un mot de passe pour cet utilisateur.

-s

Modifier le shell de connexion du compte utilisateur spécifié.

-u

Modifier l'UID du compte utilisateur spécifié.

-U

Déverrouiller le compte utilisateur spécifié. Cette opération supprime le point d'exclamation devant le mot de passe chiffré dans le fichier `/etc/shadow`.

Consultez la page de manuel de la commande `usermod` pour la liste complète des options.

TIP

N'oubliez pas que lorsque vous modifiez le login d'un compte utilisateur, vous devrez probablement renommer le répertoire personnel de cet utilisateur et d'autres éléments connexes comme les fichiers spool de courrier. De même, si vous modifiez l'UID d'un compte utilisateur, vous devrez probablement corriger les droits d'accès des fichiers et des répertoires situés en dehors du répertoire personnel de l'utilisateur (l'UID est modifié automatiquement pour la boîte aux lettres de l'utilisateur et pour tous les fichiers qui lui appartiennent et qui se trouvent dans son répertoire personnel).

Supprimer des comptes utilisateurs

Si vous souhaitez supprimer un compte utilisateur, vous pouvez utiliser la commande `userdel`. Plus précisément, cette commande met à jour les informations stockées dans les bases de données des comptes en supprimant toutes les entrées relatives à l'utilisateur spécifié. L'option `-r` supprimera également le répertoire personnel de l'utilisateur et tout son contenu ainsi que le spool de courrier de l'utilisateur. Les autres fichiers situés ailleurs devront être recherchés et supprimés manuellement.

```
# userdel -r michael
```

Tout comme pour `useradd` et `usermod`, vous avez besoin des droits de root pour supprimer des comptes utilisateurs.

Ajouter, modifier et supprimer des groupes

Tout comme pour la gestion des utilisateurs, vous pouvez ajouter, modifier et supprimer des groupes en utilisant les commandes `groupadd`, `groupmod` et `groupdel` avec les privilèges de root. Si vous souhaitez créer un nouveau groupe nommé `developer`, vous pouvez exécuter la commande suivante :

```
# groupadd -g 1090 developer
```

L'option `-g` de cette commande crée un groupe avec un GID spécifique.

WARNING

Rappelez-vous que lorsque vous ajoutez un nouveau compte utilisateur, le groupe primaire et les groupes secondaires auxquels il appartient *doivent* exister avant de lancer la commande `useradd`.

Par la suite, si vous voulez renommer le groupe `developer` en `web-developer` en modifiant son GID, vous pouvez exécuter la commande suivante :

```
# groupmod -n web-developer -g 1050 developer
```

TIP

Rappelez-vous que si vous changez le GID en utilisant l'option `-g`, vous devrez changer le GID de tous les fichiers et répertoires censés appartenir au groupe.

Enfin, si vous voulez supprimer le groupe `web-developer`, vous pouvez invoquer cette commande :

```
# groupdel web-developer
```

Vous ne pouvez pas supprimer un groupe s'il s'agit du groupe primaire d'un compte utilisateur. Vous devrez donc supprimer l'utilisateur avant de supprimer le groupe. Tout comme pour les utilisateurs, si vous supprimez un groupe, les fichiers appartenant à ce groupe restent dans votre système de fichiers et ne sont pas supprimés ou attribués à un autre groupe.

Le répertoire squelette

Lorsque vous ajoutez un nouveau compte utilisateur, même en créant son répertoire personnel, ce dernier est peuplé de fichiers et de dossiers copiés depuis le répertoire squelette (par défaut `/etc/skel`). L'idée derrière ceci est simple : un administrateur système veut ajouter de nouveaux utilisateurs avec les mêmes fichiers et répertoires dans leur répertoire personnel. Par conséquent, si vous voulez personnaliser les fichiers et les dossiers qui sont créés automatiquement dans le répertoire personnel des nouveaux comptes utilisateurs, vous devrez ajouter ces nouveaux fichiers et dossiers au répertoire squelette.

TIP

Notez que si vous voulez afficher la liste de tous les fichiers et répertoires du répertoire squelette, vous devrez utiliser la commande `ls -al`.

Le fichier `/etc/login.defs`

Sous Linux, le fichier `/etc/login.defs` définit les paramètres de configuration qui régissent la création des utilisateurs et des groupes. Par ailleurs, les commandes présentées dans les sections précédentes puisent leurs valeurs par défaut dans ce fichier.

Voici les principales directives :

UID_MIN et UID_MAX

La plage des identifiants utilisateurs qui peuvent être attribués aux nouveaux utilisateurs ordinaires.

GID_MIN et GID_MAX

La plage des identifiants de groupe qui peuvent être attribués aux nouveaux groupes ordinaires.

CREATE_HOME

Indique si un répertoire personnel doit être créé par défaut pour les nouveaux utilisateurs.

USERGROUPS_ENAB

Indique si le système doit créer par défaut un nouveau groupe pour chaque nouveau compte utilisateur avec le même nom que l'utilisateur, et si la suppression du compte utilisateur a pour effet de supprimer également le groupe primaire de l'utilisateur s'il ne compte plus de membres.

MAIL_DIR

Le répertoire de stockage du courrier.

PASS_MAX_DAYS

Le nombre maximum de jours pendant lesquels un mot de passe pourra être utilisé.

PASS_MIN_DAYS

Le nombre minimum de jours autorisés entre deux changements de mot de passe.

PASS_MIN_LEN

La longueur minimale acceptable du mot de passe.

PASS_WARN_AGE

Le nombre de jours de préavis avant l'expiration d'un mot de passe.

TIP | Lorsque vous gérez des utilisateurs et des groupes, vérifiez toujours ce fichier pour

voir et modifier au besoin le comportement par défaut du système.

La commande `passwd`

Cette commande est utilisée essentiellement pour modifier le mot de passe d'un utilisateur. Comme nous l'avons déjà dit, chaque utilisateur peut modifier son propre mot de passe, mais seul root a le droit de modifier le mot de passe de *n'importe quel* utilisateur. Ceci est dû au fait que la commande `passwd` a le bit SUID activé (un `s` à la place du fanion exécutable pour le propriétaire), ce qui signifie qu'elle s'exécute avec les privilèges du propriétaire du fichier (en l'occurrence root).

```
# ls -l /usr/bin/passwd
-rwsr-xr-x 1 root root 42096 mag 17 2015 /usr/bin/passwd
```

En fonction des options utilisées avec `passwd`, vous pouvez contrôler des aspects spécifiques à l'expiration des mots de passe :

-d

Supprimer le mot de passe d'un compte utilisateur (en le désactivant ainsi).

-e

Obliger l'utilisateur à modifier son mot de passe.

-i

Définir le nombre de jours d'inactivité après l'expiration d'un mot de passe pendant lesquels l'utilisateur devra mettre à jour son mot de passe (faute de quoi le compte sera désactivé).

-l

Verrouiller le compte utilisateur (le mot de passe chiffré est préfixé d'un point d'exclamation dans le fichier `/etc/shadow`).

-n

Définir la durée de vie minimale du mot de passe.

-S

Afficher les informations sur le statut du mot de passe d'un compte utilisateur donné.

-u

Déverrouiller le compte utilisateur (le point d'exclamation est supprimé du champ du mot de passe dans le fichier `/etc/shadow`).

-X

Définir la durée de vie maximale du mot de passe.

-W

Définir le nombre de jours d'avertissement avant l'expiration du mot de passe pendant lesquels l'utilisateur sera averti que le mot de passe devra être modifié.

NOTE

Les groupes peuvent également avoir un mot de passe, qui est défini à l'aide de la commande `gpasswd`. Les utilisateurs qui ne sont pas membres d'un groupe mais qui connaissent son mot de passe peuvent le rejoindre temporairement en utilisant la commande `newgrp`. Rappelez-vous que `gpasswd` est également utilisé pour ajouter et supprimer des utilisateurs d'un groupe et pour définir la liste des administrateurs et des membres ordinaires du groupe.

La commande `chage`

Cette commande, qui signifie changer l'âge (*change age*), est utilisée pour modifier les informations d'expiration du mot de passe d'un utilisateur. La commande `chage` est réservée à l'utilisateur `root`, à l'exception de l'option `-l` qui peut être utilisée par les utilisateurs ordinaires pour obtenir la liste des informations d'expiration du mot de passe de leur propre compte.

Voici les autres options qui s'appliquent à la commande `chage` :

-d

Définir le dernier changement de mot de passe pour un compte utilisateur.

-E

Définir la date d'expiration d'un compte utilisateur.

-I

Définir le nombre de jours d'inactivité après l'expiration d'un mot de passe pendant lesquels l'utilisateur devra mettre à jour son mot de passe (faute de quoi le compte sera désactivé).

-m

Définir la durée de vie minimale du mot de passe pour un compte utilisateur.

-M

Définir la durée de vie maximale du mot de passe pour un compte utilisateur.

-W

Définir le nombre de jours d'avertissement avant l'expiration du mot de passe pendant lesquels

l'utilisateur sera averti que le mot de passe devra être modifié.

Exercices guidés

1. Pour chacune des commandes ci-dessous, indiquez l'objectif correspondant :

<code>usermod -L</code>	
<code>passwd -u</code>	
<code>chage -E</code>	
<code>groupdel</code>	
<code>useradd -s</code>	
<code>groupadd -g</code>	
<code>userdel -r</code>	
<code>usermod -l</code>	
<code>groupmod -n</code>	
<code>useradd -m</code>	

2. Pour chacune des commandes `passwd` ci-dessous, identifiez la commande `chage` correspondante :

<code>passwd -n</code>	
<code>passwd -x</code>	
<code>passwd -w</code>	
<code>passwd -i</code>	
<code>passwd -S</code>	

3. Expliquez en détail la fonction des commandes de la question précédente :

4. Quelles commandes pouvez-vous utiliser pour verrouiller un compte utilisateur ? Et quelles commandes pour le déverrouiller ?

Exercices d'approfondissement

1. En utilisant la commande `groupadd`, créez les groupes `administrators` et `developers`. Admettons que vous travaillez en tant que `root`.

2. Maintenant que vous avez créé ces groupes, exécutez la commande suivante : `useradd -G administrators,developers kevin`. Quelles sont les actions effectuées par cette commande ? Admettons que `CREATE_HOME` et `USERGROUPS_ENAB` dans `/etc/login.defs` sont mis sur `yes`.

3. Créez un nouveau groupe nommé `designers`, renommez-le en `web-designers` et ajoutez ce nouveau groupe aux groupes secondaires du compte utilisateur `kevin`. Identifiez tous les groupes auxquels `kevin` appartient ainsi que leurs identifiants.

4. Supprimez uniquement le groupe `developers` des groupes secondaires de `kevin`.

5. Définissez le mot de passe pour le compte utilisateur `kevin`.

6. En utilisant la commande `chage`, vérifiez d'abord la date d'expiration du compte utilisateur `kevin` et fixez-la au 31 décembre 2022. Quelle autre commande pouvez-vous utiliser pour modifier la date d'expiration d'un compte utilisateur ?

7. Ajoutez un nouveau compte utilisateur nommé `emma` avec l'UID 1050 et définissez `administrators` comme groupe primaire et `developers` et `web-designers` comme groupes secondaires.

8. Changez le shell de connexion de `emma` en `/bin/sh`.

9. Supprimez les comptes utilisateurs `emma` et `kevin` ainsi que les groupes `administrators`, `developers` et `web-designers`.

Résumé

Voici ce que nous avons vu dans cette leçon :

- Les principes fondamentaux de la gestion des utilisateurs et des groupes sous Linux.
- Comment ajouter, modifier et supprimer des comptes utilisateurs.
- Comment ajouter, modifier et supprimer des groupes.
- Gérer le répertoire squelette.
- Éditer le fichier qui contrôle la création des utilisateurs et des groupes.
- Modifier les mots de passe des comptes utilisateurs.
- Modifier les informations relatives à l'expiration des mots de passe des comptes utilisateurs.

Voici les commandes et les fichiers qui ont été abordés dans cette leçon :

useradd

Créer un nouveau compte utilisateur.

usermod

Modifier un compte utilisateur.

userdel

Supprimer un compte utilisateur.

groupadd

Créer un nouveau groupe.

groupmod

Modifier un groupe.

groupdel

Supprimer un groupe.

passwd

Modifier les mots de passe des comptes utilisateurs et contrôler tous les aspects de l'expiration des mots de passe.

chage

Modifier les informations relatives à l'expiration du mot de passe de l'utilisateur.

/etc/skel

L'emplacement par défaut du répertoire squelette.

/etc/login.defs

Le fichier qui contrôle la création des utilisateurs et des groupes et qui fournit les valeurs par défaut pour plusieurs paramètres des comptes utilisateurs.

Réponses aux exercices guidés

1. Pour chacune des commandes ci-dessous, indiquez l'objectif correspondant :

<code>usermod -L</code>	Verrouiller le compte utilisateur
<code>passwd -u</code>	Déverrouiller le compte utilisateur
<code>chage -E</code>	Définir la date d'expiration du compte utilisateur
<code>groupdel</code>	Supprimer le groupe
<code>useradd -s</code>	Créer un nouveau compte utilisateur avec un shell de connexion spécifique
<code>groupadd -g</code>	Créer un nouveau groupe avec un GID spécifique
<code>userdel -r</code>	Supprimer le compte utilisateur et tous les fichiers de son répertoire personnel, le répertoire personnel lui-même et le spool de courrier de l'utilisateur
<code>usermod -l</code>	Modifier le nom d'utilisateur du compte
<code>groupmod -n</code>	Modifier le nom du groupe
<code>useradd -m</code>	Créer un nouveau compte utilisateur et son répertoire personnel

2. Pour chacune des commandes `passwd` ci-dessous, identifiez la commande `chage` correspondante :

<code>passwd -n</code>	<code>chage -m</code>
<code>passwd -x</code>	<code>chage -M</code>
<code>passwd -w</code>	<code>chage -W</code>
<code>passwd -i</code>	<code>chage -I</code>
<code>passwd -S</code>	<code>chage -l</code>

3. Expliquez en détail la fonction des commandes de la question précédente :

Sous Linux, vous pouvez utiliser la commande `passwd -n` (ou `chage -m`) pour définir le nombre minimum de jours entre deux changements de mot de passe, la commande `passwd -x`

(ou `chage -M`) pour définir le nombre maximum de jours pendant lesquels un mot de passe est valide, la commande `passwd -w` (ou `chage -W`) pour définir le nombre de jours d'avertissement avant que le mot de passe n'expire, la commande `passwd -i` (ou `chage -I`) pour définir le nombre de jours d'inactivité pendant lesquels l'utilisateur doit changer son mot de passe et la commande `passwd -S` (ou `chage -l`) pour afficher des informations succinctes sur le mot de passe du compte utilisateur.

4. Quelles commandes pouvez-vous utiliser pour verrouiller un compte utilisateur ? Et quelles commandes pour le déverrouiller ?

Si vous voulez verrouiller un compte utilisateur, vous pouvez utiliser l'une de ces commandes : `usermod -L`, `usermod --lock` et `passwd -l`. À l'inverse, si vous souhaitez le déverrouiller, vous pouvez utiliser `usermod -U`, `usermod --unlock` et `passwd -u`.

Réponses aux exercices d'approfondissement

1. En utilisant la commande `groupadd`, créez les groupes `administrators` et `developers`. Admettons que vous travaillez en tant que `root`.

```
# groupadd administrators
# groupadd developers
```

2. Maintenant que vous avez créé ces groupes, exécutez la commande suivante : `useradd -G administrators,developers kevin`. Quelles sont les actions effectuées par cette commande ? Admettons que `CREATE_HOME` et `USERGROUPS_ENAB` dans `/etc/login.defs` sont mis sur `yes`.

La commande ajoute un nouvel utilisateur nommé `kevin` à la liste des utilisateurs du système, crée son répertoire personnel (`CREATE_HOME` est réglé sur `yes` et vous pouvez donc omettre l'option `-m`) ainsi qu'un nouveau groupe nommé `kevin` comme groupe primaire de ce compte utilisateur (`USERGROUPS_ENAB` est réglé sur `yes`). Enfin, les fichiers et dossiers contenus dans le répertoire squelette sont copiés vers le répertoire personnel de `kevin`.

3. Créez un nouveau groupe nommé `designers`, renommez-le en `web-designers` et ajoutez ce nouveau groupe aux groupes secondaires du compte utilisateur `kevin`. Identifiez tous les groupes auxquels `kevin` appartient ainsi que leurs identifiants.

```
# groupadd designers
# groupmod -n web-designers designers
# usermod -a -G web-designers kevin
# id kevin
uid=1010(kevin) gid=1030(kevin)
groups=1030(kevin),1028(administrators),1029(developers),1031(web-designers)
```

4. Supprimez uniquement le groupe `developers` des groupes secondaires de `kevin`.

```
# usermod -G administrators,web-designers kevin
# id kevin
uid=1010(kevin) gid=1030(kevin) groups=1030(kevin),1028(administrators),1031(web-designers)
```

La commande `usermod` n'a pas d'option pour supprimer un seul groupe ; par conséquent, vous devez spécifier tous les groupes secondaires auxquels l'utilisateur appartient.

5. Définissez le mot de passe pour le compte utilisateur kevin.

```
# passwd kevin
Changing password for user kevin.
New UNIX password:
Retype new UNIX password:
passwd: all authentication tokens updated successfully.
```

6. En utilisant la commande `chage`, vérifiez d'abord la date d'expiration du compte utilisateur kevin et fixez-la au 31 décembre 2022. Quelle autre commande pouvez-vous utiliser pour modifier la date d'expiration d'un compte utilisateur ?

```
# chage -l kevin | grep "Account expires"
Account expires : never
# chage -E 2022-12-31 kevin
# chage -l kevin | grep "Account expires"
Account expires : dec 31, 2022
```

La commande `usermod` avec l'option `-e` équivaut à `chage -E`.

7. Ajoutez un nouveau compte utilisateur nommé emma avec l'UID 1050 et définissez `administrators` comme groupe primaire et `developers` et `web-designers` comme groupes secondaires.

```
# useradd -u 1050 -g administrators -G developers,web-designers emma
# id emma
uid=1050(emma) gid=1028(administrators)
groups=1028(administrators),1029(developers),1031(web-designers)
```

8. Changez le shell de connexion de emma en `/bin/sh`.

```
# usermod -s /bin/sh emma
```

9. Supprimez les comptes utilisateurs emma et kevin ainsi que les groupes administrators, developers et web-designers.

```
# userdel -r emma
# userdel -r kevin
# groupdel administrators
```

```
# groupdel developers  
# groupdel web-designers
```



107.1 Leçon 2

Certification :	LPIC-1
Version :	5.0
Thème :	107 Tâches administratives
Objectif :	107.1 Gérer les comptes utilisateurs et les groupes ainsi que les fichiers système correspondants
Leçon :	2 sur 2

Introduction

Les outils en ligne de commande présentés dans la leçon précédente ainsi que les applications graphiques fournies par chaque distribution qui effectuent les mêmes tâches mettent à jour une série de fichiers qui stockent les informations relatives aux utilisateurs et aux groupes.

Ces fichiers sont situés dans le répertoire `/etc/`. Les voici :

`/etc/passwd`

Un fichier de sept champs délimités par deux points contenant des informations de base sur les utilisateurs.

`/etc/group`

Un fichier de quatre champs délimités par deux points contenant des informations de base sur les groupes.

/etc/shadow

Un fichier de neuf champs délimités par deux points contenant les mots de passe chiffrés des utilisateurs.

/etc/gshadow

Un fichier de quatre champs délimités par deux points contenant les mots de passe chiffrés des groupes.

Même si ces quatre fichiers sont au format texte simple, ils ne doivent pas être édités directement, mais toujours à l'aide des outils fournis par votre distribution.

/etc/passwd

Ce fichier lisible par tous contient une liste d'utilisateurs, chacun sur une ligne distincte. Chaque ligne est constituée de sept champs délimités par deux points :

Nom d'utilisateur

Le nom utilisé lorsque l'utilisateur se connecte au système.

Mot de passe

Le mot de passe chiffré (ou un `x` si l'on utilise des mots de passe *shadow*).

ID utilisateur (UID)

Le numéro d'identification attribué à l'utilisateur dans le système.

ID du groupe (GID)

L'identifiant du groupe primaire de l'utilisateur dans le système.

GECOS

Un champ de commentaire facultatif, utilisé pour ajouter des informations supplémentaires sur l'utilisateur (comme son nom complet). Le champ peut contenir plusieurs entrées séparées par des virgules.

Répertoire personnel

Le chemin absolu vers le répertoire personnel de l'utilisateur.

Shell

Le chemin absolu vers le programme lancé automatiquement lorsque l'utilisateur se connecte au système (le plus souvent un shell interactif comme `/bin/bash`).

/etc/group

Ce fichier lisible par tous contient une liste de groupes, chacun sur une ligne distincte. Chaque ligne est constituée de quatre champs délimités par deux points :

Nom du groupe

Le nom du groupe.

Mot de passe du groupe

Le mot de passe chiffré du groupe (ou un `x` si l'on utilise des mots de passe *shadow*).

ID du groupe (GID)

Le numéro d'identification attribué au groupe dans le système.

Liste des membres

Une liste délimitée par des virgules de tous les utilisateurs appartenant au groupe, à l'exception de ceux dont c'est le groupe principal.

/etc/shadow

Ce fichier ne peut être lu que par root et par les utilisateurs disposant des privilèges de root. Il contient les mots de passe chiffrés des utilisateurs, chacun sur une ligne distincte. Chaque ligne est constituée de neuf champs délimités par deux points :

Nom d'utilisateur

Le nom utilisé lorsque l'utilisateur se connecte au système.

Mot de passe chiffré

Le mot de passe chiffré de l'utilisateur (si la valeur commence par `!`, le compte est verrouillé).

Date du dernier changement de mot de passe

La date du dernier changement de mot de passe, en nombre de jours depuis le 1er janvier 1970 (une valeur de 0 signifie que l'utilisateur devra changer son mot de passe lors de la prochaine connexion).

Âge minimum du mot de passe

Le nombre minimum de jours qui devront s'écouler après un changement de mot de passe avant que l'utilisateur ne soit autorisé à modifier à nouveau son mot de passe.

Âge maximum du mot de passe

Le nombre maximal de jours qui peuvent s'écouler avant qu'un changement de mot de passe ne soit nécessaire.

Délai d'avertissement du mot de passe

Le nombre de jours avant l'expiration du mot de passe pendant lesquels l'utilisateur est averti que celui-ci devra être modifié.

Période d'inactivité du mot de passe

Le nombre de jours après l'expiration du mot de passe pendant lesquels l'utilisateur devra mettre à jour son mot de passe. Au terme de cette période, le compte sera désactivé si l'utilisateur ne modifie pas son mot de passe.

Date d'expiration du compte

La date, exprimée en nombre de jours depuis le 1er janvier 1970, à laquelle le compte utilisateur sera désactivé (un champ vide signifie que le compte utilisateur n'expirera jamais).

Un champ réservé

Ce champ est réservé pour une utilisation future.

/etc/gshadow

Ce fichier ne peut être lu que par root et par les utilisateurs disposant des privilèges de root. Il contient les mots de passe chiffrés des groupes, chacun sur une ligne distincte. Chaque ligne est constituée de quatre champs délimités par deux points.

Nom du groupe

Le nom du groupe.

Mot de passe chiffré

Le mot de passe chiffré du groupe (utilisé lorsqu'un utilisateur qui n'est pas membre du groupe souhaite rejoindre le groupe en utilisant la commande `newgrp` — si le mot de passe commence par `!`, personne n'est autorisé à accéder au groupe avec `newgrp`).

Administrateurs du groupe

La liste des administrateurs du groupe, délimitée par des virgules (ils peuvent changer le mot de passe du groupe et ajouter ou supprimer des membres du groupe avec la commande `gpasswd`).

Membres du groupe

La liste des membres du groupe, délimitée par des virgules.

Filtrer les bases de données de mots de passe et des groupes

Il peut être nécessaire assez fréquemment d'examiner les informations sur les utilisateurs et les groupes stockées dans ces quatre fichiers pour rechercher des enregistrements spécifiques. Pour effectuer cette tâche, vous pouvez utiliser la commande `grep` ou concaténer `cat` et `grep`.

```
# grep emma /etc/passwd
emma:x:1020:1020:User Emma:/home/emma:/bin/bash
# cat /etc/group | grep db-admin
db-admin:x:1050:grace,frank
```

La commande `getent` constitue un autre moyen pour consulter ces bases de données. En général, cette commande affiche les entrées des bases de données supportées par les bibliothèques *Name Service Switch* (NSS) et requiert le nom de la base et une clé de recherche. En l'absence d'argument clé, toutes les entrées de la base de données spécifiée sont affichées (à moins que la base de données ne supporte pas l'énumération). Dans le cas contraire, si une ou plusieurs clés sont fournies, la base de données sera filtrée en conséquence.

```
# getent passwd emma
emma:x:1020:1020:User Emma:/home/emma:/bin/bash
# getent group db-admin
db-admin:x:1050:grace,frank
```

La commande `getent` ne requiert pas les privilèges de root ; vous devez juste être capable de lire la base de données à partir de laquelle vous souhaitez récupérer des enregistrements.

NOTE

Rappelez-vous que `getent` ne peut accéder qu'aux bases de données configurées dans le fichier `/etc/nsswitch.conf`.

Exercices guidés

1. Examinez l’affichage suivant et répondez aux questions ci-dessous :

```
# cat /etc/passwd | grep '\(root\|mail\|catherine\|kevin\)'
root:x:0:0:root:/root:/bin/bash
mail:x:8:8:mail:/var/spool/mail:/sbin/nologin
catherine:x:1030:1025:User Chaterine:/home/catherine:/bin/bash
kevin:x:1040:1015:User Kevin:/home/kevin:/bin/bash
# cat /etc/group | grep '\(root\|mail\|db-admin\|app-developer\)'
root:x:0:
mail:x:8:
db-admin:x:1015:emma,grace
app-developer:x:1016:catherine,dave,christian
# cat /etc/shadow | grep '\(root\|mail\|catherine\|kevin\)'
root:$6$1u36Ipok$1jt8ooPMLewAhkQPf.1YgGopAB.jClT06ljscdzxvkLPkpi/amgp.zyfAN680zrLLp2avvpd
KA0llpssdfcPppOp:18015:0:99999:7:::
mail:!:18015:0:99999:7:::
catherine:$6$ABCD25jlld14hpPthEFGnssEWw1234yioMpliABCdef1f3478kAfhhAfgbAMjY1/BAeeAsl/FeE
dddKd12345g6kPACcik:18015:20:90:5:::
kevin:$6$DEFGabc123WrLp223fsvp0ddx3dbA7pPPc4LMaa123u6Lp02Lpvm123456pyphhh5ps012vbArL245.P
R1345kkA3Gas12P:18015:0:60:7:2:::
# cat /etc/gshadow | grep '\(root\|mail\|db-admin\|app-developer\)'
root:*:
mail:*:
db-admin:!:emma:emma,grace
app-developer:!:catherine,dave,christian
```

- Quel est l’ID utilisateur (UID) et l’ID de groupe (GID) de root et de catherine ?
- Quel est le nom du groupe primaire de kevin ? Y a-t-il d’autres membres dans ce groupe ?
- Quel shell est défini pour mail ? Qu’est-ce que cela signifie ?
- Qui fait partie du groupe app-developer ? Quels sont les administrateurs du groupe et quels sont les membres ordinaires ?

- Quelle est la durée de vie minimale du mot de passe pour `catherine` ? Et quelle est la durée de vie maximale du mot de passe ?

- Quelle est la période d'inactivité du mot de passe pour `kevin` ?

2. Par convention, quels sont les ID attribués aux comptes système et quels sont ceux attribués aux utilisateurs classiques ?

3. Comment savoir si un compte utilisateur qui était auparavant en mesure d'accéder au système se retrouve verrouillé ? Partons du principe que votre système utilise des mots de passe *shadow*.

Exercices d'approfondissement

1. Créez un compte utilisateur nommé `christian` à l'aide de la commande `useradd -m` et identifiez son ID utilisateur (UID), son ID de groupe (GID) et son shell.

2. Identifiez le nom du groupe primaire de `christian`. Que pouvez-vous en déduire ?

3. En utilisant la commande `getent`, vérifiez les informations de conservation du mot de passe pour le compte utilisateur `christian`.

4. Ajoutez le groupe `editor` aux groupes secondaires de `christian`. Supposons que ce groupe contienne déjà `emma`, `dave` et `frank` comme membres ordinaires. Comment pouvez-vous vérifier qu'il n'y a pas d'administrateurs pour ce groupe ?

5. Lancez la commande `ls -l /etc/passwd /etc/group /etc/shadow /etc/gshadow` et décrivez le résultat en termes de permissions de fichiers. Parmi ces quatre fichiers, lesquels sont protégés pour des raisons de sécurité ? Partez du principe que votre système utilise des mots de passe *shadow*.

Résumé

Voici ce que nous avons appris dans cette leçon :

- L'emplacement des fichiers qui enregistrent les informations sur les utilisateurs et les groupes.
- Gérer les informations relatives aux utilisateurs et aux groupes enregistrées dans les bases de données des mots de passe et des groupes.
- Récupérer des informations dans les bases de données des mots de passe et des groupes.

Voici les commandes et les fichiers que nous avons abordés dans cette leçon :

/etc/passwd

Le fichier qui contient les informations de base sur les utilisateurs.

/etc/group

Le fichier qui contient les informations de base sur les groupes.

/etc/shadow

Le fichier qui contient les mots de passe chiffrés des utilisateurs.

/etc/gshadow

Le fichier qui contient les mots de passe chiffrés des groupes.

getent

Filtrer les bases de données des mots de passe et des groupes.

Réponses aux exercices guidés

1. Examinez l’affichage suivant et répondez aux questions ci-dessous :

```
# cat /etc/passwd | grep '\(root\|mail\|catherine\|kevin\)'
root:x:0:0:root:/root:/bin/bash
mail:x:8:8:mail:/var/spool/mail:/sbin/nologin
catherine:x:1030:1025:User Chaterine:/home/catherine:/bin/bash
kevin:x:1040:1015:User Kevin:/home/kevin:/bin/bash
# cat /etc/group | grep '\(root\|mail\|db-admin\|app-developer\)'
root:x:0:
mail:x:8:
db-admin:x:1015:emma,grace
app-developer:x:1016:catherine,dave,christian
# cat /etc/shadow | grep '\(root\|mail\|catherine\|kevin\)'
root:$6$1u36Ipok$1jt8ooPMLewAhkQPf.1YgGopAB.jClT06ljscdzxvkLPkpi/amgp.zyfAN680zrLLp2avvpd
KA0llpssdfcPppOp:18015:0:99999:7:::
mail:!:18015:0:99999:7:::
catherine:$6$ABCD25jlld14hpPthEFGnssEWw1234yioMpliABCdef1f3478kAfhhAfgbAMjY1/BAeeAsl/FeE
dddKd12345g6kPACcik:18015:20:90:5:::
kevin:$6$DEFGabc123WrLp223fsvp0ddx3dbA7pPPc4LMaa123u6Lp02Lpvm123456pyphhh5ps012vbArL245.P
R1345kkA3Gas12P:18015:0:60:7:2:::
# cat /etc/gshadow | grep '\(root\|mail\|db-admin\|app-developer\)'
root:::
mail:::
db-admin:!:emma:emma,grace
app-developer:!:catherine,dave,christian
```

- Quel est l’ID utilisateur (UID) et l’ID de groupe (GID) de root et de catherine ?

L’UID et le GID de root sont 0 et 0, tandis que l’UID et le GID de catherine sont 1030 et 1025.

- Quel est le nom du groupe primaire de kevin ? Y a-t-il d’autres membres dans ce groupe ?

Le nom du groupe est db-admin. emma et grace font également partie de ce groupe.

- Quel shell est défini pour mail ? Qu’est-ce que cela signifie ?

mail est un compte système et son shell est /sbin/nologin. En fait, les comptes système tels que mail, ftp, news et daemon sont utilisés pour effectuer des tâches administratives et par conséquent, une connexion normale devrait être impossible pour ces comptes. C’est

pourquoi le shell est habituellement paramétré à `/sbin/nologin` ou `/bin/false`.

- Qui fait partie du groupe `app-developer` ? Quels sont les administrateurs du groupe et quels sont les membres ordinaires ?

Les membres sont `catherine`, `dave` et `christian` et ils sont tous des membres ordinaires.

- Quelle est la durée de vie minimale du mot de passe pour `catherine` ? Et quelle est la durée de vie maximale du mot de passe ?

La durée de vie minimale du mot de passe est de 20 jours, tandis que la durée de vie maximale est de 90 jours.

- Quelle est la période d'inactivité du mot de passe pour `kevin` ?

La période d'inactivité du mot de passe est de deux jours. Pendant cette période, `kevin` devra mettre à jour le mot de passe, faute de quoi le compte sera désactivé.

2. Par convention, quels sont les ID attribués aux comptes système et quels sont ceux attribués aux utilisateurs classiques ?

Les comptes système ont généralement des UID inférieurs à 100 ou compris entre 500 et 1000, tandis que les utilisateurs normaux ont des UID à partir de 1000, même si certains systèmes historiques commencent la numérotation à 500. L'utilisateur `root` a un UID de 0. Rappelez-vous que les valeurs `UID_MIN` et `UID_MAX` dans `/etc/login.defs` définissent la plage d'UIDs utilisée pour la création d'utilisateurs ordinaires. Du point de vue de LPI Linux Essentials et LPIC-1, les comptes système ont des UIDs inférieurs à 1000 et les utilisateurs ordinaires ont des UIDs supérieurs à 1000.

3. Comment savoir si un compte utilisateur qui était auparavant en mesure d'accéder au système se retrouve verrouillé ? Partons du principe que votre système utilise des mots de passe *shadow*.

Lorsque les mots de passe *shadow* sont utilisés, le second champ de `/etc/passwd` contient le caractère `x` pour chaque compte utilisateur, puisque les mots de passe utilisateur chiffrés sont stockés dans `/etc/shadow`. Plus précisément, le mot de passe chiffré d'un compte utilisateur est enregistré dans le deuxième champ de ce fichier et il est verrouillé s'il commence par un point d'exclamation.

Réponses aux exercices d'approfondissement

1. Créez un compte utilisateur nommé `christian` à l'aide de la commande `useradd -m` et identifiez son ID utilisateur (UID), son ID de groupe (GID) et son shell.

```
# useradd -m christian
# cat /etc/passwd | grep christian
christian:x:1050:1060::/home/christian:/bin/bash
```

L'UID et le GID de `christian` sont respectivement 1050 et 1060 (le troisième et le quatrième champ de `/etc/passwd`). `/bin/bash` est le shell défini pour ce compte utilisateur (le septième champ de `/etc/passwd`).

2. Identifiez le nom du groupe primaire de `christian`. Que pouvez-vous en déduire ?

```
# cat /etc/group | grep 1060
christian:x:1060:
```

Le nom du groupe primaire de `christian` est `christian` (le premier champ dans `/etc/group`). Par conséquent, `USERGROUPS_ENAB` dans `/etc/login.defs` est défini à `yes` pour que `useradd` crée par défaut un groupe avec le même nom que le compte utilisateur.

3. En utilisant la commande `getent`, vérifiez les informations de conservation du mot de passe pour le compte utilisateur `christian`.

```
# getent shadow christian
christian:!:18015:0:99999:7:::
```

Le compte utilisateur `christian` n'a pas de mot de passe défini et il est maintenant verrouillé (le second champ de `/etc/shadow` contient un point d'exclamation). Il n'y a pas d'âge minimum ou maximum pour le mot de passe de ce compte utilisateur (le quatrième et le cinquième champ de `/etc/shadow` sont définis à 0 et 99999 jours), alors que la période d'avertissement du mot de passe est fixée à 7 jours (le sixième champ de `/etc/shadow`). Enfin, il n'y a pas de période d'inactivité (septième champ de `/etc/shadow`) et le compte n'expire jamais (huitième champ de `/etc/shadow`).

4. Ajoutez le groupe `editor` aux groupes secondaires de `christian`. Supposons que ce groupe contienne déjà `emma`, `dave` et `frank` comme membres ordinaires. Comment pouvez-vous vérifier qu'il n'y a pas d'administrateurs pour ce groupe ?

```
# cat /etc/group | grep editor
editor:x:1100:emma,dave,frank
# usermod -a -G editor christian
# cat /etc/group | grep editor
editor:x:1100:emma,dave,frank,christian
# cat /etc/gshadow | grep editor
editor:::emma,dave,frank,christian
```

Le troisième et le quatrième champ de `/etc/gshadow` contiennent les administrateurs et les membres ordinaires du groupe spécifié. Ainsi, puisque le troisième champ est vide pour `editor`, il n'y a pas d'administrateurs pour ce groupe (`emma`, `dave`, `frank` et `christian` sont tous des membres ordinaires).

5. Lancez la commande `ls -l /etc/passwd /etc/group /etc/shadow /etc/gshadow` et décrivez le résultat en termes de permissions de fichiers. Parmi ces quatre fichiers, lesquels sont protégés pour des raisons de sécurité ? Partez du principe que votre système utilise des mots de passe *shadow*.

```
# ls -l /etc/passwd /etc/group /etc/shadow /etc/gshadow
-rw-r--r-- 1 root root 853 mag 1 08:00 /etc/group
-rw-r----- 1 root shadow 1203 mag 1 08:00 /etc/gshadow
-rw-r--r-- 1 root root 1354 mag 1 08:00 /etc/passwd
-rw-r----- 1 root shadow 1563 mag 1 08:00 /etc/shadow
```

Les fichiers `/etc/passwd` et `/etc/group` sont lisibles par tous et font l'objet d'un *shadow* pour des raisons de sécurité. Lorsque les mots de passe *shadow* sont utilisés, vous pouvez voir un `x` dans le second champ de ces fichiers, étant donné que les mots de passe chiffrés pour les utilisateurs et les groupes sont enregistrés dans `/etc/shadow` et `/etc/gshadow`, qui ne sont lisibles que par `root` et, dans mon système, par les membres du groupe `shadow`.



107.2 Automate system administration tasks by scheduling jobs

Référence aux objectifs de LPI

LPIC-1 version 5.0, Exam 102, Objective 107.2

Valeur

4

Domaines de connaissance les plus importants

- Gestion des tâches cron et at.
- Configuration des accès aux services cron et atd.
- Compréhension des unités d'horloge systemd (systemd timer units).

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

- `/etc/cron.{d,daily,hourly,monthly,weekly}/`
- `/etc/at.deny`
- `/etc/at.allow`
- `/etc/crontab`
- `/etc/cron.allow`
- `/etc/cron.deny`
- `/var/spool/cron/`
- `crontab`
- `at`
- `atq`
- `atrm`

- `systemctl`
- `systemd-run`



107.2 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	107 Tâches administratives
Objectif :	107.2 Automatisation des tâches d'administration par la planification des travaux
Leçon :	1 sur 2

Introduction

Une des missions les plus essentielles d'un administrateur système digne de ce nom consiste à organiser les tâches qui doivent être exécutées de manière régulière. À titre d'exemple, un administrateur peut créer et automatiser des tâches pour les sauvegardes et les mises à jour du système et pour effectuer toutes sortes d'autres activités répétitives. Pour ce faire, vous pouvez utiliser la fonction `cron`, qui permet d'automatiser la planification des tâches périodiques.

Planifier des tâches avec Cron

Sous Linux, `cron` est un démon qui fonctionne en continu et qui se réveille toutes les minutes pour interroger un ensemble de tables dans le but d'y trouver des tâches à exécuter. Ces tables sont connues sous le nom de *crontabs* et contiennent ce que l'on appelle les *cron jobs*. Cron convient aux serveurs et aux systèmes qui sont constamment allumés dans la mesure où chaque tâche cron n'est exécutée que si le système fonctionne à l'heure prévue. Il peut être utilisé par des utilisateurs lambda qui disposent chacun de leur propre *crontab* ainsi que par l'utilisateur root qui gère les crontabs du système.

NOTE

Sous Linux, il existe également l'outil `anacron` adapté aux systèmes qui peuvent être éteints (comme les ordinateurs de bureau ou les ordinateurs portables). Il ne peut être utilisé que par root. Si la machine est éteinte lorsque les tâches `anacron` doivent être exécutées, elles le seront lors de la prochaine mise sous tension de la machine. `anacron` ne fait pas partie du champ d'application de la certification LPIC-1.

Crontabs utilisateur

Les *crontabs utilisateur* sont des fichiers texte qui gèrent l'ordonnancement des tâches cron définies par les utilisateurs. Ils sont toujours nommés en fonction du compte utilisateur qui les a créés, mais l'emplacement exact de ces fichiers dépend de la distribution utilisée (généralement un sous-répertoire de `/var/spool/cron`).

Chaque ligne d'une crontab utilisateur contient six champs séparés par un espace :

- La minute de l'heure (0-59).
- L'heure du jour (0-23).
- Le jour du mois (1-31).
- Le mois de l'année (1-12).
- Le jour de la semaine (0-7 avec dimanche=0 ou dimanche=7).
- La commande à exécuter.

Pour le mois de l'année et le jour de la semaine, vous pouvez utiliser les trois premières lettres du nom au lieu du chiffre correspondant.

Les cinq premiers champs indiquent à quel moment il faut exécuter la commande spécifiée dans le sixième champ. Ils peuvent contenir une ou plusieurs valeurs. En particulier, vous pouvez spécifier plusieurs valeurs en utilisant :

*** (astérisque)**

Désigne toute valeur.

, (virgule)

Désigne une liste de valeurs possibles.

- (tiret)

Désigne une fourchette de valeurs possibles.

/ (barre oblique)

Désigne des valeurs par paliers.

La plupart des distributions fournissent le fichier `/etc/crontab` qui peut être utilisé comme référence pour la syntaxe d'un fichier `cron`. Voici un exemple de fichier `/etc/crontab` d'une installation Debian :

```
SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# Example of job definition:
# .----- minute (0 - 59)
# | .----- hour (0 - 23)
# | | .----- day of month (1 - 31)
# | | | .----- month (1 - 12) OR jan,feb,mar,apr ...
# | | | | .---- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
# | | | | |
# * * * * * user-name command to be executed
```

Crontabs système

Les *crontabs système* sont des fichiers texte qui gèrent l'ordonnancement des tâches `cron` du système et qui ne peuvent être édités que par l'utilisateur `root`. `/etc/crontab` et tous les fichiers du répertoire `/etc/cron.d` sont des *crontabs système*.

La plupart des distributions fournissent également les répertoires `/etc/cron.hourly`, `/etc/cron.daily`, `/etc/cron.weekly` et `/etc/cron.monthly` qui contiennent les scripts à exécuter à la fréquence correspondante. Par exemple, si vous voulez exécuter un script une fois par jour, vous pouvez le placer dans `/etc/cron.daily`.

WARNING

Certaines distributions utilisent `/etc/cron.d/hourly`, `/etc/cron.d/daily`, `/etc/cron.d/weekly` et `/etc/cron.d/monthly`. N'oubliez pas de bien vérifier les répertoires dans lesquels vous rangez les scripts que vous voulez faire exécuter par `cron`.

La syntaxe des *crontabs système* ressemble à celle des *crontabs utilisateur*, avec en plus un champ obligatoire qui définit l'utilisateur censé exécuter la tâche `cron`. Par conséquent, chaque ligne d'une *crontab système* contient sept champs séparés par un espace :

- La minute de l'heure (0-59).
- L'heure du jour (0-23).

- Le jour du mois (1-31).
- Le mois de l'année (1-12).
- Le jour de la semaine (0-7 avec dimanche=0 ou dimanche=7).
- Le nom du compte utilisateur censé exécuter la commande.
- La commande à exécuter.

Comme pour les crontabs utilisateurs, vous pouvez spécifier plusieurs valeurs pour les champs horaires en utilisant les opérateurs `*`, `,`, `-` et `/`. Vous pouvez également indiquer le mois de l'année et le jour de la semaine en utilisant les trois premières lettres du nom au lieu du nombre correspondant.

Spécifications horaires particulières

Lorsque vous éditez des fichiers crontab, vous pouvez également utiliser une série de raccourcis spécifiques dans les cinq premières colonnes en remplacement des spécifications horaires :

@reboot

Exécute la tâche spécifiée une fois après un redémarrage.

@hourly

Exécute la tâche spécifiée une fois par heure au début de l'heure.

@daily (ou @midnight)

Exécute la tâche spécifiée une fois par jour à minuit.

@weekly

Exécute la tâche spécifiée une fois par semaine, le dimanche à minuit.

@monthly

Exécute la tâche spécifiée une fois par mois à minuit le premier jour du mois.

@yearly (ou @annually)

Exécute la tâche spécifiée une fois par an, le 1er janvier à minuit.

Variables Crontab

Dans un fichier crontab, il peut y avoir des affectations de variables en amont de la planification des tâches. Voici les variables d'environnement couramment définies :

HOME

Le répertoire où `cron` invoque les commandes (par défaut le répertoire personnel de l'utilisateur).

MAILTO

Le nom de l'utilisateur ou l'adresse à laquelle la sortie standard et les messages d'erreur sont envoyés (par défaut, le propriétaire de la crontab). Plusieurs valeurs séparées par des virgules sont également autorisées et une valeur vide indique qu'aucun e-mail ne doit être envoyé.

PATH

Le chemin où les commandes pourront être trouvées.

SHELL

Le shell à utiliser (par défaut `/bin/sh`).

Créer des tâches Cron utilisateurs

La commande `crontab` est utilisée pour maintenir les fichiers `crontab` pour les utilisateurs individuels. Plus concrètement, vous pouvez lancer la commande `crontab -e` pour éditer votre propre fichier `crontab` ou pour en créer un s'il n'existe pas encore.

```
$ crontab -e
no crontab for frank - using an empty one

Select an editor. To change later, run 'select-editor'.
 1. /bin/ed
 2. /bin/nano      < ---- easiest
 3. /usr/bin/emacs24
 4. /usr/bin/vim.tiny

Choose 1-4 [2]:
```

Par défaut, la commande `crontab` lance l'éditeur spécifié par les variables d'environnement `VISUAL` ou `EDITOR` pour vous permettre d'éditer votre fichier `crontab` avec votre éditeur préféré. Certaines distributions, comme le montre l'exemple ci-dessus, vous permettent de sélectionner l'éditeur à partir d'une liste lors de la première exécution de `crontab`.

Si vous voulez exécuter le script `foo.sh` dans votre répertoire personnel tous les jours à 10:00, vous pouvez ajouter la ligne suivante à votre fichier `crontab` :

```
0 10 * * * /home/frank/foo.sh
```

Jetez un œil sur ces exemples de crontabs :

```
0,15,30,45 08 * * 2 /home/frank/bar.sh
30 20 1-15 1,6 1-5 /home/frank/foobar.sh
```

Dans la première ligne, le script `bar.sh` est exécuté tous les mardis à 08h00, à 08h15, à 08h30 et à 08h45. À la deuxième ligne, le script `foobar.sh` est exécuté à 20h30 du lundi au vendredi pendant les quinze premiers jours de janvier et de juin.

WARNING

Même si les fichiers crontab peuvent être édités manuellement, il vaut mieux utiliser la commande `crontab`. En règle générale, les permissions sur les fichiers crontab ne permettent de les éditer que par le biais de la commande `crontab`.

En dehors de l'option `-e` mentionnée ci-dessus, la commande `crontab` comporte d'autres options utiles :

-l

Affiche la crontab actuelle sur la sortie standard.

-r

Supprime la crontab actuelle.

-u

Spécifie le nom de l'utilisateur dont la crontab doit être modifiée. Cette option requiert les privilèges de l'utilisateur root et lui permet de modifier les fichiers crontab des utilisateurs.

Créer des tâches Cron système

Contrairement aux crontabs utilisateurs, les crontabs système sont mises à jour avec un éditeur : ce n'est donc pas la peine de lancer la commande `crontab` pour éditer `/etc/crontab` et les fichiers dans `/etc/cron.d`. Notez que lorsque vous éditez des crontabs système, vous devez indiquer le compte qui sera utilisé pour exécuter la tâche cron (généralement l'utilisateur root).

Par exemple, si vous voulez exécuter le script `barfoo.sh` situé dans le répertoire `/root` tous les jours à 01:30 du matin, vous pouvez ouvrir `/etc/crontab` avec votre éditeur préféré et ajouter la ligne suivante :


```
30 01 * * * root /root/barfoo.sh >>/root/output.log 2>>/root/error.log
```

Dans l'exemple ci-dessus, la sortie de la tâche est ajoutée à `/root/output.log`, tandis que les erreurs s'ajoutent à `/root/error.log`.

WARNING

À moins que la sortie ne soit redirigée vers un fichier comme dans l'exemple ci-dessus (ou que la variable `MAILTO` ne soit définie à une valeur vide), toute la sortie de la tâche cron sera envoyée à l'utilisateur par e-mail. Une pratique courante consiste à rediriger la sortie standard vers `/dev/null` (ou vers un fichier pour examen ultérieur si nécessaire) et à ne pas rediriger l'erreur standard. De cette manière, l'utilisateur sera immédiatement notifié de toute erreur par e-mail.

Configurer l'accès à la planification des tâches

Sous Linux, les fichiers `/etc/cron.allow` et `/etc/cron.deny` sont utilisés pour définir les restrictions de `crontab`. Plus précisément, on les utilise pour autoriser ou interdire la planification de tâches cron pour différents utilisateurs. Si `/etc/cron.allow` existe, seuls les utilisateurs non root répertoriés dans ce fichier ont le droit de planifier des tâches cron en utilisant la commande `crontab`. Si `/etc/cron.allow` n'existe pas mais que `/etc/cron.deny` existe, les utilisateurs non root listés dans ce fichier ne peuvent pas planifier de tâches cron en utilisant la commande `crontab` (dans ce cas, un `/etc/cron.deny` vide signifie que chaque utilisateur est autorisé à planifier des tâches cron avec `crontab`). Si aucun de ces fichiers n'existe, l'accès des utilisateurs à la planification des tâches cron dépend de la distribution utilisée.

NOTE

Les fichiers `/etc/cron.allow` et `/etc/cron.deny` contiennent une liste de noms d'utilisateurs, chacun sur une ligne séparée.

Une alternative à Cron

Si vous gérez le système et les services avec `systemd`, vous pouvez utiliser les *timers* comme alternative à `cron` pour planifier vos tâches. Les *timers* sont des fichiers d'unité `systemd` identifiés par le suffixe `.timer`, et pour chacun d'entre eux, il doit y avoir un fichier unité correspondant qui décrit l'unité à activer lorsque le *timer* est écoulé. Par défaut, un *timer* active un service portant le même nom, au suffixe près.

Un *timer* comprend une section `[Timer]` qui spécifie l'heure à laquelle les tâches planifiées doivent être exécutées. Plus précisément, vous pouvez utiliser l'option `OnCalendar=` pour définir des *timers temps réel* qui fonctionnent de la même manière que les tâches cron (ils sont basés sur des expressions d'événements calendaires). L'option `OnCalendar=` requiert la syntaxe suivante :

```
DayOfWeek Year-Month-Day Hour:Minute:Second
```

avec `DayOfWeek` qui est optionnel. Les opérateurs `*`, `/` et `,` ont la même signification que ceux utilisés pour les tâches cron, et vous pouvez utiliser `..` entre deux valeurs pour indiquer une plage contiguë. Pour la spécification de `DayOfWeek`, vous pouvez utiliser les trois premières lettres du nom ou le nom complet.

NOTE

Vous pouvez également définir des *timers monotoniques* qui s'activent après un certain temps écoulé à partir d'un point de départ donné (par exemple, lorsque la machine a été démarrée ou lorsque le *timer* lui-même a été activé).

Par exemple, si vous voulez lancer le service `/etc/systemd/system/foobar.service` à 05:30 le premier lundi de chaque mois, vous pouvez ajouter les lignes suivantes dans le fichier d'unité `/etc/systemd/system/foobar.timer` correspondant.

```
[Unit]
Description=Run the foobar service

[Timer]
OnCalendar=Mon *-*-1..7 05:30:00
Persistent=true

[Install]
WantedBy=timers.target
```

Une fois que vous avez créé le nouveau *timer*, vous pouvez l'activer et le démarrer en exécutant les commandes suivantes en tant que root :

```
# systemctl enable foobar.timer
# systemctl start foobar.timer
```

Vous pouvez modifier la fréquence de votre tâche planifiée en éditant la valeur `OnCalendar` puis en invoquant la commande `systemctl daemon-reload`.

Enfin, si vous voulez voir la liste des *timers* actifs triés par ordre chronologique, vous pouvez utiliser la commande `systemctl list-timers`. Vous pouvez ajouter l'option `--all` pour afficher également les unités de *timers* inactives.

NOTE

Rappelez-vous que les *timers* sont enregistrés dans le journal de `systemd` et que vous pouvez consulter les journaux des différentes unités en utilisant la commande

`journalctl`. Notez aussi que si vous travaillez en tant qu'utilisateur `lambda`, vous devez ajouter l'option `--user` aux commandes `systemctl` et `journalctl`.

En remplacement de la forme normalisée longue décrite ci-dessus, vous pouvez utiliser des expressions spécifiques qui décrivent les fréquences particulières de l'exécution des tâches :

hourly

Exécuter la tâche spécifiée une fois par heure au début de l'heure.

daily

Exécuter la tâche spécifiée une fois par jour à minuit.

weekly

Exécuter la tâche spécifiée une fois par semaine, le lundi à minuit.

monthly

Exécuter la tâche spécifiée une fois par mois à minuit le premier jour du mois.

yearly

Exécuter la tâche spécifiée une fois par an à minuit le 1er janvier.

Vous pouvez consulter les pages du manuel `systemd.timer(5)` pour la liste complète des spécifications de la date et de l'heure.

Exercices guidés

1. Pour chacun des raccourcis `crontab` suivants, indiquez la spécification de temps correspondante (les cinq premières colonnes d'un fichier `crontab`) :

@hourly	
@daily	
@weekly	
@monthly	
@annually	

2. Pour chacun des raccourcis `OnCalendar` suivants, indiquez la spécification temporelle correspondante (la forme normalisée longue) :

hourly	
daily	
weekly	
monthly	
yearly	

3. Expliquez la signification des spécifications temporelles suivantes pour un fichier `crontab` :

30 13 * * 1-5	
00 09-18 * * *	
30 08 1 1 *	
0,20,40 11 * * Sun	
00 09 10-20 1-3 *	
*/20 * * * *	

4. Expliquez la signification des spécifications temporelles suivantes utilisées dans l'option `OnCalendar` d'un fichier `timer` :

--* 08:30:00	
Sat,Sun *-*-* 05:00:00	

- -01 13:15,30,45:00	
Fri *-09..12-* 16:20:00	
Mon,Tue *-*-1,15 08:30:00	
--* *:00/05:00	

Exercices d'approfondissement

1. En partant du principe que vous êtes autorisé à planifier des tâches avec `cron` en tant qu'utilisateur ordinaire, quelle commande utiliseriez-vous pour créer votre propre fichier `crontab` ?

2. Créez une tâche planifiée simple qui exécute la commande `date` tous les vendredi à 01:00 de l'après-midi. Où pouvez-vous voir le résultat de cette tâche ?

3. Créez une autre tâche planifiée qui exécute le script `foobar.sh` toutes les minutes, en redirigeant la sortie vers le fichier `output.log` dans votre répertoire personnel de façon à ce que seule l'erreur standard vous soit envoyée par e-mail.

4. Jetez un œil à l'entrée `crontab` de la tâche planifiée nouvellement créée. Pourquoi ne pas préciser le chemin absolu du fichier dans lequel la sortie standard est sauvegardée ? Et pourquoi utiliser la commande `./foobar.sh` pour exécuter le script ?

5. Éditez l'entrée `crontab` précédente en supprimant la redirection de sortie et désactivez la première tâche `cron` que vous avez créée.

6. Comment faire pour envoyer le résultat ainsi que les erreurs de votre tâche planifiée au compte utilisateur `emma` par e-mail ? Et comment éviter d'envoyer la sortie standard et les erreurs par e-mail ?

7. Exécutez la commande `ls -l /usr/bin/crontab`. Quelle permission spéciale est activée et quelle est sa signification ?

Résumé

Voici ce que nous avons appris dans cette leçon :

- Utiliser `cron` pour exécuter des tâches à des intervalles réguliers.
- Gérer les tâches `cron`.
- Configurer l'accès utilisateur à la planification des tâches `cron`.
- Comprendre le rôle des unités *timer* de `systemd` en tant qu'alternative à `cron`.

Voici les commandes et les fichiers que nous avons abordés dans cette leçon :

`crontab`

Gérer les fichiers `crontab` pour les utilisateurs particuliers.

`/etc/cron.allow` et `/etc/cron.deny`

Fichiers permettant de définir les restrictions de `crontab`.

`/etc/crontab`

Fichier `crontab` du système.

`/etc/cron.d`

Le répertoire qui contient les fichiers `crontab` du système.

`systemctl`

Contrôle le gestionnaire de système et de services `systemd`. En relation avec les timers, la commande peut être utilisée pour les activer et les démarrer.

Réponses aux exercices guidés

1. Pour chacun des raccourcis `crontab` suivants, indiquez la spécification de temps correspondante (les cinq premières colonnes d'un fichier `crontab`) :

@hourly	0 * * * *
@daily	0 0 * * *
@weekly	0 0 * * 0
@monthly	0 0 1 * *
@annually	0 0 1 1 *

2. Pour chacun des raccourcis `OnCalendar` suivants, indiquez la spécification temporelle correspondante (la forme normalisée longue) :

hourly	*-*-* *:00:00
daily	*-*-* 00:00:00
weekly	Mon *-*-* 00:00:00
monthly	*-*-01 00:00:00
yearly	*-01-01 00:00:00

3. Expliquez la signification des spécifications temporelles suivantes pour un fichier `crontab` :

30 13 * * 1-5	À 13h30 tous les jours de la semaine du lundi au vendredi
00 09-18 * * *	Tous les jours et toutes les heures de 09h00 à 18h00
30 08 1 1 *	Le premier jour du mois de janvier à 8h30
0,20,40 11 * * Sun	Chaque dimanche à 11h00, 11h20 et 11h40
00 09 10-20 1-3 *	À 09h00 du 10 au 20 janvier, février et mars
*/20 * * * *	Toutes les vingt minutes

4. Expliquez la signification des spécifications temporelles suivantes utilisées dans l'option `OnCalendar` d'un fichier `timer` :

--* 08:30:00	Tous les jours à 08h30
----------------	------------------------

Sat,Sun *-*- * 05:00:00	À 05h00 le samedi et le dimanche
--01 13:15,30,45:00	À 13 h 15, 13 h 30 et 13 h 45 le premier jour du mois
Fri *-09..12-* 16:20:00	À 16 h 20 tous les vendredi des mois de septembre, octobre, novembre et décembre
Mon,Tue *-*-1,15 08:30:00	Le premier ou le quinzième jour de chaque mois, à 8h30, uniquement si ce jour est un lundi ou un mardi.
--* *:00/05:00	Toutes les cinq minutes

Réponses aux exercices d'approfondissement

1. En partant du principe que vous êtes autorisé à planifier des tâches avec `cron` en tant qu'utilisateur ordinaire, quelle commande utiliseriez-vous pour créer votre propre fichier `crontab` ?

```
dave@hostname ~ $ crontab -e
no crontab for dave - using an empty one

Select an editor. To change later, run 'select-editor'.
 1. /bin/ed
 2. /bin/nano      < ---- easiest
 3. /usr/bin/emacs24
 4. /usr/bin/vim.tiny

Choose 1-4 [2]:
```

2. Créez une tâche planifiée simple qui exécute la commande `date` tous les vendredi à 01:00 de l'après-midi. Où pouvez-vous voir le résultat de cette tâche ?

```
00 13 * * 5 date
```

Le résultat est envoyé par e-mail à l'utilisateur ; pour le visualiser, vous pouvez utiliser la commande `mail`.

3. Créez une autre tâche planifiée qui exécute le script `foobar.sh` toutes les minutes, en redirigeant la sortie vers le fichier `output.log` dans votre répertoire personnel de façon à ce que seule l'erreur standard vous soit envoyée par e-mail.

```
* /1 * * * * ./foobar.sh >> output.log
```

4. Jetez un œil à l'entrée `crontab` de la tâche planifiée nouvellement créée. Pourquoi ne pas préciser le chemin absolu du fichier dans lequel la sortie standard est sauvegardée ? Et pourquoi utiliser la commande `./foobar.sh` pour exécuter le script ?

`cron` invoque les commandes depuis le répertoire personnel de l'utilisateur, à moins qu'un autre emplacement ne soit spécifié par la variable d'environnement `HOME` dans le fichier `crontab`. Pour cette raison, vous pouvez utiliser le chemin relatif du fichier de sortie et lancer le script avec `./foobar.sh`.

5. Éditez l'entrée `crontab` précédente en supprimant la redirection de sortie et désactivez la première tâche cron que vous avez créée.

```
#00 13 * * 5 date
*/1 * * * * ./foobar.sh
```

Pour désactiver une tâche cron, il suffit de commenter la ligne correspondante dans le fichier `crontab`.

6. Comment faire pour envoyer le résultat ainsi que les erreurs de votre tâche planifiée au compte utilisateur `emma` par e-mail ? Et comment éviter d'envoyer la sortie standard et les erreurs par e-mail ?

Pour envoyer la sortie standard et les erreurs à `emma`, vous devez définir la variable d'environnement `MAILTO` dans votre fichier `crontab` comme ceci :

```
MAILTO="emma"
```

Pour indiquer à cron qu'aucun e-mail ne doit être envoyé, vous pouvez assigner une valeur vide à la variable d'environnement `MAILTO`.

```
MAILTO=""
```

7. Exécutez la commande `ls -l /usr/bin/crontab`. Quelle permission spéciale est activée et quelle est sa signification ?

```
$ ls -l /usr/bin/crontab
-rwxr-sr-x 1 root crontab 25104 feb 10 2015 /usr/bin/crontab
```

La commande `crontab` a le bit SGID défini (le caractère `s` à la place du fanion exécutable pour le groupe), ce qui signifie qu'elle est exécutée avec les privilèges du groupe (en l'occurrence `crontab`). C'est pourquoi les utilisateurs `lambda` peuvent éditer leur fichier `crontab` en utilisant la commande `crontab`. Notez que dans la plupart des distributions, les permissions sur les fichiers sont définies de telle sorte que les fichiers `crontab` ne peuvent être édités que par la commande `crontab`.



107.2 Leçon 2

Certification :	LPIC-1
Version :	5.0
Thème :	107 Tâches administratives
Objectif :	107.2 Automatisation des tâches d'administration par la planification des travaux
Leçon :	2 sur 2

Introduction

Comme vous l'avez vu dans la précédente leçon, vous pouvez planifier des tâches régulières en utilisant `cron` ou les *timers* de `systemd`. Or, il peut arriver que vous ayez besoin d'exécuter une tâche une seule fois à un moment donné dans le futur. Pour ce faire, vous pouvez utiliser un autre outil puissant : la commande `at`.

Planifier les tâches avec `at`

La commande `at` est utilisée pour la planification d'une seule tâche. Elle nécessite seulement de spécifier à quel moment la tâche devra être exécutée dans le futur. Une fois que vous saisissez `at` suivi du temps requis, vous accédez à l'invite `at` dans laquelle vous pouvez définir les commandes à exécuter. La combinaison de touches `Ctrl` + `D` vous permet alors de quitter cette invite.

```
$ at now +5 minutes
warning: commands will be executed using /bin/sh
at> date
```

```
at> Ctrl+D
job 12 at Sat Sep 14 09:15:00 2019
```

La tâche `at` dans l'exemple ci-dessus exécute simplement la commande `date` au bout de cinq minutes. Comme pour `cron`, la sortie standard et les erreurs vous sont envoyées par e-mail. Notez que le démon `atd` doit tourner sur le système pour que vous puissiez utiliser la planification des tâches `at`.

NOTE

Sous Linux, la commande `batch` est similaire à `at`, mais les tâches `batch` ne sont exécutées que lorsque la charge du système est suffisamment basse pour le permettre.

Voici les options les plus importantes qui peuvent s'appliquer à la commande `at` :

-c

Afficher les commandes d'un ID de tâche donné sur la sortie standard.

-d

Supprimer les tâches en fonction de leur ID. C'est un alias de `atrm`.

-f

Lire la tâche depuis un fichier au lieu de l'entrée standard.

-l

Afficher la liste des tâches en attente de l'utilisateur. Si l'utilisateur est `root`, toutes les tâches de tous les utilisateurs sont affichées. C'est un alias de `atq`.

-m

Envoyer un e-mail à l'utilisateur à la fin de la tâche, même en l'absence de résultat.

-q

Spécifier une file d'attente sous forme d'une seule lettre de `a` à `z` et de `A` à `Z` (par défaut `a` pour `at` et `b` pour `batch`). Les tâches dans les files d'attente avec les lettres les plus élevées sont exécutées avec un `nice` plus élevé. Les tâches soumises à une file d'attente en lettres majuscules sont traitées comme des tâches `batch`.

-v

Afficher l'heure à laquelle la tâche sera exécutée avant de la lire.

Afficher les tâches planifiées avec `atq`

Planifions maintenant deux autres tâches `at` : la première exécute le script `foo.sh` à 09h30, tandis que la seconde exécute le script `bar.sh` au bout d'une heure.

```
$ at 09:30 AM
warning: commands will be executed using /bin/sh
at> ./foo.sh
at> Ctrl+D
job 13 at Sat Sep 14 09:30:00 2019
$ at now +2 hours
warning: commands will be executed using /bin/sh
at> ./bar.sh
at> Ctrl+D
job 14 at Sat Sep 14 11:10:00 2019
```

Pour afficher la liste des tâches en attente, vous pouvez utiliser la commande `atq` qui présente les informations suivantes pour chaque tâche : ID de la tâche, date et heure d'exécution, file d'attente et nom d'utilisateur.

```
$ atq
14      Sat Sep 14 11:10:00 2019 a frank
13      Sat Sep 14 09:30:00 2019 a frank
12      Sat Sep 14 09:15:00 2019 a frank
```

Rappelez-vous que la commande `at -l` est un alias de `atq`.

NOTE

Si vous lancez `atq` en tant que `root`, les tâches en file d'attente seront affichées pour tous les utilisateurs.

Supprimer des tâches avec `atrm`

Si vous voulez supprimer une tâche `at`, vous pouvez utiliser la commande `atrm` suivie de l'ID de la tâche. Par exemple, pour supprimer la tâche avec l'ID 14, vous pouvez exécuter la commande suivante :

```
$ atrm 14
```

Vous pouvez supprimer plusieurs tâches avec `atrm` en spécifiant plusieurs ID séparés par des espaces. Rappelez-vous que la commande `at -d` est un alias de `atrm`.

NOTE

Si vous lancez `atrm` en tant que `root`, vous pouvez supprimer les tâches de tous les utilisateurs.

Configurer l'accès à la planification des tâches

L'autorisation pour les utilisateurs ordinaires de planifier des tâches `at` est établie par les fichiers `/etc/at.allow` et `/etc/at.deny`. Si `/etc/at.allow` existe, seuls les utilisateurs non-root mentionnés dans ce fichier peuvent planifier des tâches `at`. Si `/etc/at.allow` n'existe pas mais que `/etc/at.deny` existe, les utilisateurs non-root énumérés dans ce fichier n'ont pas le droit de planifier des tâches `at` (dans ce cas, un fichier `/etc/at.deny` vide signifie que chaque utilisateur a le droit de planifier des tâches `at`). Si aucun de ces fichiers n'existe, l'accès des utilisateurs à la planification des tâches `at` dépend de la distribution utilisée.

Spécifications horaires

Vous pouvez spécifier l'heure d'exécution de la tâche `at` en utilisant le format `HH:MM`, éventuellement suivi de `AM` ou `PM` pour le format 12 heures. Si l'heure spécifiée est déjà révolue, le lendemain sera pris en compte. Si vous souhaitez programmer une date particulière à laquelle la tâche sera exécutée, vous devez ajouter la date après l'heure en utilisant un des formats suivants : `nom-du-mois jour-du-mois`, `nom-du-mois jour-du mois année`, `MMDDYY`, `MM/DD/YY`, `DD.MM.YY` et `YYYY-MM-DD`).

Les mots-clés suivants sont également acceptés : `midnight`, `noon`, `teatime` (16:00) et `now` suivi d'un signe plus (+) et d'un laps de temps (minutes, heures, jours et semaines). Enfin, vous pouvez demander à `at` d'exécuter la tâche aujourd'hui ou demain en faisant précéder l'heure des mots `today` ou `tomorrow`. Par exemple, vous pouvez utiliser `at 07:15 AM Jan 01` pour exécuter une tâche à 07:15 du matin le 1er janvier et `at now +5 minutes` pour exécuter une tâche dans cinq minutes à partir de maintenant. Pour en savoir plus sur la définition exacte des spécifications horaires, consultez le fichier `timespec` dans l'arborescence `/usr/share`.

Une alternative à at

Si vous utilisez `systemd` comme système d'initialisation, vous pouvez également planifier des tâches ponctuelles avec la commande `systemd-run`. Elle est typiquement utilisée pour créer une unité de *timer* transitoire qui permet d'exécuter une commande à un moment donné sans qu'il soit nécessaire de créer un fichier de service pour autant. Par exemple, en tant que `root`, vous pouvez exécuter la commande `date` à 11:30 du matin le 10 octobre 2019 en utilisant ce qui suit :

```
# systemd-run --on-calendar='2019-10-06 11:30' date
```

Et si vous voulez exécuter le script `foo.sh` dans votre répertoire courant au bout de deux minutes, vous pouvez utiliser :

```
# systemd-run --on-active="2m" ./foo.sh
```

Consultez les pages du manuel `systemd-run(1)` pour connaître toutes les utilisations possibles de `systemd-run`.

NOTE

Rappelez-vous que les *timers* sont enregistrés dans le journal de `systemd` et que vous pouvez consulter les journaux des différentes unités en utilisant la commande `journalctl`. Rappelez-vous également que si vous travaillez en tant qu'utilisateur ordinaire, vous devez utiliser l'option `--user` avec les commandes `systemd-run` et `journalctl`.

Exercices guidés

1. Pour chacune des spécifications horaires suivantes, indiquez celles qui sont valides et celles qui sont non valides pour `at` :

<code>at 08:30 AM next week</code>	
<code>at midday</code>	
<code>at 01-01-2020 07:30 PM</code>	
<code>at 21:50 01.01.20</code>	
<code>at now +4 days</code>	
<code>at 10:15 PM 31/03/2021</code>	
<code>at tomorrow 08:30 AM</code>	

2. Une fois que vous avez planifié une tâche avec `at`, comment pouvez-vous afficher le détail des commandes ?

3. Quelles commandes pouvez-vous utiliser pour consulter vos tâches `at` en attente ? Et quelles commandes utiliseriez-vous pour les supprimer ?

4. Avec `systemd`, quelle commande est utilisée comme alternative à `at` ?

Exercices d'approfondissement

1. Créez une tâche `at` qui exécute le script `foo.sh` dans votre répertoire personnel le 31 octobre prochain à 10h30. Vous travaillez en tant qu'utilisateur ordinaire.

2. Connectez-vous au système en tant que nouvel utilisateur ordinaire et créez une autre tâche `at` qui exécute le script `bar.sh` demain à 10h00. Le script est rangé dans le répertoire personnel de l'utilisateur.

3. Reconnectez-vous encore une fois pour changer d'identité et créez une autre tâche `at` qui exécute le script `foobar.sh` au bout de 30 minutes. Le script est rangé dans le répertoire personnel de l'utilisateur.

4. Connectez-vous en tant que `root` et lancez la commande `atq` pour passer en revue les tâches planifiées `at` de tous les utilisateurs. Que se passe-t-il si un utilisateur ordinaire exécute cette commande ?

5. En tant que `root`, supprimez toutes ces tâches `at` en attente avec une seule commande.

6. Lancez la commande `ls -l /usr/bin/at` et observez les permissions.

Résumé

Voici ce que nous avons appris dans cette leçon :

- Utiliser `at` pour exécuter des tâches ponctuelles à un moment défini.
- Gérer les tâches `at`.
- Configurer l'accès des utilisateurs à la planification des tâches `at`.
- Utiliser `systemd-run` comme alternative à `at`.

Voici les commandes et les fichiers que nous avons abordés dans cette leçon :

`at`

Exécuter des commandes à un moment donné.

`atq`

Afficher la liste des tâches `at` en attente de l'utilisateur, à moins qu'il ne s'agisse du superutilisateur.

`atrm`

Supprimer les tâches `at`, identifiées par leur numéro de tâche.

`/etc/at.allow` and `/etc/at.deny`

Fichiers utilisés pour définir les restrictions de `at`.

`systemd-run`

Créer et démarrer une unité de *timer* transitoire comme alternative à `at` pour la planification ponctuelle.

Réponses aux exercices guidés

1. Pour chacune des spécifications horaires suivantes, indiquez celles qui sont valides et celles qui sont non valides pour `at` :

<code>at 08:30 AM next week</code>	Valide
<code>at midday</code>	Non valide
<code>at 01-01-2020 07:30 PM</code>	Non valide
<code>at 21:50 01.01.20</code>	Valide
<code>at now +4 days</code>	Valide
<code>at 10:15 PM 31/03/2021</code>	Non valide
<code>at tomorrow 08:30 AM monotonic</code>	Non valide

2. Une fois que vous avez planifié une tâche avec `at`, comment pouvez-vous afficher le détail des commandes ?

Vous pouvez utiliser la commande `at -c` suivie de l'ID de la tâche dont vous souhaitez vérifier les commandes. Notez que le résultat contient également la plupart de l'environnement actif au moment où la tâche a été planifiée. Rappelez-vous que l'utilisateur `root` peut consulter les tâches de tous les utilisateurs.

3. Quelles commandes pouvez-vous utiliser pour consulter vos tâches `at` en attente ? Et quelles commandes utiliseriez-vous pour les supprimer ?

Vous pouvez utiliser la commande `at -l` pour consulter vos tâches en attente, et la commande `at -d` pour les supprimer. `at -l` est un alias de `atq` et `at -d` un alias de `atrm`. Rappelez-vous que `root` peut afficher et supprimer les tâches de tous les utilisateurs.

4. Avec `systemd`, quelle commande est utilisée comme alternative à `at` ?

La commande `systemd-run` peut être utilisée comme alternative à `at` pour planifier des tâches ponctuelles. Par exemple, vous pouvez l'utiliser pour exécuter des commandes à un moment précis, en définissant un *calendar timer* ou un *monotonic timer* par rapport à différents points de départ.

Réponses aux exercices d'approfondissement

1. Créez une tâche `at` qui exécute le script `foo.sh` dans votre répertoire personnel le 31 octobre prochain à 10h30. Vous travaillez en tant qu'utilisateur ordinaire.

```
$ at 10:30 AM October 31
warning: commands will be executed using /bin/sh
at> ./foo.sh
at> Ctrl+D
job 50 at Thu Oct 31 10:30:00 2019
```

2. Connectez-vous au système en tant que nouvel utilisateur ordinaire et créez une autre tâche `at` qui exécute le script `bar.sh` demain à 10h00. Le script est rangé dans le répertoire personnel de l'utilisateur.

```
$ at 10:00 AM tomorrow
warning: commands will be executed using /bin/sh
at> ./bar.sh
at> Ctrl+D
job 51 at Sun Oct 6 10:00:00 2019
```

3. Reconnectez-vous encore une fois pour changer d'identité et créez une autre tâche `at` qui exécute le script `foobar.sh` au bout de 30 minutes. Le script est rangé dans le répertoire personnel de l'utilisateur.

```
$ at now +30 minutes
warning: commands will be executed using /bin/sh
at> ./foobar.sh
at> Ctrl+D
job 52 at Sat Oct 5 10:19:00 2019
```

4. Connectez-vous en tant que `root` et lancez la commande `atq` pour passer en revue les tâches planifiées `at` de tous les utilisateurs. Que se passe-t-il si un utilisateur ordinaire exécute cette commande ?

```
# atq
52      Sat Oct  5 10:19:00 2019 a dave
50      Thu Oct 31 10:30:00 2019 a frank
51      Sun Oct  6 10:00:00 2019 a emma
```

Si vous invoquez la commande `atq` en tant que root, toutes les tâches `at` en attente de tous les utilisateurs sont énumérées. Si vous l'exécutez en tant qu'utilisateur ordinaire, seules vos propres tâches `at` en attente sont affichées.

5. En tant que root, supprimez toutes ces tâches `at` en attente avec une seule commande.

```
# atrm 50 51 52
```

6. En tant que root, lancez la commande `ls -l /usr/bin/at` et observez les permissions.

```
# ls -l /usr/bin/at
-rwsr-sr-x 1 daemon daemon 43762 Dec  1 2015 /usr/bin/at
```

Dans cette distribution, la commande `at` dispose à la fois des bits SUID (caractère `s` au lieu du fanion exécutable pour le propriétaire) et SGID (caractère `s` au lieu du fanion exécutable pour le groupe), ce qui signifie qu'elle sera exécutée avec les privilèges du propriétaire et du groupe du fichier (daemon pour les deux). C'est la raison pour laquelle les utilisateurs ordinaires sont capables de planifier des tâches avec `at`.



107.3 Paramètres régionaux et langues

Référence aux objectifs de LPI

[https://wiki.lpi.org/wiki/LPIC-1_Objectives_V5.0\(FR\)#107.3_Param.C3.A8tres_r.C3.A9gionaux_et_langues](https://wiki.lpi.org/wiki/LPIC-1_Objectives_V5.0(FR)#107.3_Param.C3.A8tres_r.C3.A9gionaux_et_langues)[LPIC-1 version 5.0, Exam 102, Objective 107.3]

Valeur

3

Domaines de connaissance les plus importants

- Configuration de l'environnement linguistique et des variables d'environnement correspondantes.
- Configuration du fuseau horaire et des variables d'environnement correspondantes.

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

- `/etc/timezone`
- `/etc/localtime`
- `/usr/share/zoneinfo/`
- `LC_*`
- `LC_ALL`
- `LANG`
- `TZ`
- `/usr/bin/locale`
- `tzselect`
- `timedatectl`

- `date`
- `iconv`
- `UTF-8`
- `ISO-8859`
- `ASCII`
- `Unicode`



107.3 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	107 Tâches administratives
Objectif :	107.3 Paramètres régionaux et langues
Leçon :	1 sur 1

Introduction

Toutes les distributions Linux majeures peuvent être configurées pour utiliser des paramètres régionaux personnalisés. Ces paramètres comprennent des définitions liées à la région et à la langue, notamment le fuseau horaire, la langue de l'interface et l'encodage des caractères. Ils peuvent être modifiés lors de l'installation du système d'exploitation ou à tout moment par la suite.

Les applications s'appuient sur des variables d'environnement, des fichiers de configuration du système et des commandes pour décider de l'heure et de la langue à utiliser ; c'est pourquoi la plupart des distributions Linux ont en commun une méthode standardisée pour ajuster les paramètres relatifs à l'heure et à la localisation. Ces ajustements sont importants non seulement pour améliorer l'expérience de l'utilisateur, mais également pour s'assurer que la chronologie des événements du système — importante, par exemple, pour signaler des problèmes de sécurité — est correctement déterminée.

La représentation d'un texte écrit, quelle que soit la langue parlée, nécessite pour les systèmes d'exploitation modernes une norme de référence pour l'encodage des caractères, et les systèmes Linux ne dérogent pas à cette règle. Les ordinateurs ne peuvent traiter que des nombres, et un

caractère de texte n'est rien d'autre qu'un nombre associé à un symbole graphique. Différentes plateformes informatiques peuvent associer des valeurs numériques distinctes à un même caractère, si bien qu'une norme commune d'encodage des caractères est indispensable pour les rendre compatibles. Un document texte créé sur tel système ne sera lisible sur tel autre système que si tous deux s'accordent sur le format d'encodage et sur le nombre associé à tel ou tel caractère, ou du moins s'ils savent effectuer la conversion entre les deux normes.

La nature hétérogène des paramètres régionaux dans les systèmes Linux entraîne des différences subtiles entre les distributions. En dépit de ces différences, toutes les distributions partagent les mêmes outils et les mêmes concepts de base pour internationaliser un système.

Les fuseaux horaires

Les fuseaux horaires correspondent à des zones distinctes de la surface de la Terre, à peu près proportionnelles, et qui s'étendent sur l'équivalent d'une heure, c'est-à-dire des régions du monde qui connaissent la même heure de la journée à un moment donné. Comme il n'existe pas de longitude unique qui puisse être considérée comme le début de la journée pour le monde entier, les fuseaux horaires sont relatifs au *méridien d'origine*, où l'angle longitudinal de la Terre est défini comme étant égal à 0. L'heure au méridien d'origine est appelée *Temps universel coordonné*, abrégé en UTC par convention. Pour des raisons pratiques, les fuseaux horaires ne suivent pas la distance longitudinale exacte du point de référence (le méridien d'origine). Au lieu de cela, les fuseaux horaires s'adaptent artificiellement pour suivre les frontières des pays ou d'autres subdivisions importantes.

Les subdivisions politiques sont si importantes que les fuseaux horaires sont nommés d'après un élément géographique majeur de la région concernée, généralement le nom d'un grand pays ou d'une grande ville à l'intérieur du fuseau. Cependant, les fuseaux horaires sont subdivisés en fonction de leur décalage horaire par rapport à l'UTC et ce décalage peut également être utilisé pour identifier le fuseau en question. Le fuseau horaire *GMT-5*, par exemple, indique une région pour laquelle l'heure UTC est en avance de cinq heures, c'est-à-dire que cette région est en retard de cinq heures par rapport à l'heure UTC. De même, le fuseau horaire *GMT+3* indique une région pour laquelle l'heure UTC est en retard de trois heures. Le terme GMT — de *Greenwich Mean Time* — est utilisé comme synonyme d'UTC dans les noms de fuseaux horaires basés sur le décalage.

Une machine connectée peut être accessible depuis différentes régions du monde, c'est pourquoi il est bon de régler l'horloge interne sur UTC (le fuseau horaire GMT+0) et de laisser le choix du fuseau horaire à chaque cas particulier. Les services cloud, par exemple, sont généralement configurés pour utiliser l'UTC, étant donné que cela permet de pallier les incohérences occasionnelles entre l'heure locale et l'heure des clients ou d'autres serveurs. En contrepartie, les utilisateurs qui démarrent une session à distance sur le serveur peuvent vouloir utiliser leur

fuseau horaire local. C'est donc au système d'exploitation que revient la tâche de configurer le fuseau horaire correct en fonction de chaque cas.

En dehors de la date et de l'heure en cours, la commande `date` indique également le fuseau horaire en vigueur :

```
$ date
Mon Oct 21 10:45:21 -03 2019
```

Le décalage par rapport à l'UTC est fourni par la valeur `-03`, ce qui signifie que l'heure affichée est inférieure de trois heures à l'UTC. Par conséquent, l'heure UTC est en avance de trois heures, ce qui fait de *GMT-3* le fuseau horaire correspondant à l'heure donnée. La commande `timedatectl`, disponible dans les distributions utilisant `systemd`, affiche davantage de détails sur l'heure et la date du système :

```
$ timedatectl
                Local time: Sat 2019-10-19 17:53:18 -03
                Universal time: Sat 2019-10-19 20:53:18 UTC
                   RTC time: Sat 2019-10-19 20:53:18
                   Time zone: America/Sao_Paulo (-03, -0300)
System clock synchronized: yes
systemd-timesyncd.service active: yes
                   RTC in local TZ: no
```

Comme on peut le voir dans l'entrée `Time zone`, les noms de fuseaux horaires basés sur des localités — comme `America/Sao_Paulo` — sont également acceptés. Le fuseau horaire par défaut du système est enregistré dans le fichier `/etc/timezone`, soit par le nom descriptif complet du fuseau, soit par son décalage. Les noms génériques de fuseaux horaires définis en fonction du décalage UTC doivent inclure `Etc` dans la première partie du nom. Par exemple, pour définir le fuseau horaire par défaut à GMT+3, le nom du fuseau horaire doit être `Etc/GMT+3` :

```
$ cat /etc/timezone
Etc/GMT+3
```

Même si les noms de fuseaux horaires basés sur les localités ne nécessitent pas de décalage horaire pour fonctionner, le choix n'est pas si simple. Une même zone peut avoir plus d'un nom, ce qui peut la rendre difficile à mémoriser. Pour résoudre ce problème, la commande `tzselect` offre une méthode interactive qui guide l'utilisateur vers la bonne définition du fuseau horaire. La commande `tzselect` est disponible par défaut dans toutes les distributions Linux, dans la

mesure où elle est fournie par le paquet qui contient une série d'outils liés à la bibliothèque C de GNU.

La commande `tzselect` pourra être utile, par exemple, à un utilisateur qui souhaite déterminer le fuseau horaire de “São Paulo City” à “Brazil”. `tzselect` commence par demander la macro-région de l'endroit choisi :

```
$ tzselect
```

```
Please identify a location so that time zone rules can be set correctly.
```

```
Please select a continent, ocean, "coord", or "TZ".
```

- 1) Africa
- 2) Americas
- 3) Antarctica
- 4) Asia
- 5) Atlantic Ocean
- 6) Australia
- 7) Europe
- 8) Indian Ocean
- 9) Pacific Ocean
- 10) coord - I want to use geographical coordinates.
- 11) TZ - I want to specify the time zone using the Posix TZ format.

```
#? 2
```

L'option 2 est destinée aux pays d'Amérique du Nord et du Sud, pas forcément dans le même fuseau horaire. Vous pouvez également spécifier le fuseau horaire à l'aide de coordonnées géographiques ou de la notation décalée, également connue sous le nom de *Posix TZ format*. L'étape suivante consiste à choisir le pays :

```
Please select a country whose clocks agree with yours.
```

- | | | |
|----------------------|------------------------|--------------------------|
| 1) Anguilla | 19) Dominican Republic | 37) Peru |
| 2) Antigua & Barbuda | 20) Ecuador | 38) Puerto Rico |
| 3) Argentina | 21) El Salvador | 39) St Barthelemy |
| 4) Aruba | 22) French Guiana | 40) St Kitts & Nevis |
| 5) Bahamas | 23) Greenland | 41) St Lucia |
| 6) Barbados | 24) Grenada | 42) St Maarten (Dutch) |
| 7) Belize | 25) Guadeloupe | 43) St Martin (French) |
| 8) Bolivia | 26) Guatemala | 44) St Pierre & Miquelon |
| 9) Brazil | 27) Guyana | 45) St Vincent |
| 10) Canada | 28) Haiti | 46) Suriname |
| 11) Caribbean NL | 29) Honduras | 47) Trinidad & Tobago |
| 12) Cayman Islands | 30) Jamaica | 48) Turks & Caicos Is |
| 13) Chile | 31) Martinique | 49) United States |

```

14) Colombia          32) Mexico          50) Uruguay
15) Costa Rica        33) Montserrat      51) Venezuela
16) Cuba              34) Nicaragua       52) Virgin Islands (UK)
17) Curaçao           35) Panama           53) Virgin Islands (US)
18) Dominica          36) Paraguay
#? 9

```

Le territoire brésilien s'étend sur quatre fuseaux horaires, de sorte que les informations sur le pays ne suffisent pas pour définir le fuseau horaire. Dans l'étape suivante, `tzselect` demandera à l'utilisateur de spécifier la province :

```

Please select one of the following time zone regions.
1) Atlantic islands
2) Pará (east); Amapá
3) Brazil (northeast: MA, PI, CE, RN, PB)
4) Pernambuco
5) Tocantins
6) Alagoas, Sergipe
7) Bahia
8) Brazil (southeast: GO, DF, MG, ES, RJ, SP, PR, SC, RS)
9) Mato Grosso do Sul
10) Mato Grosso
11) Pará (west)
12) Rondônia
13) Roraima
14) Amazonas (east)
15) Amazonas (west)
16) Acre
#? 8

```

Tous les noms de villes ne sont pas disponibles, mais le choix de la région la plus proche est suffisant. L'information recueillie sera ensuite utilisée par `tzselect` pour afficher le fuseau horaire correspondant :

```

The following information has been given:

    Brazil
    Brazil (southeast: GO, DF, MG, ES, RJ, SP, PR, SC, RS)

Therefore TZ='America/Sao_Paulo' will be used.
Selected time is now:    sex out 18 18:47:07 -03 2019.
Universal Time is now:  sex out 18 21:47:07 UTC 2019.

```

Is the above information OK?

1) Yes

2) No

#? 1

You can make this change permanent for yourself by appending the line

```
TZ='America/Sao_Paulo'; export TZ
```

to the file `~/.profile` in your home directory; then log out and log in again.

Here is that TZ value again, this time on standard output so that you can use the `/usr/bin/tzselect` command in shell scripts:

```
America/Sao_Paulo
```

Le nom du fuseau horaire résultant, `America/Sao_Paulo`, pourra également être utilisé comme contenu du fichier `/etc/timezone` pour renseigner le fuseau horaire par défaut du système :

```
$ cat /etc/timezone
America/Sao_Paulo
```

Comme l'indique le résultat de `tzselect`, la variable d'environnement `TZ` définit le fuseau horaire de la session shell, indépendamment du fuseau horaire par défaut du système. Si vous ajoutez la ligne `TZ='America/Sao_Paulo'; export TZ` au fichier `~/.profile`, `America/Sao_Paulo` deviendra le fuseau horaire pour les futures sessions de l'utilisateur. La variable `TZ` peut également être modifiée temporairement pendant la session en cours, en vue d'afficher l'heure dans un fuseau horaire différent :

```
$ env TZ='Africa/Cairo' date
Mon Oct 21 15:45:21 EET 2019
```

Dans l'exemple, la commande `env` lancera la commande donnée dans un nouveau shell avec les mêmes variables d'environnement que la session actuelle à l'exception de la variable `TZ`, modifiée par l'argument `TZ='Africa/Cairo'`.

L'heure d'été

Beaucoup de pays adoptent l'heure d'été pendant une partie de l'année — lorsque les horloges sont ajustées d'une heure — ce qui peut amener un système mal configuré à indiquer une heure erronée pendant cette période de l'année.

Le fichier `/etc/localtime` contient les données utilisées par le système d'exploitation pour

ajuster l'horloge en conséquence. Les systèmes Linux standard disposent des fichiers pour tous les fuseaux horaires dans le répertoire `/usr/share/zoneinfo/`, et `/etc/localtime` n'est qu'un lien symbolique vers les données de ce répertoire. Les fichiers dans `/usr/share/zoneinfo/` sont organisés en fonction du nom du fuseau horaire correspondant, de sorte que le fichier de données pour le fuseau horaire `America/Sao_Paulo` sera `/usr/share/zoneinfo/America/Sao_Paulo`.

Comme les définitions de l'heure d'été peuvent changer, il est important de garder les fichiers dans `/usr/share/zoneinfo/` à jour. La commande de mise à jour du gestionnaire de paquets fourni par la distribution devrait les mettre à jour chaque fois qu'une nouvelle version est disponible.

Langue et encodage des caractères

Les systèmes Linux peuvent être utilisés avec une grande variété de langues et d'encodages de caractères non occidentaux, dont les définitions sont connues sous le nom de *locales*. La configuration locale la plus élémentaire concerne la définition de la variable d'environnement `LANG`, à partir de laquelle la plupart des programmes du shell identifient la langue à utiliser.

Le contenu de la variable `LANG` suit le format `ab_CD`, où `ab` représente le code de la langue et `CD` le code de la région. Le code de la langue suit la norme ISO-639 et le code de la région la norme ISO-3166. Un système configuré pour utiliser le portugais brésilien, par exemple, doit définir la variable `LANG` à `pt_BR.UTF-8` :

```
$ echo $LANG
pt_BR.UTF-8
```

Comme on peut le voir dans le résultat de l'exemple, la variable `LANG` contient également l'encodage de caractères prévu pour le système. ASCII, l'abréviation de *American Standard Code for Information Interchange*, a été la première norme d'encodage de caractères largement utilisée pour la communication électronique. En revanche, comme l'ASCII dispose d'une plage très limitée de valeurs numériques disponibles et qu'il est basé sur l'alphabet anglais, il ne contient ni les caractères utilisés par d'autres langues, ni un ensemble étendu de symboles non alphabétiques. L'encodage UTF-8 est un *Unicode Standard* pour les caractères occidentaux ordinaires, ainsi que pour de nombreux autres symboles moins courants. Comme l'indique le *Unicode Consortium* qui assure la maintenance du *Unicode Standard*, il devrait être adopté par défaut pour garantir la compatibilité entre les plateformes numériques :

La norme Unicode fournit un code unique pour chaque caractère, indépendamment de la plateforme, de l'appareil, de l'application ou de la langue. Elle a été adoptée par tous les fournisseurs de logiciels modernes et permet désormais aux données d'être transportées à

travers de nombreuses plateformes, appareils et applications différents sans être altérées. La prise en charge d'Unicode constitue la base de la représentation des langues et des symboles dans tous les principaux systèmes d'exploitation, moteurs de recherche, navigateurs, ordinateurs portables et smartphones, ainsi que sur l'internet et le World Wide Web (URL, HTML, XML, CSS, JSON, etc.). (...) la norme Unicode et la disponibilité des outils qui la prennent en charge font partie des tendances mondiales les plus significatives en matière de technologie logicielle.

— Consortium Unicode, Qu'est-ce que l'Unicode ?

Certains systèmes peuvent encore utiliser des normes ISO — comme le standard ISO-8859-1 — pour l'encodage des caractères non-ASCII. Or, ces normes d'encodage de caractères devraient être abandonnées au profit des normes d'encodage Unicode. Cela dit, tous les systèmes d'exploitation majeurs tendent à adopter la norme Unicode par défaut.

Les paramètres régionaux du système sont définis dans le fichier `/etc/locale.conf`. La variable `LANG` et les autres variables liées à la localisation y sont assignées comme de simples variables shell, par exemple :

```
$ cat /etc/locale.conf
LANG=pt_BR.UTF-8
```

Les utilisateurs peuvent très bien configurer une localisation personnalisée en modifiant la variable d'environnement `LANG`. Ils peuvent le faire pour la session en cours ou pour les sessions à venir, en ajoutant la nouvelle définition à leur profil Bash dans `~/.bash_profile` ou `~/.profile`. En revanche, en attendant que l'utilisateur se connecte, les paramètres régionaux par défaut du système seront toujours appliqués par les programmes indépendants de l'utilisateur, comme l'écran de connexion du gestionnaire d'affichage.

TIP

La commande `localectl`, disponible sur les systèmes qui utilisent `systemd` comme système d'initialisation, peut également être utilisée pour interroger et modifier les paramètres régionaux du système. Par exemple : `localectl set-locale LANG=fr_US.UTF-8`.

En dehors de la variable `LANG`, d'autres variables d'environnement ont un effet sur des aspects spécifiques des paramètres régionaux, comme le symbole de la devise à utiliser ou le séparateur de milliers correct pour les nombres :

LC_COLLATE

Définit l'ordre alphabétique. L'un de ses usages consiste à définir l'ordre dans lequel les fichiers et les répertoires sont affichés.

LC_CTYPE

Définit la manière dont le système traitera certains jeux de caractères. Par exemple, quels caractères doivent être considérés comme des *majuscules* ou des *minuscules*.

LC_MESSAGES

Définit la langue d’affichage des messages des programmes (surtout les programmes GNU).

LC_MONETARY

Définit l’unité monétaire et le format de la devise.

LC_NUMERIC

Définit le format numérique pour les valeurs non monétaires. Son rôle principal consiste à définir les séparateurs de milliers et de décimales.

LC_TIME

Définit le format de l’heure et de la date.

LC_PAPER

Définit le format de papier standard.

LC_ALL

Remplace toutes les autres variables, y compris LANG.

La commande `locale` va afficher toutes les variables définies dans la configuration linguistique en vigueur :

```
$ locale
LANG=pt_BR.UTF-8
LC_CTYPE="pt_BR.UTF-8"
LC_NUMERIC=pt_BR.UTF-8
LC_TIME=pt_BR.UTF-8
LC_COLLATE="pt_BR.UTF-8"
LC_MONETARY=pt_BR.UTF-8
LC_MESSAGES="pt_BR.UTF-8"
LC_PAPER=pt_BR.UTF-8
LC_NAME=pt_BR.UTF-8
LC_ADDRESS=pt_BR.UTF-8
LC_TELEPHONE=pt_BR.UTF-8
LC_MEASUREMENT=pt_BR.UTF-8
LC_IDENTIFICATION=pt_BR.UTF-8
LC_ALL=
```

La seule variable non définie est `LC_ALL`, qui peut être utilisée pour remplacer temporairement tous les autres paramètres linguistiques. L'exemple ci-dessous montre comment la commande `date`—exécutée dans un système configuré avec la locale `pt_BR.UTF-8`—va modifier son résultat pour se conformer à la nouvelle variable `LC_ALL` :

```
$ date
seg out 21 10:45:21 -03 2019
$ env LC_ALL=en_US.UTF-8 date
Mon Oct 21 10:45:21 -03 2019
```

La modification de la variable `LC_ALL` a permis d'afficher les abréviations du jour de la semaine et du mois en anglais américain (`en_US`). En revanche, il n'est pas forcément nécessaire de définir les mêmes paramètres linguistiques pour toutes les variables. On peut très bien, par exemple, choisir la langue `pt_BR` et le format numérique (`LC_NUMERIC`) au standard américain.

Certains paramètres régionaux modifient la façon dont les programmes gèrent l'ordre alphabétique et le format des nombres. Alors que les programmes conventionnels sont généralement prêts à choisir correctement une locale commune pour de telles situations, les scripts peuvent se comporter de manière inattendue lorsqu'ils tentent de classer correctement une liste d'éléments par ordre alphabétique, par exemple. Pour cette raison, il est recommandé d'attribuer à la variable d'environnement `LANG` la valeur de la locale courante `C`, comme dans `LANG=C`, de façon à ce que le script produise des résultats non ambigus, quelles que soient les définitions de localisation utilisées dans le système où il est exécuté. Le paramètre local `C` n'effectue qu'une simple comparaison par ordre alphabétique, de sorte que ses résultats seront également plus probants que ceux des autres paramètres.

Conversion de l'encodage

Un texte peut s'afficher avec des caractères illisibles lorsqu'il est affiché sur un système avec une configuration d'encodage de caractères différente de celle du système sur lequel le texte a été créé. On peut alors utiliser la commande `iconv` pour résoudre ce problème, en convertissant le fichier depuis son encodage d'origine vers l'encodage souhaité. Par exemple, pour convertir un fichier nommé `original.txt` depuis l'encodage `ISO-8859-1` vers le fichier nommé `converted.txt` avec l'encodage `UTF-8`, on peut utiliser la commande suivante :

```
$ iconv -f ISO-8859-1 -t UTF-8 original.txt > converted.txt
```

L'option `-f ISO-8859-1` (ou `--from-code=ISO-8859-1`) définit l'encodage du fichier d'origine et l'option `-t UTF-8` (ou `--to-code=UTF-8`) définit l'encodage du fichier transformé. Tous les

encodages supportés par la commande `iconv` sont affichés avec la commande `iconv -l` ou `iconv --list`. Au lieu d'utiliser la redirection de sortie, comme dans l'exemple, l'option `-o converti.txt` ou `--output converti.txt` peut aussi être appliquée.

Exercices guidés

1. D'après le résultat suivant de la commande `date`, quel est le fuseau horaire du système en notation GMT ?

```
$ date
Mon Oct 21 18:45:21 +05 2019
```

2. Vers quel fichier doit pointer le lien symbolique `/etc/localtime` pour que `Europe/Brussels` soit l'heure locale par défaut du système ?

3. Les caractères des fichiers texte peuvent ne pas être rendus correctement dans un système dont l'encodage de caractères est différent de celui utilisé dans le document texte. Comment utiliser `iconv` pour convertir le fichier `old.txt` encodé en `WINDOWS-1252` vers le fichier `new.txt` en encodage `UTF-8` ?

Exercices d'approfondissement

1. Quelle commande fera de Pacific/Auckland le fuseau horaire par défaut pour la session shell en cours ?

2. La commande `uptime` affiche, entre autres, la *charge moyenne* du système avec des décimales. Elle utilise les paramètres régionaux en cours pour décider si le séparateur de décimales doit être un point ou une virgule. Si, par exemple, la locale courante est `de_DE.UTF-8` (la localisation standard de l'Allemagne), `uptime` va utiliser une virgule comme séparateur. Sachant que dans la langue anglaise américaine c'est le point qui est utilisé comme séparateur, quelle commande fera en sorte que `uptime` affiche les décimales en utilisant un point au lieu d'une virgule pour le reste de la session en cours ?

3. La commande `iconv` va remplacer tous les caractères en dehors du jeu de caractères cible par un point d'interrogation. Si `//TRANSLIT` est ajouté à l'encodage cible, les caractères absents du jeu de caractères cible seront remplacés (translittérés) par un ou plusieurs caractères similaires. Comment peut-on utiliser cette méthode pour convertir un fichier texte UTF-8 nommé `readme.txt` en un fichier ASCII nommé `ascii.txt` ?

Résumé

Cette leçon explique la configuration d'un système Linux pour qu'il fonctionne avec des langues et des paramètres horaires personnalisés. Les concepts d'encodage des caractères et leur paramétrage sont également abordés, étant donné qu'ils sont très importants pour assurer un rendu correct du contenu textuel. La leçon aborde les sujets suivants :

- Comment les systèmes Linux sélectionnent la langue d'affichage des messages du shell.
- Comprendre comment les fuseaux horaires affectent l'heure locale.
- Comment identifier le fuseau horaire approprié et modifier les paramètres du système en conséquence.
- Qu'est-ce qu'un encodage de caractères et comment faire la conversion entre les différents encodages.

Voici les commandes et les procédures abordées :

- Variables d'environnement liées à la langue et à l'heure, telles que `LC_ALL`, `LANG` et `TZ`.
- `/etc/timezone`
- `/etc/localtime`
- `/usr/share/zoneinfo/`
- `locale`
- `tzselect`
- `timedatectl`
- `date`
- `iconv`

Réponses aux exercices guidés

1. D'après le résultat suivant de la commande `date`, quel est le fuseau horaire du système en notation GMT ?

```
$ date
Mon Oct 21 18:45:21 +05 2019
```

Il s'agit du fuseau horaire `Etc/GMT+5`.

2. Vers quel fichier doit pointer le lien symbolique `/etc/localtime` pour que `Europe/Brussels` soit l'heure locale par défaut du système ?

Le lien `/etc/localtime` doit pointer vers `/usr/share/zoneinfo/Europe/Brussels`.

3. Les caractères des fichiers texte peuvent ne pas être rendus correctement dans un système dont l'encodage de caractères est différent de celui utilisé dans le document texte. Comment utiliser `iconv` pour convertir le fichier `old.txt` encodé en `WINDOWS-1252` vers le fichier `new.txt` en encodage `UTF-8` ?

La commande `iconv -f WINDOWS-1252 -t UTF-8 -o new.txt old.txt` va effectuer la conversion requise.

Réponses aux exercices d'approfondissement

1. Quelle commande fera de Pacific/Auckland le fuseau horaire par défaut pour la session shell en cours ?

```
export TZ=Pacific/Auckland
```

2. La commande `uptime` affiche, entre autres, la *charge moyenne* du système avec des décimales. Elle utilise les paramètres régionaux en cours pour décider si le séparateur de décimales doit être un point ou une virgule. Si, par exemple, la locale courante est de `_DE.UTF-8` (la localisation standard de l'Allemagne), `uptime` va utiliser une virgule comme séparateur. Sachant que dans la langue anglaise américaine c'est le point qui est utilisé comme séparateur, quelle commande fera en sorte que `uptime` affiche les décimales en utilisant un point au lieu d'une virgule pour le reste de la session en cours ?

La commande `export LC_NUMERIC=en_US.UTF-8` ou `export LC_ALL=en_US.UTF-8`.

3. La commande `iconv` va remplacer tous les caractères en dehors du jeu de caractères cible par un point d'interrogation. Si `//TRANSLIT` est ajouté à l'encodage cible, les caractères absents du jeu de caractères cible seront remplacés (translittérés) par un ou plusieurs caractères similaires. Comment peut-on utiliser cette méthode pour convertir un fichier texte UTF-8 nommé `readme.txt` en un fichier ASCII nommé `ascii.txt` ?

La commande `iconv -f UTF-8 -t ASCII//TRANSLIT -o ascii.txt readme.txt` va effectuer la conversion requise.



**Linux
Professional
Institute**

Thème 108: Services systèmes essentiels



108.1 Gestion de l'horloge système

Référence aux objectifs de LPI

LPIC-1 version 5.0, Exam 102, Objective 108.1

Valeur

3

Domaines de connaissance les plus importants

- Configuration de la date et de l'heure système.
- Configuration de l'horloge matérielle correctement en temps UTC.
- Configuration du fuseau horaire.
- Configuration élémentaire de NTP avec ntpd et chrony.
- Connaissance du service pool.ntp.org.
- Connaissance de base de la commande ntpq.

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

- `/usr/share/zoneinfo/`
- `/etc/timezone`
- `/etc/localtime`
- `/etc/ntp.conf`
- `/etc/chrony.conf`
- `date`
- `hwclock`
- `timedatectl`

- ntpd
- ntpdate
- chronyc
- pool.ntp.org



108.1 Leçon 1

Certification :	LPIC-1 (102)
Version :	5.0
Thème :	108 Services Systèmes Essentiels
Objectif :	108.1 Maintenir l'heure du système
Leçon :	1 sur 2

Introduction

Une gestion précise du temps est un élément fondamental de l'informatique moderne. En revanche, la mise en œuvre de la mesure du temps s'avère d'une complexité surprenante. La pratique de la gestion du temps peut sembler triviale pour un utilisateur final, mais le système doit être capable de gérer intelligemment toute une série d'idiosyncrasies et de cas particuliers. Il faut savoir que les fuseaux horaires ne sont pas statiques, mais qu'ils peuvent être modifiés par une décision administrative ou politique. Un pays peut décider de ne plus observer l'heure d'été. Un système doit être capable de gérer ces changements de manière logique. Heureusement pour les administrateurs système, les solutions de gestion du temps sous Linux sont matures et robustes et fonctionnent généralement sans trop d'interférences.

Lorsqu'un système sous Linux démarre, il commence à garder l'heure. Nous parlons alors d'une *horloge système*, étant donné qu'elle est mise à jour par le système d'exploitation. Par ailleurs, les ordinateurs modernes disposent également d'une *horloge matérielle* ou *horloge en temps réel*. Cette horloge matérielle est souvent intégrée à la carte mère et reste à l'heure, que l'ordinateur soit allumé ou non. Lors du démarrage, l'heure du système est réglée sur l'horloge matérielle, mais la plupart du temps, ces deux horloges fonctionnent indépendamment l'une de l'autre. Dans cette leçon, nous aborderons les méthodes qui permettent d'interagir à la fois avec l'horloge

système et l'horloge matérielle.

Sur la plupart des systèmes Linux modernes, l'heure du système et l'heure matérielle sont synchronisées avec l'heure du réseau, qui est mise en œuvre par le *Network Time Protocol* (NTP). Dans la grande majorité des cas, la seule configuration qu'un utilisateur normal devra faire consistera à définir son fuseau horaire et le protocole NTP s'occupera du reste. Cependant, nous allons aborder certaines façons de gérer l'heure manuellement et les spécificités de la configuration de l'heure du réseau seront abordées dans la prochaine leçon.

Heure locale et heure universelle

L'horloge système est réglée sur le temps universel coordonné (UTC), qui correspond à l'heure locale à Greenwich, au Royaume-Uni. En général, un utilisateur souhaite connaître son *heure locale*. L'heure locale est calculée en partant de l'heure UTC et en appliquant un *décalage* en fonction du fuseau horaire et de l'heure d'été. Cette méthode permet d'éviter beaucoup de complexité.

L'horloge système peut être réglée sur l'heure UTC ou sur l'heure locale, mais il est recommandé de la régler également sur l'heure UTC.

La date

La commande `date` est un outil de base qui affiche simplement l'heure locale :

```
$ date
Sun Nov 17 12:55:06 EST 2019
```

Les options de la commande `date` permettent de jouer sur le format d'affichage.

Par exemple, un utilisateur peut utiliser `date -u` pour afficher l'heure UTC actuelle.

```
$ date -u
Sun Nov 17 18:02:51 UTC 2019
```

D'autres options couramment utilisées affichent l'heure locale dans un format conforme à un format RFC standard :

-I

Date/heure au format ISO 8601. L'ajout de `date (-I date)` limitera l'affichage à la seule date. Les autres formats sont `hours`, `minutes`, `seconds` et `ns` pour les nanosecondes.

-R

Renvoie la date et l'heure au format RFC 5322.

--rfc-3339

Renvoie la date et l'heure au format RFC 3339.

Le format d'affichage de `date` peut être personnalisé par l'utilisateur avec des séquences spécifiées dans le manuel. Par exemple, l'heure actuelle peut être formatée en heure Unix de la manière suivante :

```
$ date +%s
1574014515
```

La page de manuel de `date` montre que `%s` fait référence au temps Unix.

Le temps Unix est utilisé en interne sur la plupart des systèmes de type Unix. Il stocke le temps UTC comme le nombre de secondes depuis *Epoch*, qui a été défini comme étant le 1er janvier 1970.

NOTE

Le nombre de bits requis pour stocker l'heure Unix à l'heure actuelle est de 32 bits. À l'avenir, ces 32 bits ne suffiront plus pour contenir l'heure actuelle au format Unix. Cela va entraîner de graves problèmes pour tous les systèmes Linux 32 bits. Heureusement, cela ne se produira pas avant le 19 janvier 2038.

En utilisant ces séquences, nous sommes en mesure de formater la date et l'heure dans presque tous les formats requis par n'importe quelle application. Bien entendu, dans la plupart des cas, il est préférable de respecter une norme acceptée.

Par ailleurs, `date --date` peut être utilisé pour formater une heure qui n'est pas l'heure actuelle. Dans ce cas de figure, un utilisateur peut spécifier la date à appliquer au système en utilisant l'heure Unix par exemple :

```
$ date --date='@1564013011'
Wed Jul 24 20:03:31 EDT 2019
```

L'option `--debug` peut s'avérer fort utile pour s'assurer qu'une date peut être traitée correctement. Observez ce qui se passe lorsque vous passez une date valide à la commande :

```
$ date --debug --date="Fri, 03 Jan 2020 14:00:17 -0500"
date: parsed day part: Fri (day ordinal=0 number=5)
date: parsed date part: (Y-M-D) 2020-01-03
```

```

date: parsed time part: 14:00:17 UTC-05
date: input timezone: parsed date/time string (-05)
date: using specified time as starting value: '14:00:17'
date: warning: day (Fri) ignored when explicit dates are given
date: starting date/time: '(Y-M-D) 2020-01-03 14:00:17 TZ=-05'
date: '(Y-M-D) 2020-01-03 14:00:17 TZ=-05' = 1578078017 epoch-seconds
date: timezone: system default
date: final: 1578078017.000000000 (epoch-seconds)
date: final: (Y-M-D) 2020-01-03 19:00:17 (UTC)
date: final: (Y-M-D) 2020-01-03 14:00:17 (UTC-05)

```

Cet outil peut s'avérer pratique pour déboguer une application qui génère une date.

Horloge matérielle

Un utilisateur peut invoquer la commande `hwclock` pour voir l'heure telle qu'elle est maintenue sur l'horloge en temps réel. Cette commande nécessite des privilèges élevés, nous utiliserons donc `sudo` pour lancer la commande dans ce cas :

```

$ sudo hwclock
2019-11-20 11:31:29.217627-05:00

```

L'option `--verbose` renverra davantage de résultats, ce qui peut être utile pour le débogage :

```

$ sudo hwclock --verbose
hwclock from util-linux 2.34
System Time: 1578079387.976029
Trying to open: /dev/rtc0
Using the rtc interface to the clock.
Assuming hardware clock is kept in UTC time.
Waiting for clock tick...
...got clock tick
Time read from Hardware Clock: 2020/01/03 19:23:08
Hw clock time : 2020/01/03 19:23:08 = 1578079388 seconds since 1969
Time since last adjustment is 1578079388 seconds
Calculated Hardware Clock drift is 0.000000 seconds
2020-01-03 14:23:07.948436-05:00

```

Notez la dérive calculée de l'horloge matérielle (*Calculated Hardware Clock drift*). Cette donnée peut vous indiquer si l'heure système et l'heure matérielle s'écartent l'une de l'autre.

timedatectl

`timedatectl` est une commande qui peut être utilisée pour vérifier l'état général de la date et de l'heure, y compris si l'heure réseau a été synchronisée ou non (le protocole NTP sera abordé dans la prochaine leçon).

Par défaut, `timedatectl` renvoie des informations similaires à `date`, mais avec en plus l'heure RTC (matérielle) ainsi que l'état du service NTP :

```
$ timedatectl
      Local time: Thu 2019-12-05 11:08:05 EST
      Universal time: Thu 2019-12-05 16:08:05 UTC
          RTC time: Thu 2019-12-05 16:08:05
      Time zone: America/Toronto (EST, -0500)
System clock synchronized: yes
      NTP service: active
    RTC in local TZ: no
```

Régler l'heure avec timedatectl

Si NTP n'est pas disponible, il est recommandé d'utiliser `timedatectl` plutôt que `date` ou `hwclock` pour régler l'heure :

```
# timedatectl set-time '2011-11-25 14:00:00'
```

La procédure est similaire à celle de `date`. L'utilisateur peut également définir l'heure indépendamment de la date en utilisant le format HH:MM:SS.

Régler le fuseau horaire avec timedatectl

`timedatectl` est la méthode préférée pour régler le fuseau horaire local sur les systèmes Linux basés sur `systemd` en l'absence d'interface graphique. `timedatectl` va afficher la liste des fuseaux horaires disponibles, et ensuite le fuseau horaire pourra être défini avec l'un de ces derniers en argument.

Dans un premier temps, nous établissons la liste des fuseaux horaires disponibles :

```
$ timedatectl list-timezones
Africa/Abidjan
Africa/Accra
```



```
Africa/Algiers
Africa/Bissau
Africa/Cairo
...
```

La liste des fuseaux horaires disponibles est longue, et il vaut mieux utiliser la commande `grep` dans ce contexte.

Ensuite, nous pouvons définir le fuseau horaire en utilisant l'un des éléments de la liste obtenue :

```
$ timedatectl set-timezone Africa/Cairo
$ timedatectl
           Local time: Thu 2019-12-05 18:18:10 EET
       Universal time: Thu 2019-12-05 16:18:10 UTC
           RTC time: Thu 2019-12-05 16:18:10
           Time zone: Africa/Cairo (EET, +0200)
System clock synchronized: yes
              NTP service: active
          RTC in local TZ: no
```

Gardez à l'esprit que le nom du fuseau horaire doit correspondre exactement. Par exemple, `Africa/Cairo` changera le fuseau horaire, mais `cairo` ou `africa/cairo` ne le fera pas.

Désactiver NTP avec `timedatectl`

Dans certains cas, il peut être nécessaire de désactiver NTP. Cela peut être fait en utilisant `systemctl` mais nous allons le faire en utilisant `timedatectl` :

```
# timedatectl set-ntp no
$ timedatectl
           Local time: Thu 2019-12-05 18:19:04 EET Universal time: Thu 2019-12-05 16:19:04
       UTC
           RTC time: Thu 2019-12-05 16:19:04
           Time zone: Africa/Cairo (EET, +0200)
          NTP enabled: no
        NTP synchronized: no
          RTC in local TZ: no
              DST active: n/a
```

Régler le fuseau horaire sans `timedatectl`

Le réglage des informations relatives au fuseau horaire constitue une étape standard lors de l'installation de Linux sur une nouvelle machine. S'il existe une procédure d'installation graphique, cette étape sera très probablement gérée sans aucune autre intervention de l'utilisateur.

Le répertoire `/usr/share/zoneinfo` contient des informations sur les différents fuseaux horaires disponibles. Dans le répertoire `zoneinfo`, il y a des sous-répertoires avec les noms des continents ainsi que d'autres liens symboliques. Pour retrouver le répertoire `zoneinfo` de votre région, utilisez votre continent comme point de départ.

Les fichiers `zoneinfo` contiennent les règles nécessaires au calcul du décalage de l'heure locale par rapport à l'UTC. Ils sont également importants si votre région observe l'heure d'été. Le contenu de `/etc/localtime` sera lu lorsque Linux aura besoin de déterminer le fuseau horaire local. Pour définir le fuseau horaire sans passer par l'interface graphique, l'utilisateur devra créer un lien symbolique pour son emplacement à partir de `/usr/share/zoneinfo` vers `/etc/localtime`. Par exemple :

```
$ ln -s /usr/share/zoneinfo/Canada/Eastern /etc/localtime
```

Une fois le fuseau horaire correctement réglé, il est conseillé d'exécuter la commande :

```
# hwclock --systohc
```

Ceci aura pour effet de régler *l'horloge matérielle* sur *l'horloge système* (c'est-à-dire que l'horloge en temps réel sera réglée sur la même heure que `date`). Veuillez noter que cette commande est exécutée avec les privilèges de root, dans ce cas en étant connecté en tant que root.

`/etc/timezone` est similaire à `/etc/localtime`. C'est une représentation en données du fuseau horaire local, et en tant que telle, elle peut être lue en utilisant `cat` :

```
$ cat /etc/timezone  
America/Toronto
```

Notez que ce fichier n'est pas utilisé par toutes les distributions Linux.

Régler la date et l'heure sans `timedatectl`

NOTE

La plupart des systèmes Linux modernes utilisent `systemd` pour leur configuration et leurs services, et il n'est donc pas recommandé d'utiliser `date` ou `hwclock` pour régler l'heure. `systemd` utilise `timedatectl` pour cela. Ceci étant dit, il est important de connaître ces anciennes commandes dans le cas où vous auriez à administrer un système plus ancien.

Avec `date`

`date` a une option pour régler l'heure du système. Utilisez `--set` ou `-s` pour régler la date et l'heure. Vous pouvez également choisir l'option `--debug` pour vérifier l'interprétation correcte de la commande :

```
# date --set="11 Nov 2011 11:11:11"
```

Notez que les privilèges de root sont nécessaires pour régler la date ici. Nous pouvons également choisir de modifier l'heure ou la date indépendamment :

```
# date +%Y%m%d -s "20111125"
```

Ici, nous devons spécifier les séquences de manière à ce que notre chaîne soit interprétée correctement. Par exemple, `%Y` fait référence à l'année, et donc les quatre premiers chiffres 2011 seront interprétés comme l'année 2011. De même, `%T` est la séquence pour le temps, et elle est illustrée ici par le réglage du temps :

```
# date +%T -s "13:11:00"
```

Une fois l'heure système modifiée, il est recommandé de régler également l'horloge matérielle de manière à ce que l'horloge système et l'horloge matérielle soient synchronisées :

```
# hwclock --systohc
```

`systohc` signifie "horloge système vers horloge matérielle" (*system clock to hardware clock*).

Avec `hwclock`

Plutôt que de régler l'horloge système et de mettre à jour l'horloge matérielle, vous pouvez choisir

d'inverser le processus. Nous commencerons par régler l'horloge matérielle :

```
# hwclock --set --date "4/12/2019 11:15:19"
# hwclock
Fri 12 Apr 2019 6:15:19 AM EST -0.562862 seconds
```

Notez que par défaut `hwclock` s'attend à recevoir l'heure UTC, mais renvoie l'heure locale par défaut.

Une fois l'horloge matérielle réglée, nous devons mettre à jour l'horloge système à partir de celle-ci. `hctosys` peut être compris comme "horloge matérielle vers horloge système" (*hardware clock to system clock*).

```
# hwclock --hctosys
```

Exercices guidés

1. Indiquez si les commandes ci-dessous affichent ou modifient l'horloge système ou l'horloge matérielle :

Commande	Système	Matérielle	Les deux
<code>date -u</code>			
<code>hwclock --set --date "12:00:00"</code>			
<code>timedatectl</code>			
<code>timedatectl grep RTC</code>			
<code>hwclock --hctosys</code>			
<code>date +%T -s "08:00:00"</code>			
<code>timedatectl set- time 1980-01-10</code>			

2. Observez le résultat suivant, puis corrigez le format de l'argument pour que la commande fonctionne :

```
$ date --debug --date "20/20/12 0:10 -3"
```

```
date: warning: value 20 has less than 4 digits. Assuming MM/DD/YY[YY]
date: parsed date part: (Y-M-D) 0002-20-20
date: parsed time part: 00:10:00 UTC-03
date: input timezone: parsed date/time string (-03)
date: using specified time as starting value: '00:10:00'
date: error: invalid date/time value:
date:   user provided time: '(Y-M-D) 0002-20-20 00:10:00 TZ=-03'
date:   normalized time: '(Y-M-D) 0003-08-20 00:10:00 TZ=-03'
date:                                     ---- --
date:   possible reasons:
date:     numeric values overflow;
date:     incorrect timezone
date: invalid date '20/20/2 0:10 -3'
```

3. Utilisez la commande `date` avec les séquences requises pour que le mois du système soit défini à février. Laissez le reste de la date et de l'heure inchangées.

4. En supposant que la commande ci-dessus ait réussi, utilisez `hwclock` pour régler l'horloge matérielle depuis l'horloge système.

5. Il existe un lieu appelé `eucla`. De quel continent fait-il partie ? Utilisez la commande `grep` pour le découvrir.

6. Réglez votre fuseau horaire actuel sur celui de `eucla`.

Exercices d'approfondissement

1. Quelle est la méthode préférée pour régler l'heure ? Dans quel cas la méthode préférée pourrait-elle s'avérer impossible à utiliser ?

2. Pourquoi pensez-vous qu'il existe autant de méthodes pour accomplir la même chose, c'est-à-dire régler l'heure du système ?

3. Après le 19 janvier 2038, l'heure du système Linux nécessitera un nombre de 64 bits pour être stockée. Cependant, il est possible que nous choissions simplement de définir une "nouvelle époque". Par exemple, le 1er janvier 2038 à minuit pourrait être fixé à un temps de nouvelle époque de 0. Pourquoi pensez-vous que cette solution n'a pas été retenue ?

Résumé

Voici ce que nous avons appris dans cette leçon :

- Afficher l'heure dans différents formats en ligne de commande.
- La différence entre l'horloge système et l'horloge matérielle sous Linux.
- Régler manuellement l'horloge système.
- Régler manuellement l'horloge matérielle.
- Modifier le fuseau horaire du système.

Commandes utilisées dans cette leçon :

date

Affiche ou modifie l'horloge système. Autres options :

-u

Affiche l'heure UTC.

+%s

Utilise une séquence pour afficher l'heure Epoch.

--date=

Spécifie une heure donnée à afficher, par opposition à l'heure actuelle.

--debug

Affiche les messages de débogage lors de l'analyse d'une date saisie par l'utilisateur.

-s

Règle l'horloge système manuellement.

hwclock

Affiche ou modifie l'horloge matérielle.

--systohc

Utilise l'horloge système pour régler l'horloge matérielle.

--hctosys

Utilise l'horloge matérielle pour régler l'horloge système.

--set --date

Règle l'horloge matérielle manuellement.

timedatectl

Affiche les horloges système et matérielle ainsi que la configuration NTP sur les systèmes Linux basés sur `systemd`.

set-time

Règle l'heure manuellement.

list-timezones

Affiche les fuseaux horaires disponibles.

set-timezone

Règle le fuseau horaire manuellement.

set-ntp

Active/désactive NTP.

Réponses aux exercices guidés

1. Indiquez si les commandes ci-dessous affichent ou modifient l'horloge système ou l'horloge matérielle :

Commande	Système	Matérielle	Les deux
<code>date -u</code>	X		
<code>hwclock --set --date "12:00:00"</code>		X	
<code>timedatectl</code>			X
<code>timedatectl grep RTC</code>		X	
<code>hwclock --hctosys</code>	X		
<code>date +%T -s "08:00:00"</code>	X		
<code>timedatectl set- time 1980-01-10</code>			X

2. Observez le résultat suivant, puis corrigez le format de l'argument pour que la commande fonctionne :

```
$ date --debug --date "20/20/12 0:10 -3"
```

```
date: warning: value 20 has less than 4 digits. Assuming MM/DD/YY[YY]
date: parsed date part: (Y-M-D) 0002-20-20
date: parsed time part: 00:10:00 UTC-03
date: input timezone: parsed date/time string (-03)
date: using specified time as starting value: '00:10:00'
date: error: invalid date/time value:
date:   user provided time: '(Y-M-D) 0002-20-20 00:10:00 TZ=-03'
date:   normalized time: '(Y-M-D) 0003-08-20 00:10:00 TZ=-03'
date:                                     ---- --
date:   possible reasons:
date:     numeric values overflow;
date:     incorrect timezone
date: invalid date '20/20/2 0:10 -3'
```

```
date --debug --set "12/20/20 0:10 -3"
```

3. Utilisez la commande `date` avec les séquences requises pour que le mois du système soit défini à février. Laissez le reste de la date et de l'heure inchangées.

```
date +%m -s "2"
```

4. En supposant que la commande ci-dessus ait réussi, utilisez `hwclock` pour régler l'horloge matérielle depuis l'horloge système.

```
hwclock --systohc
```

5. Il existe un lieu appelé `eucla`. De quel continent fait-il partie ? Utilisez la commande `grep` pour le découvrir. Saisissez la commande complète ci-dessous :

```
timedatectl list-timezones | grep -i eucla
```

OU

```
grep -ri eucla /usr/share/zoneinfo
```

6. Réglez votre fuseau horaire actuel sur celui de `eucla`.

```
timedatectl set-timezone 'Australia/Eucla'
```

ou

```
ln -s /usr/share/zoneinfo/Australia/Eucla /etc/localtime
```

Réponses aux exercices d'approfondissement

1. Quelle est la méthode préférée pour régler l'heure ? Dans quel cas la méthode préférée pourrait-elle s'avérer impossible à utiliser ?

Dans la plupart des distributions Linux, le protocole NTP est activé par défaut et il convient de le laisser régler l'heure système sans interférence. Cependant, si un système Linux n'est pas connecté à Internet, NTP sera inaccessible. Par exemple, un système Linux embarqué qui fonctionne sur un équipement industriel peut ne pas avoir de connectivité réseau.

2. Pourquoi pensez-vous qu'il existe autant de méthodes pour accomplir la même chose, c'est-à-dire régler l'heure du système ?

Étant donné que la mise à l'heure est une exigence de tous les systèmes *nix depuis des décennies, il existe de nombreuses méthodes traditionnelles de mise à l'heure qui sont encore maintenues.

3. Après le 19 janvier 2038, l'heure du système Linux nécessitera un nombre de 64 bits pour être stockée. Cependant, il est possible que nous choissions simplement de définir une "nouvelle époque". Par exemple, le 1er janvier 2038 à minuit pourrait être fixé à un temps de nouvelle époque de 0. Pourquoi pensez-vous que cette solution n'a pas été retenue ?

D'ici 2038, l'immense majorité des ordinateurs seront déjà équipés de processeurs 64 bits, et l'utilisation d'un nombre 64 bits ne dégradera pas les performances de manière significative. Cependant, il serait impossible d'estimer les risques de "réinitialisation" du temps d'époque de cette manière. Beaucoup de logiciels anciens pourraient être affectés. Les banques et les grandes entreprises, par exemple, disposent souvent d'un grand nombre de programmes anciens sur lesquels elles s'appuient pour leur usage interne. Ce scénario, comme tant d'autres, constitue donc une étude des compromis à faire. Tout système 32 bits fonctionnant encore en 2038 serait affecté par un dépassement d'époque, mais les logiciels existants seraient affectés par la modification de la valeur d'époque.



108.1 Leçon 2

Certification :	LPIC-1 (102)
Version :	5.0
Thème :	108 Services Systèmes Essentiels
Objectif :	108.1 Maintenir l'heure du système
Leçon :	2 sur 2

Introduction

Les ordinateurs personnels sont capables de conserver l'heure de manière raisonnablement précise, mais les environnements informatiques de production et de réseau requièrent une très grande précision de l'heure. Le temps le plus précis est mesuré par des *horloges de référence*, qui sont typiquement des horloges atomiques. Le monde moderne a conçu un système dans lequel tous les systèmes informatiques connectés à Internet peuvent être synchronisés avec ces horloges de référence grâce à ce que l'on appelle le protocole de temps réseau ou *Network Time Protocol* (NTP). Un système informatique doté du protocole NTP sera en mesure de synchroniser ses horloges système avec le temps fourni par ces horloges de référence. Si l'heure du système et l'heure mesurée sur ces serveurs sont différentes, l'ordinateur accélère ou ralentit progressivement l'heure système interne jusqu'à ce que celle-ci corresponde à l'heure du réseau.

NTP utilise une structure hiérarchique pour diffuser le temps. Les horloges de référence sont connectées à des serveurs situés au sommet de la hiérarchie. Ces serveurs sont des machines *stratum 1* et ne sont normalement pas accessibles au public. Les machines *stratum 1* sont toutefois accessibles aux machines *stratum 2*, qui sont accessibles aux machines *stratum 3*, et ainsi de suite. Les serveurs *stratum 2* sont accessibles au public, de même que toutes les machines situées plus bas dans la hiérarchie. Lors de la mise en place du protocole NTP pour un réseau important, une

bonne pratique consiste à faire en sorte qu'un petit nombre d'ordinateurs se connectent aux serveurs *stratum* 2+ et que ces machines fournissent le protocole NTP à toutes les autres machines. De cette manière, il est possible de minimiser les demandes sur les machines *stratum* 2.

Il existe un certain nombre de termes importants qui reviennent régulièrement lorsqu'il est question de NTP. Quelques-uns de ces termes sont utilisés dans les commandes que nous utiliserons pour suivre l'état de NTP sur nos machines :

Offset

Il s'agit de la différence absolue entre l'heure système et l'heure NTP. Par exemple, si l'horloge système indique 12:00:02 et l'heure NTP 11:59:58, le décalage (*offset*) entre les deux horloges sera de quatre secondes.

Step

Si le décalage du temps entre le serveur NTP et un client est supérieur à 128 ms, NTP va procéder à un seul changement significatif de l'heure système, au lieu de ralentir ou d'accélérer l'heure système. C'est ce qu'on appelle le *stepping*.

Slew

Le *slewing* fait référence aux changements apportés à l'heure système lorsque le décalage entre l'heure système et celle de NTP est inférieur à 128 ms. Dans ce cas, les changements s'effectuent progressivement. On parle alors de *slewing*.

Insane Time

Si le décalage entre l'heure système et l'heure NTP est supérieur à 17 minutes, l'heure système est considérée comme *insane* (insensée) et le démon NTP n'introduira aucun changement à l'heure système. Des mesures spéciales devront être prises pour ramener l'heure système à moins de 17 minutes de l'heure correcte.

Drift

Le *drift* (dérive) désigne le phénomène par lequel deux horloges se désynchronisent au fil du temps. En fait, si deux horloges sont initialement synchronisées, mais qu'elles se désynchronisent avec le temps, il y a une dérive de l'horloge.

Jitter

Le *jitter* (gigue) fait référence à la quantité de *drift* (dérive) depuis la dernière fois qu'une horloge a été sollicitée. Si la dernière synchronisation NTP a eu lieu il y a 17 minutes et que le décalage entre le serveur NTP et le client est de 3 millisecondes, le *jitter* est de 3 millisecondes.

Nous allons maintenant passer en revue quelques-unes des méthodes spécifiques utilisées par Linux pour mettre en œuvre le protocole NTP.

timedatectl

Si votre distribution Linux utilise `timedatectl`, elle implémente par défaut un client *SNTP* plutôt qu'une implémentation NTP complète. Il s'agit d'une implémentation moins complexe de l'heure réseau et cela signifie que votre machine ne va pas servir du NTP à d'autres ordinateurs connectés.

Dans ce cas, *SNTP* ne va pas fonctionner si le service `timesyncd` n'est pas en cours d'exécution. Comme pour tous les services `systemd`, nous pouvons vérifier qu'il est lancé avec :

```
$ systemctl status systemd-timesyncd
• systemd-timesyncd.service - Network Time Synchronization
   Loaded: loaded (/lib/systemd/system/systemd-timesyncd.service; enabled; vendor preset:
   enabled)
   Drop-In: /lib/systemd/system/systemd-timesyncd.service.d
            └─disable-with-time-daemon.conf
   Active: active (running) since Thu 2020-01-09 21:01:50 EST; 2 weeks 1 days ago
     Docs: man:systemd-timesyncd.service(8)
  Main PID: 1032 (systemd-timesyn)
    Status: "Synchronized to time server for the first time 91.189.89.198:123
(ntp.ubuntu.com)."
```

```
   Tasks: 2 (limit: 4915)
  Memory: 3.0M
    CGroup: /system.slice/systemd-timesyncd.service
            └─1032 /lib/systemd/systemd-timesyncd

Jan 11 13:06:18 NeoMx systemd-timesyncd[1032]: Synchronized to time server for the first
time 91.189.91.157:123 (ntp.ubuntu.com).
...
```

L'état de la synchronisation *SNTP* de `timedatectl` peut être vérifié en utilisant `show-timesync` :

```
$ timedatectl show-timesync --all
LinkNTPServers=
SystemNTPServers=
FallbackNTPServers=ntp.ubuntu.com
ServerName=ntp.ubuntu.com
ServerAddress=91.189.89.198
RootDistanceMaxUsec=5s
PollIntervalMinUsec=32s
PollIntervalMaxUsec=34min 8s
PollIntervalUsec=34min 8s
```

```
NTPMessage={ Leap=0, Version=4, Mode=4, Stratum=2, Precision=-23, RootDelay=8.270ms,
RootDispersion=18.432ms, Reference=91EECB0E, OriginateTimestamp=Sat 2020-01-25 18:35:49 EST,
ReceiveTimestamp=Sat 2020-01-25 18:35:49 EST, TransmitTimestamp=Sat 2020-01-25 18:35:49 EST,
DestinationTimestamp=Sat 2020-01-25 18:35:49 EST, Ignored=no PacketCount=263, Jitter=2.751ms
}
Frequency=-211336
```

Cette configuration peut suffire dans la plupart des cas, mais comme nous l'avons déjà dit, elle sera inadaptée si l'on souhaite synchroniser plusieurs clients dans un réseau. Dans ce cas, il est recommandé d'installer un client NTP complet.

Le service NTP

L'heure système est comparée à l'heure du réseau à intervalles réguliers. Pour que cela fonctionne, il faut qu'un démon (*daemon*) tourne en arrière-plan. Pour de nombreux systèmes Linux, le nom de ce démon est `ntpd`. `ntpd` permet à une machine d'être non seulement un *consommateur de temps* (c'est-à-dire capable de synchroniser sa propre horloge avec une source extérieure), mais aussi de *fournir* du temps à d'autres machines.

Admettons que notre ordinateur soit basé sur `systemd` et qu'il utilise `systemctl` pour contrôler les démons. Nous allons installer les paquets `ntp` en utilisant le gestionnaire de paquets approprié, puis nous assurer que notre démon `ntpd` fonctionne en vérifiant son état :

```
$ systemctl status ntpd
```

```
• ntpd.service - Network Time Service
  Loaded: loaded (/usr/lib/systemd/system/ntpd.service; enabled; vendor preset: disabled)
  Active: active (running) since Fri 2019-12-06 03:27:21 EST; 7h ago
  Process: 856 ExecStart=/usr/sbin/ntpd -u ntp:ntp $OPTIONS (code=exited, status=0/SUCCESS)
  Main PID: 867 (ntpd)
  CGroup: /system.slice/ntpd.service
          └─867 /usr/sbin/ntpd -u ntp:ntp -g
```

Dans certains cas, il peut être nécessaire de démarrer et d'activer `ntpd`. Sur la plupart des machines Linux, cela se fait avec :

```
# systemctl enable ntpd && systemctl start ntpd
```

Les requêtes NTP sont effectuées sur le port TCP 123. Si NTP échoue, assurez-vous que ce port est ouvert et en écoute.

Configuration NTP

NTP est capable d'interroger plusieurs sources et de sélectionner les meilleurs candidats à utiliser pour régler l'heure système. En cas de perte de connexion réseau, NTP utilise les ajustements précédents dans son historique pour évaluer les ajustements à venir.

Selon votre distribution Linux, la liste des serveurs de temps du réseau sera stockée à un endroit différent. Admettons que `ntp` soit installé sur votre machine.

Le fichier `/etc/ntp.conf` contient des informations de configuration sur la façon dont votre système se synchronise avec l'heure du réseau. Ce fichier peut être lu et modifié en utilisant `vi` ou `nano`.

Par défaut, les serveurs NTP utilisés seront spécifiés dans une section comme celle-ci :

```
# Use public servers from the pool.ntp.org project.
# Please consider joining the pool (http://www.pool.ntp.org/join.html).
server 0.centos.pool.ntp.org iburst
server 1.centos.pool.ntp.org iburst
server 2.centos.pool.ntp.org iburst
server 3.centos.pool.ntp.org iburst
```

Voici la syntaxe pour ajouter des serveurs NTP :

```
server (IP Address)
server server.url.localhost
```

Les adresses de serveurs peuvent être des adresses IP ou des URL si le DNS a été correctement configuré. Dans ce cas, le serveur sera toujours interrogé.

Un administrateur réseau peut également envisager d'utiliser (ou de mettre en place) un *pool*. Dans ce cas, nous supposons qu'il existe plusieurs fournisseurs NTP, qui exécutent tous des démons NTP et qui sont réglés à la même heure. Lorsqu'un client interroge un *pool*, un fournisseur est sélectionné au hasard. Cela permet de répartir la charge du réseau entre plusieurs machines, de manière à ce qu'aucune machine du *pool* ne traite toutes les requêtes NTP.

De manière générale, `/etc/ntp.conf` sera rempli avec un *pool* de serveurs appelé `pool.ntp.org`. Par exemple, `server 0.centos.pool.ntp.org` est un pool NTP par défaut fourni aux machines CentOS.

pool.ntp.org

Les serveurs NTP utilisés par défaut sont un projet open source. De plus amples informations sont disponibles à l'adresse suivante : ntp.pool.org.

Réfléchissez à la pertinence du pool NTP pour votre utilisation. Si votre entreprise, votre organisation ou la vie humaine dépendent de l'exactitude de l'heure ou peuvent être affectées par une heure erronée, vous ne devriez pas vous contenter de la trouver sur Internet. Le pool NTP est généralement de très bonne qualité, mais il s'agit d'un service géré par des bénévoles pendant leur temps libre. Veuillez vous adresser à vos fournisseurs d'équipement et de services pour obtenir un service local et fiable. Consultez également nos conditions d'utilisation. Nous recommandons les serveurs de temps de Meinberg, mais vous pouvez également trouver des serveurs de temps de End Run, Spectracom et bien d'autres.

— ntp.pool.org

ntpd

Lors de la configuration initiale, l'heure système et l'heure NTP peuvent être sérieusement désynchronisées. Si le décalage entre l'heure système et l'heure NTP est supérieur à 17 minutes, le démon NTP ne modifiera pas l'heure système. Dans ce cas, une intervention manuelle sera nécessaire.

Tout d'abord, si `ntpd` est en cours d'exécution, il faudra *arrêter* le service. Utilisez `systemctl stop ntpd` pour ce faire.

Ensuite, utilisez `ntpdate pool.ntp.org` pour effectuer une première synchronisation ponctuelle, où `pool.ntp.org` fait référence à l'adresse IP ou à l'URL d'un serveur NTP. Plusieurs synchronisations sont parfois nécessaires.

ntpq

`ntpq` est un outil qui permet de surveiller l'état de NTP. Une fois que le démon NTP a été démarré et configuré, `ntpq` peut être utilisé pour vérifier son état :

```
$ ntpq -p
      remote           refid      st t when poll reach   delay   offset  jitter
=====
+37.44.185.42      91.189.94.4        3 u   86  128  377  126.509  -20.398   6.838
+ntp2.0x00.lv     193.204.114.233    2 u   82  128  377  143.885   -8.105   8.478
*inspektor-vlan1 121.131.112.137    2 u   17  128  377  112.878  -23.619   7.959
```

```
b1-66er.matrix. 18.26.4.105      2 u 484 128 10 34.907 -0.811 16.123
```

Dans ce cas, `-p` correspond à *print* et affiche un résumé des serveurs. Les adresses des hôtes peuvent également être retournées sous forme d'adresses IP en utilisant `-n`.

remote

nom d'hôte du fournisseur NTP.

refid

ID de référence du fournisseur NTP.

st

Stratum du fournisseur.

when

Nombre de secondes depuis la dernière requête.

poll

Nombre de secondes entre les requêtes.

reach

ID d'état indiquant si un serveur a été atteint. Les connexions réussies vont augmenter ce nombre de 1.

delay

Temps en ms entre la requête et la réponse du serveur.

offset

Temps en ms entre l'heure système et l'heure NTP.

jitter

Décalage en ms entre l'heure système et l'heure NTP lors de la dernière requête.

`ntpq` a également un mode interactif, auquel on accède lorsqu'il est exécuté sans option ni argument. L'option `?` retournera une liste de commandes que `ntpq` reconnaîtra.

chrony

`chrony` est une autre façon d'implémenter NTP. Il est installé par défaut sur certains systèmes Linux et peut être téléchargé sur toutes les distributions majeures. `chronyd` correspond au démon

chrony, et chronyc est l'interface en ligne de commande. Il faut parfois démarrer et d'activer chronyd avant d'interagir avec chronyc.

Si l'installation de chrony a une configuration par défaut, l'utilisation de la commande chronyc tracking fournira des informations sur l'heure NTP et l'heure système :

```
$ chronyc tracking
Reference ID      : 3265FB3D (bras-vprn-toroon2638w-lp130-11-50-101-251-61.dsl.)
Stratum          : 3
Ref time (UTC)   : Thu Jan 09 19:18:35 2020
System time      : 0.000134029 seconds fast of NTP time
Last offset      : +0.000166506 seconds
RMS offset       : 0.000470712 seconds
Frequency        : 919.818 ppm slow
Residual freq    : +0.078 ppm
Skew             : 0.555 ppm
Root delay       : 0.006151616 seconds
Root dispersion  : 0.010947504 seconds
Update interval  : 129.8 seconds
Leap status      : Normal
```

Ce résultat contient beaucoup d'informations, plus que ce qui est disponible dans d'autres implémentations.

Reference ID

L'ID et le nom de la référence avec laquelle l'ordinateur est actuellement synchronisé.

Stratum

Nombre de sauts vers un ordinateur doté d'une horloge de référence.

Ref time

L'heure UTC à laquelle la dernière mesure de la source de référence a été effectuée.

System time

Retard de l'horloge système par rapport au serveur synchronisé.

Last offset

Estimation du décalage de la dernière mise à jour de l'horloge.

RMS offset

Moyenne à long terme de la valeur du décalage.

Frequency

C'est la vitesse à laquelle l'horloge système se tromperait si chronyd ne la corrigeait pas. Elle est exprimée en ppm (parties par million).

Residual freq

Fréquence résiduelle indiquant la différence entre les mesures de la source de référence et la fréquence actuellement utilisée.

Skew

Limite d'erreur estimée de la fréquence.

Root delay

Total des délais du réseau jusqu'à l'ordinateur stratum, à partir duquel l'ordinateur est synchronisé.

Leap status

Il s'agit de l'état du saut qui peut prendre l'une des valeurs suivantes : normal, insertion d'une seconde, suppression d'une seconde ou non synchronisé.

Nous pouvons également consulter les informations détaillées concernant la dernière mise à jour NTP valide :

```
# chrony ntpdata
Remote address : 172.105.97.111 (AC69616F)
Remote port    : 123
Local address  : 192.168.122.81 (C0A87A51)
Leap status    : Normal
Version        : 4
Mode           : Server
Stratum        : 2
Poll interval  : 6 (64 seconds)
Precision      : -25 (0.000000030 seconds)
Root delay     : 0.000381 seconds
Root dispersion : 0.000092 seconds
Reference ID    : 61B7CE58 ()
Reference time  : Mon Jan 13 21:50:03 2020
Offset         : +0.000491960 seconds
Peer delay     : 0.004312567 seconds
Peer dispersion : 0.000000068 seconds
Response time  : 0.000037078 seconds
Jitter asymmetry: +0.00
NTP tests      : 111 111 1111
```

```

Interleaved      : No
Authenticated    : No
TX timestamping  : Daemon
RX timestamping  : Kernel
Total TX         : 15
Total RX         : 15
Total valid RX   : 15

```

Enfin, `chronyc sources` renvoie des informations sur les serveurs NTP utilisés pour synchroniser le temps :

```

$ chronyc sources
210 Number of sources = 0
MS Name/IP address      Stratum Poll Reach LastRx Last sample
=====

```

Pour l'instant, cette machine n'a pas de sources configurées. Nous pouvons ajouter des sources à partir de `pool.ntp.org` en ouvrant le fichier de configuration de `chrony`. Celui-ci est généralement situé dans `/etc/chrony.conf`. Lorsque nous ouvrons ce fichier, nous devrions voir que certains serveurs sont répertoriés par défaut :

```

210 Number of sources = 0
MS Name/IP address      Stratum Poll Reach LastRx Last sample
=====
# Most computers using chrony will send measurement requests to one or
# more 'NTP servers'. You will probably find that your Internet Service
# Provider or company have one or more NTP servers that you can specify.
# Failing that, there are a lot of public NTP servers. There is a list
# you can access at http://support.ntp.org/bin/view/Servers/WebHome or
# you can use servers from the 3.arch.pool.ntp.org project.

! server 0.arch.pool.ntp.org iburst iburst ! server 1.arch.pool.ntp.org iburst iburst !
server 2.arch.pool.ntp.org iburst iburst

! pool 3.arch.pool.ntp.org iburst

```

Ces serveurs serviront également de guide syntaxique lorsque nous ajouterons nos propres serveurs. Dans ce cas, nous supprimerons simplement les `!` au début de chaque ligne afin d'utiliser les serveurs par défaut du projet `pool.ntp.org`.

En outre, dans ce fichier, nous pouvons choisir de modifier la configuration par défaut concernant

la dérive et le décalage, ainsi que l'emplacement du fichier de dérive et du fichier clé.

Sur cette machine, nous devons effectuer une correction initiale importante de l'horloge. Nous choisirons de décommenter la ligne suivante :

```
! makestep 1.0 3
```

Après avoir modifié le fichier de configuration, redémarrez le service `chronyd` et utilisez ensuite `chronyc makestep` pour modifier manuellement l'horloge système :

```
# chronyc makestep
200 OK
```

Puis utilisez `chronyc tracking` comme précédemment pour vérifier que les changements ont bien été pris en compte.

Exercices guidés

1. Indiquez le terme approprié pour chaque définition :

Définition	Terme
Un ordinateur qui partage l'heure réseau avec vous	
La distance par rapport à une horloge de référence, en sauts ou en pas	
La différence entre l'heure système et l'heure réseau	
La différence entre l'heure système et l'heure réseau depuis la dernière requête NTP	
Groupe de serveurs qui fournissent l'heure réseau en répartissant la charge entre eux.	

2. Indiquez les commandes que vous utiliseriez pour afficher les valeurs suivantes :

Valeur	<code>chronyc tracking</code>	<code>timedatectl show-timesync --all</code>	<code>ntpq -pn</code>	<code>chrony ntpdata</code>	<code>chronyc sources</code>
Gigue (<i>jitter</i>)					
Dérive (<i>drift</i>)					
Intervalle des requêtes					
Décalage (<i>offset</i>)					
Strate (<i>stratum</i>)					
Adresse IP du fournisseur					
Retard (<i>root delay</i>)					

3. Vous mettez en place un réseau d'entreprise constitué d'un serveur Linux et de plusieurs ordinateurs de bureau Linux. Le serveur a une adresse IP statique de 192.168.0.101. Vous décidez que le serveur se connectera à `pool.ntp.org` et fournira ensuite l'heure NTP aux ordinateurs. Décrivez la configuration du serveur et des ordinateurs de bureau.

4. Une machine Linux a une heure incorrecte. Décrivez les étapes à suivre pour corriger les erreurs NTP.

Exercices d'approfondissement

1. Étudiez les différences entre SNTP et NTP.

SNTP	NTP

2. Pourquoi un administrateur système pourrait-il décider de ne pas utiliser `pool.ntp.org` ?

3. Comment un administrateur système choisirait-il de rejoindre ou de contribuer au projet `pool.ntp.org` ?

Résumé

Voici ce que nous avons appris dans cette leçon :

- NTP : qu'est-ce que c'est et pourquoi c'est important.
- Configurer le démon NTP à partir du projet `pool.ntp.org`.
- Utiliser `ntpq` pour vérifier la configuration NTP.
- Utiliser `chrony` comme service NTP alternatif.

Commandes utilisées dans cette leçon :

`timedatectl show-timesync --all`

Affiche les informations SNTP si vous utilisez `timedatectl`.

`ntpdate <address>`

Effectue une mise à jour manuelle ponctuelle de NTP.

`ntpq -p`

Affiche un historique des dernières requêtes NTP. `-n` va remplacer les URLs par des adresses IP.

`chronyc tracking`

Affiche l'état de NTP si `chrony` est utilisé.

`chronyc ntpdata`

Affiche les informations NTP relatives à la dernière requête.

`chronyc sources`

Affiche les informations sur les fournisseurs NTP.

`chronyc makestep`

Effectue une mise à jour manuelle ponctuelle de NTP si `chrony` est utilisé.

Réponses aux exercices guidés

1. Indiquez le terme approprié pour chaque définition :

Définition	Terme
Un ordinateur qui partage l'heure réseau avec vous	Fournisseur (<i>provider</i>)
La distance par rapport à une horloge de référence, en sauts ou en pas	Strate (<i>stratum</i>)
La différence entre l'heure système et l'heure réseau	Décalage (<i>offset</i>)
La différence entre l'heure système et l'heure réseau depuis la dernière requête NTP	Gigue (<i>jitter</i>)
Groupe de serveurs qui fournissent l'heure réseau en répartissant la charge entre eux.	Pool

2. Indiquez les commandes que vous utiliseriez pour afficher les valeurs suivantes :

Valeur	<code>chronyc tracking</code>	<code>timedatectl show-timesync --all</code>	<code>ntpq -pn</code>	<code>chrony ntpdata</code>	<code>chronyc sources</code>
Gigue (<i>jitter</i>)		X	X		
Dérive (<i>drift</i>)					
Intervalle des requêtes	X	X	X (colonne when)	X	X
Décalage (<i>offset</i>)	X		X	X	
Strate (<i>stratum</i>)	X	X	X	X	X
Adresse IP du fournisseur		X	X	X	X
Retard (<i>root delay</i>)	X			X	

3. Vous mettez en place un réseau d'entreprise constitué d'un serveur Linux et de plusieurs ordinateurs de bureau Linux. Le serveur a une adresse IP statique de 192.168.0.101. Vous décidez que le serveur se connectera à `pool.ntp.org` et fournira ensuite l'heure NTP aux ordinateurs. Décrivez la configuration du serveur et des ordinateurs de bureau.

Assurez-vous que le serveur dispose d'un service `ntpd` plutôt que `SNTP`. Utilisez les `pools.pool.ntp.org` dans le fichier `/etc/ntp.conf` ou `/etc/chrony.conf`. Pour chaque client, spécifiez `192.168.0.101` dans chaque fichier `/etc/ntp.conf` ou `/etc/chrony.conf`.

4. Une machine Linux a une heure incorrecte. Décrivez les étapes à suivre pour corriger les erreurs NTP.

Tout d'abord, assurez-vous que la machine est connectée à Internet. Utilisez `ping` pour cela. Vérifiez qu'un service `ntpd` ou `SNTP` est en cours d'exécution en utilisant `systemctl status ntpd` ou `systemctl status systemd-timesyncd`. Vous pouvez voir des messages d'erreur qui fournissent des informations utiles. Enfin, utilisez une commande telle que `ntpq -p` ou `chrony tracking` pour vérifier si des requêtes ont été faites. Si l'heure système est radicalement différente de l'heure réseau, il se peut que l'heure système soit considérée comme "insensée" (*insane*) et qu'elle ne soit pas modifiée sans intervention manuelle. Dans ce cas, utilisez une commande de la leçon précédente ou une commande telle que `ntpdate pool.ntp.org` pour effectuer une synchronisation NTP ponctuelle.

Réponses aux exercices d'approfondissement

1. Étudiez les différences entre SNTP et NTP.

SNTP	NTP
moins précis	plus précis
exige moins de ressources	exige plus de ressources
ne peut pas être un fournisseur de temps	peut être un fournisseur de temps
correction par étapes	correction par étapes et incrémentale
interroge une source unique pour l'heure	peut gérer plusieurs serveurs NTP et utiliser le fournisseur optimal

2. Pourquoi un administrateur système pourrait-il décider de ne pas utiliser `pool.ntp.org` ?

D'après ntppool.org : S'il est absolument crucial d'avoir l'heure exacte, vous devriez envisager une autre solution. De même, si votre fournisseur d'accès Internet dispose d'un serveur de temps, il est recommandé de l'utiliser.

3. Comment un administrateur système choisirait-il de rejoindre ou de contribuer au projet `pool.ntp.org` ?

D'après www.ntppool.org : Votre serveur doit avoir une adresse IP statique et une connexion Internet permanente. L'adresse IP statique ne doit pas changer du tout ou au maximum moins d'une fois par an. En outre, les exigences en matière de bande passante sont modestes : 384 à 512 Kbit de bande passante. Les serveurs de strate 3 ou 4 sont les bienvenus.



108.2 Journaux systèmes

Référence aux objectifs de LPI

LPIC-1 version 5.0, Exam 102, Objective 108.2

Valeur

4

Domaines de connaissance les plus importants

- Configuration de base de rsyslog.
- Compréhension des sous-systèmes (facilities), priorités et actions standards.
- Interrogation du journal systemd.
- Filtrage des données du journal systemd sur des critères comme la date, le service ou la priorité
- Configuration du stockage persistant du journal systemd et de la taille du journal
- Suppression des anciennes informations du journal systemd
- Récupération du journal systemd à partir d'une sauvegarde ou d'une copie
- Compréhension des interactions entre rsyslog et systemd-journald
- Configuration de logrotate.
- Connaissance de base de syslog et syslog-ng.

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

- `/etc/rsyslog.conf`
- `/var/log/`
- `logger`
- `logrotate`

- `/etc/logrotate.conf`
- `/etc/logrotate.d/`
- `journalctl`
- `systemd-cat`
- `/etc/systemd/journald.conf`
- `/var/log/journal/`



108.2 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	108 Services Systèmes Essentiels
Objectif :	108.2 Journalisation du système
Leçon :	1 sur 2

Introduction

Les journaux (*logs*) peuvent être le meilleur amis de l'administrateur système. Les journaux sont des fichiers (généralement des fichiers texte) dans lesquels tous les événements du système et du réseau sont enregistrés par ordre chronologique à partir du moment où votre système est démarré. Ainsi, la panoplie d'informations que l'on peut trouver dans les journaux comprend à peu près tous les aspects du système : tentatives d'authentification échouées, erreurs des programmes et des services, hôtes bloqués par le pare-feu, etc. Comme vous pouvez bien l'imaginer, les journaux facilitent considérablement la vie des administrateurs système en ce qui concerne le dépannage, le contrôle des ressources, la détection de comportements anormaux des programmes, et ainsi de suite.

Dans cette leçon, nous allons aborder l'une des applications de journalisation les plus courantes que l'on peut trouver dans les distributions GNU/Linux : `rsyslog`. Nous verrons les différents types de journaux qui existent, l'endroit où ils sont stockés, les informations qu'ils contiennent et comment celles-ci peuvent être récupérées et filtrées. Nous parlerons également de la manière dont les journaux peuvent être conservés sur des serveurs centralisés à travers des réseaux IP, de la rotation des journaux et de la mémoire tampon circulaire du noyau (*kernel ring buffer*).

Journalisation du système

Dès que le noyau et les différents processus de votre système commencent à se lancer et à communiquer les uns avec les autres, ils génèrent un grand nombre d'informations sous forme de messages qui sont — pour la plupart — envoyés vers les journaux.

Sans la journalisation, la recherche d'un événement survenu sur un serveur donnerait du fil à retordre aux administrateurs système, d'où l'importance de disposer d'un moyen standardisé et centralisé pour conserver la trace de tout ce qui se passe sur le système. Les journaux sont déterminants et révélateurs en matière de dépannage et de sécurité et constituent des sources de données fiables pour comprendre les statistiques du système et de prévoir les tendances.

En dehors de `systemd-journald` (dont nous parlerons dans la prochaine leçon), la journalisation a toujours été gérée par trois grands services dédiés : `syslog`, `syslog-ng` (*syslog new generation*) et `rsyslog` (*the rocket-fast system for log processing*). `rsyslog` a apporté d'importantes améliorations (comme le support de RELP) et constitue aujourd'hui le choix le plus populaire. Chacun de ces services collecte des messages provenant d'autres services et programmes et les stocke dans des fichiers de journalisation, typiquement dans `/var/log`. Cependant, certains services s'occupent de leurs propres journaux (comme par exemple le serveur web Apache HTTPD ou le système d'impression CUPS). De même, le noyau Linux utilise une mémoire tampon circulaire (*ring buffer*) pour stocker ses messages de journalisation.

NOTE

RELP (*Reliable Event Logging Protocol*) signifie protocole de journalisation fiable des événements et étend la fonctionnalité du protocole `syslog` pour assurer une transmission fiable des messages.

Étant donné que `rsyslog` est devenu l'outil de journalisation standard *de facto* dans toutes les grandes distributions, nous nous concentrerons sur lui pour cette leçon. `rsyslog` est basé sur un modèle client-serveur. Le client et le serveur peuvent tourner sur le même hôte ou sur des machines différentes. Les messages sont envoyés et reçus dans un format particulier et peuvent être conservés sur des serveurs `rsyslog` centralisés à travers des réseaux IP. Le démon de `rsyslog` — `rsyslogd` — travaille avec `klogd` (qui gère les messages du noyau). Dans les sections ci-dessous, nous aborderons `rsyslog` et son infrastructure de journalisation.

NOTE

Un démon est un service qui s'exécute en arrière-plan. Notez le `d` final dans les noms des démons : `klogd` ou `rsyslogd`.

Types de journaux

Les journaux étant des données *variables*, on les trouve normalement dans `/var/log`. En gros, on distingue les *journaux système* et les *journaux des services ou des programmes*.

Voyons quelques journaux système et les informations qu'ils renferment :

/var/log/auth.log

Activités liées aux processus d'authentification : utilisateurs connectés, informations `sudo`, tâches cron, tentatives de connexion échouées, etc.

/var/log/syslog

Un fichier centralisé pour pratiquement tous les journaux récupérés par `rsyslogd`. Étant donné qu'il contient énormément d'informations, les journaux sont répartis sur d'autres fichiers en fonction de la configuration fournie dans `/etc/rsyslog.conf`.

/var/log/debug

Informations de débogage des programmes.

/var/log/kern.log

Messages du noyau.

/var/log/messages

Messages informatifs qui ne sont pas liés au noyau mais à d'autres services. C'est également la destination par défaut des journaux des clients distants dans le cas de l'implémentation d'un serveur de journaux centralisé.

/var/log/daemon.log

Informations relatives aux démons ou aux services qui tournent en arrière-plan.

/var/log/mail.log

Informations relatives au serveur de messagerie comme Postfix.

/var/log/Xorg.0.log

Informations relatives à la carte graphique.

/var/run/utmp et /var/log/wtmp

Connexions réussies.

/var/log/btmp

Tentatives de connexion échouées comme les attaques en force brute via SSH.

/var/log/faillog

Tentatives d'authentification échouées.

/var/log/lastlog

Date et heure des dernières connexions des utilisateurs.

Voyons maintenant quelques exemples de journaux de services :

/var/log/cups/

Répertoire des journaux du système d'impression CUPS (*Common Unix Printing System*). Il contient généralement les fichiers de journalisation par défaut suivants : `error_log`, `page_log` et `access_log`.

/var/log/apache2/ ou /var/log/httpd

Répertoire des journaux du serveur web Apache. Il contient généralement les fichiers de journalisation par défaut suivants : `access.log`, `error_log` et `other_vhosts_access.log`.

/var/log/mysql

Répertoire des journaux du système de gestion de bases de données relationnelles MySQL. Il contient généralement les fichiers de journalisation par défaut suivants : `error_log`, `mysql.log` et `mysql-slow.log`.

/var/log/samba/

Répertoire pour les journaux du protocole *Server Message Block* (SMB). Il contient généralement les fichiers de journalisation par défaut suivants : `log.`, `log.nmbd` et `log.smbd`.

NOTE

Le nom exact et le contenu des fichiers de journalisation peuvent varier d'une distribution Linux à une autre. Il existe également des journaux spécifiques à certaines distributions, comme `/var/log/dpkg.log` (contenant les informations relatives aux paquets dpkg) dans Debian GNU/Linux et ses dérivées.

Lire les fichier de journalisation

Pour lire les fichiers de journalisation, assurez-vous d'abord d'être le root ou d'avoir les droits de lecture sur le fichier. Vous pouvez utiliser divers outils comme :

less ou more

Commandes permettent de visualiser et de faire défiler page par page :

```
root@debian:~# less /var/log/auth.log
Sep 12 18:47:56 debian sshd[441]: Received SIGHUP; restarting.
Sep 12 18:47:56 debian sshd[441]: Server listening on 0.0.0.0 port 22.
Sep 12 18:47:56 debian sshd[441]: Server listening on :: port 22.
Sep 12 18:47:56 debian sshd[441]: Received SIGHUP; restarting.
```

```
Sep 12 18:47:56 debian sshd[441]: Server listening on 0.0.0.0 port 22.
Sep 12 18:47:56 debian sshd[441]: Server listening on :: port 22.
Sep 12 18:49:46 debian sshd[905]: Accepted password for carol from 192.168.1.65 port
44296 ssh2
Sep 12 18:49:46 debian sshd[905]: pam_unix(sshd:session): session opened for user carol
by (uid=0)
Sep 12 18:49:46 debian systemd-logind[331]: New session 2 of user carol.
Sep 12 18:49:46 debian systemd: pam_unix(systemd-user:session): session opened for user
carol by (uid=0)
(...)
```

zless ou zmore

La même chose que `less` et `more`, mais utilisés avec les journaux compressés avec `gzip` (une fonction commune de `logrotate`):

```
root@debian:~# zless /var/log/auth.log.3.gz
Aug 19 20:05:57 debian sudo:    carol : TTY=pts/0 ; PWD=/home/carol ; USER=root ;
COMMAND=/sbin/shutdown -h now
Aug 19 20:05:57 debian sudo: pam_unix(sudo:session): session opened for user root by
carol(uid=0)
Aug 19 20:05:57 debian lightdm: pam_unix(lightdm-greeter:session): session closed for
user lightdm
Aug 19 23:50:49 debian systemd-logind[333]: Watching system buttons on /dev/input/event2
(Power Button)
Aug 19 23:50:49 debian systemd-logind[333]: Watching system buttons on /dev/input/event3
(Sleep Button)
Aug 19 23:50:49 debian systemd-logind[333]: Watching system buttons on /dev/input/event4
(Video Bus)
Aug 19 23:50:49 debian systemd-logind[333]: New seat seat0.
Aug 19 23:50:49 debian sshd[409]: Server listening on 0.0.0.0 port 22.
(...)
```

tail

Affiche les dernières lignes d'un fichier (10 lignes par défaut). La puissance de `tail` réside — en grande partie — dans l'option `-f` qui affiche dynamiquement les nouvelles lignes au fur et à mesure qu'elles sont ajoutées :

```
root@suse-server:~# tail -f /var/log/messages
2019-09-14T13:57:28.962780+02:00 suse-server sudo: pam_unix(sudo:session): session closed
for user root
2019-09-14T13:57:38.038298+02:00 suse-server sudo:    carol : TTY=pts/0 ;
```

```
PWD=/home/carol ; USER=root ; COMMAND=/usr/bin/tail -f /var/log/messages
2019-09-14T13:57:38.039927+02:00 suse-server sudo: pam_unix(sudo:session): session opened
for user root by carol(uid=0)
2019-09-14T14:07:22+02:00 debian carol: appending new message from client to remote
server...
```

head

Affiche les premières lignes d'un fichier (10 lignes par défaut) :

```
root@suse-server:~# head -5 /var/log/mail
2019-06-29T11:47:59.219806+02:00 suse-server postfix/postfix-script[1732]: the Postfix
mail system is not running
2019-06-29T11:48:01.355361+02:00 suse-server postfix/postfix-script[1925]: starting the
Postfix mail system
2019-06-29T11:48:01.391128+02:00 suse-server postfix/master[1930]: daemon started --
version 3.3.1, configuration /etc/postfix
2019-06-29T11:55:39.247462+02:00 suse-server postfix/postfix-script[3364]: stopping the
Postfix mail system
2019-06-29T11:55:39.249375+02:00 suse-server postfix/master[1930]: terminating on signal
15
```

grep

Outil de filtrage qui permet de rechercher des chaînes de caractères données :

```
root@debian:~# grep "dhclient" /var/log/syslog
Sep 13 11:58:48 debian dhclient[448]: DHCPREQUEST of 192.168.1.4 on enp0s3 to 192.168.1.1
port 67
Sep 13 11:58:49 debian dhclient[448]: DHCPACK of 192.168.1.4 from 192.168.1.1
Sep 13 11:58:49 debian dhclient[448]: bound to 192.168.1.4 -- renewal in 1368 seconds.
(...)
```

Comme vous l'aurez remarqué, le résultat s'affiche dans le format suivant :

- Horodatage
- Nom d'hôte à l'origine du message
- Nom du programme/service qui a généré le message
- Le PID du programme qui a généré le message
- Description de l'action qui a eu lieu

Dans certains cas de figure, les fichiers de journalisation ne sont pas du texte, mais des fichiers binaires et — par conséquent — il vous faut utiliser des commandes spécifiques pour les analyser :

/var/log/wtmp

Utilisez `who` (ou `w`):

```
root@debian:~# who
root    pts/0      2020-09-14 13:05 (192.168.1.75)
root    pts/1      2020-09-14 13:43 (192.168.1.75)
```

/var/log/btmp

Utilisez `utmpdump` ou `last -f`:

```
root@debian:~# utmpdump /var/log/btmp
Utmp dump of /var/log/btmp
[6] [01287] [   ] [dave   ] [ssh:notty   ] [192.168.1.75   ] [192.168.1.75   ]
[2019-09-07T19:33:32,000000+0000]
```

/var/log/faillog

Utilisez `faillog`:

```
root@debian:~# faillog -a | less
Login      Failures Maximum Latest           On

root        0          0  01/01/70 01:00:00 +0100
daemon      0          0  01/01/70 01:00:00 +0100
bin         0          0  01/01/70 01:00:00 +0100
sys         0          0  01/01/70 01:00:00 +0100
sync        0          0  01/01/70 01:00:00 +0100
games       0          0  01/01/70 01:00:00 +0100
man         0          0  01/01/70 01:00:00 +0100
lp          0          0  01/01/70 01:00:00 +0100
mail        0          0  01/01/70 01:00:00 +0100
(...)
```

/var/log/lastlog

Utilisez `lastlog`:

```
root@debian:~# lastlog | less
```

Username	Port	From	Latest
root			Never logged in
daemon			Never logged in
bin			Never logged in
sys			Never logged in
(...)			
sync			Never logged in
avahi			Never logged in
colord			Never logged in
saned			Never logged in
hplip			Never logged in
carol	pts/1	192.168.1.75	Sat Sep 14 13:43:06 +0200 2019
dave	pts/3	192.168.1.75	Mon Sep 2 14:22:08 +0200 2019

NOTE

Il existe également des outils graphiques pour lire les fichiers de journalisation, par exemple : `gnome-logs` et `KSystemLog`.

Comment les messages deviennent des journaux

Voici comment un message est écrit dans un fichier de journalisation :

1. Les applications, les services et le noyau écrivent leurs messages dans des fichiers spéciaux (sockets et mémoires tampons) comme `/dev/log` ou `/dev/kmsg`.
2. `rsyslogd` récupère les informations depuis les sockets ou les mémoires tampons.
3. En fonction des règles trouvées dans `/etc/rsyslog.conf` et/ou les fichiers dans `/etc/rsyslog.d/`, `rsyslogd` déplace les informations vers les fichiers de journalisation correspondants (que l'on trouve généralement dans `/var/log`).

NOTE

Un socket est un fichier spécial qui permet de transférer des informations entre différents processus. Pour afficher la liste de tous les sockets de votre système, vous pouvez utiliser la commande `systemctl list-sockets --all`.

Ressources, priorités et actions

Le fichier de configuration de `rsyslog` est `/etc/rsyslog.conf` (dans certaines distributions, vous trouverez également des fichiers de configuration dans `/etc/rsyslog.d/`). En général, il est organisé en trois sections : `MODULES`, `GLOBAL DIRECTIVES` et `RULES`. Jetons-y un œil et examinons le fichier `rsyslog.conf` de notre hôte Debian GNU/Linux 10 (buster) - vous pouvez utiliser `sudo less /etc/rsyslog.conf` pour le faire.

`MODULES` inclut le support des modules pour la journalisation, le traitement des messages et la

réception UDP/TCP des journaux :

```
#####
#### MODULES ####
#####

module(load="imuxsock") # provides support for local system logging
module(load="imklog")   # provides kernel logging support
#module(load="immark")  # provides --MARK-- message capability

# provides UDP syslog reception
#module(load="imudp")
#input(type="imudp" port="514")

# provides TCP syslog reception
#module(load="imtcp")
#input(type="imtcp" port="514")
```

GLOBAL DIRECTIVES nous permet de configurer un certain nombre de choses comme les permissions des fichiers et des répertoires de journalisation :

```
#####
#### GLOBAL DIRECTIVES ####
#####

#
# Use traditional timestamp format.
# To enable high precision timestamps, comment out the following line.
#
$ActionFileDefaultTemplate RSYSLOG_TraditionalFileFormat

#
# Set the default permissions for all log files.
#
$FileOwner root
$FileGroup adm
$FileCreateMode 0640
$DirCreateMode 0755
$Umask 0022

#
# Where to place spool and state files
#
```

```
$WorkDirectory /var/spool/rsyslog

#
# Include all config files in /etc/rsyslog.d/
#
$IncludeConfig /etc/rsyslog.d/*.conf
```

RULES est l'endroit où les *ressources*, les *priorités* et les *actions* entrent en jeu. Les paramètres de cette section indiquent au démon de journalisation de filtrer les messages en fonction de certaines règles et de les enregistrer ou de les envoyer là où c'est nécessaire. Pour comprendre ces règles, nous devons d'abord expliquer les concepts de ressources et de priorités propres à `rsyslog`. Chaque message de journalisation est doté d'un numéro de *ressource* et d'un mot-clé associés au sous-système interne de Linux qui génère le message :

Numéro	Mot clé	Description
0	kern	Messages du noyau Linux
1	user	Messages au niveau utilisateur
2	mail	Système de messagerie
3	daemon	Démons du système
4	auth, authpriv	Messages relatifs à la sécurité et à l'autorisation
5	syslog	Messages syslogd
6	lpr	Sous-système d'impression
7	news	Sous-système de news en réseau
8	uucp	Sous-système UUCP (<i>Unix-to-Unix Copy Protocol</i>)
9	cron	Démon de l'horloge
10	auth, authpriv	Messages relatifs à la sécurité et à l'autorisation
11	ftp	Démon FTP (<i>File Transfer Protocol</i>)
12	ntp	Démon NTP (<i>Network Time Protocol</i>)
13	security	Audit du journal

Numéro	Mot clé	Description
14	console	Alerte du journal
15	cron	Démon de l'horloge
16 - 23	local0 à local7	Utilisation locale 0 - 7

Par ailleurs, chaque message est associé à un niveau de *priorité* :

Numéro	Sévérité	Mot clé	Description
0	Urgence	emerg, panic	Système inutilisable
1	Alerte	alert	Action immédiate requise
2	Critique	crit	Conditions critiques
3	Erreur	err, error	Conditions d'erreur
4	Avertissement	warn, warning	Conditions d'avertissement
5	Avis	notice	État normal mais significatif
6	Information	info	Messages à titre d'information
7	Débogage	debug	Messages de débogage

Voici un extrait de `rsyslog.conf` de notre système Debian GNU/Linux 10 (buster) avec quelques exemples de règles :

```
#####
#### RULES ####
#####

# First some standard log files.  Log by facility.
#
auth,authpriv.*          /var/log/auth.log
*.*;auth,authpriv.none   -/var/log/syslog
#cron.*                  /var/log/cron.log
daemon.*                 -/var/log/daemon.log
kern.*                   -/var/log/kern.log
lpr.*                    -/var/log/lpr.log
mail.*                   -/var/log/mail.log
```

```

user.*                                -/var/log/user.log

#
# Logging for the mail system. Split it up so that
# it is easy to write scripts to parse these files.
#
mail.info                            -/var/log/mail.info
mail.warn                            -/var/log/mail.warn
mail.err                             /var/log/mail.err

#
# Some "catch-all" log files.
#
*.=debug;\
    auth,authpriv.none;\
    news.none;mail.none             -/var/log/debug
*.=info;*.=notice;*.=warn;\
    auth,authpriv.none;\
    cron,daemon.none;\
    mail,news.none                  -/var/log/messages

```

Le format des règles est défini comme ceci : `<ressource>.<priorité> <action>`

Le sélecteur `<ressource>.<priorité>` filtre les messages à faire correspondre. Les niveaux de priorité sont hiérarchiquement inclusifs, ce qui signifie que `rsyslog` va faire correspondre les messages du niveau de priorité spécifié et plus. L'élément `<action>` indique l'action à entreprendre (où envoyer le message). Voici quelques exemples pour plus de clarté :

```
auth,authpriv.*                      /var/log/auth.log
```

Peu importe leur priorité (*), tous les messages provenant des ressources `auth` ou `authpriv` seront envoyés vers `/var/log/auth.log`.

```
*.*;auth,authpriv.none              -/var/log/syslog
```

Tous les messages — indépendamment de leur priorité (*) — de toutes les ressources (*) — sans tenir compte de ceux de `auth` ou `authpriv` (d'où le suffixe `.none`) — seront envoyés vers `/var/log/syslog` (le signe moins (-) avant le chemin évite les écritures excessives sur le disque). Notez le point-virgule (;) pour séparer le sélecteur et la virgule (,) pour concaténer deux ressources dans la même règle (`auth,authpriv`).

```
mail.err                                /var/log/mail.err
```

Les messages de la ressource `mail` avec un niveau de priorité `error` ou plus (`critical`, `alert` ou `emergency`) seront envoyés vers `/var/log/mail.err`.

```
*.=debug;\
    auth,authpriv.none;\
    news.none;mail.none    -/var/log/debug
```

Les messages en provenance de toutes les ressources avec la priorité `debug` et aucune autre (=) seront envoyés vers `/var/log/debug` — à l'exception de tous les messages en provenance des ressources `auth`, `authpriv`, `news` et `mail` (notez la syntaxe : `;` \).

Entrées manuelles dans les journaux du système : `logger`

La commande `logger` est pratique pour les scripts shell ou pour les tests. `logger` enverra tout message reçu vers `/var/log/syslog` (ou vers `/var/log/messages` si l'enregistrement se fait sur un serveur central distant, comme nous allons le voir plus loin dans cette leçon) :

```
carol@debian:~$ logger this comment goes into "/var/log/syslog"
```

Pour afficher la dernière ligne dans `/var/log/syslog`, utilisez la commande `tail` avec l'option `-1` :

```
root@debian:~# tail -1 /var/log/syslog
Sep 17 17:55:33 debian carol: this comment goes into /var/log/syslog
```

`rsyslog` comme serveur de journalisation centralisé

Pour expliquer ce sujet, nous allons ajouter un nouvel hôte à notre configuration. La voici :

Rôle	Nom d'hôte	OS	Adresse IP
Serveur de journalisation centralisé	suse-server	openSUSE Leap 15.1	192.168.1.6
Client	debian	Debian GNU/Linux 10 (buster)	192.168.1.4

Commençons par configurer le serveur. Tout d'abord, nous allons nous assurer que `rsyslog` fonctionne :

```
root@suse-server:~# systemctl status rsyslog
rsyslog.service - System Logging Service
   Loaded: loaded (/usr/lib/systemd/system/rsyslog.service; enabled; vendor preset: enabled)
   Active: active (running) since Thu 2019-09-17 18:45:58 CEST; 7min ago
     Docs: man:rsyslogd(8)
           http://www.rsyslog.com/doc/
  Main PID: 832 (rsyslogd)
    Tasks: 5 (limit: 4915)
   CGroup: /system.slice/rsyslog.service
           └─832 /usr/sbin/rsyslogd -n -iNONE
```

openSUSE comprend un fichier de configuration dédié à la journalisation à distance : `/etc/rsyslog.d/remote.conf`. Nous allons activer la réception des messages depuis les clients (hôtes distants) via TCP. Pour ce faire, nous devons décommenter les lignes qui chargent le module et qui démarrent le serveur TCP sur le port 514 :

```
# ##### Receiving Messages from Remote Hosts #####
# TCP Syslog Server:
# provides TCP syslog reception and GSS-API (if compiled to support it)
$ModLoad imtcp.so # load module
##$UDPServerAddress 10.10.0.1 # force to listen on this IP only
$InputTCPServerRun 514 # Starts a TCP server on selected port

# UDP Syslog Server:
#$ModLoad imudp.so # provides UDP syslog reception
##$UDPServerAddress 10.10.0.1 # force to listen on this IP only
#$UDPServerRun 514 # start a UDP syslog server at standard port 514
```

Une fois que c'est fait, nous devons redémarrer le service `rsyslog` et vérifier que le serveur écoute sur le port 514 :

```
root@suse-server:~# systemctl restart rsyslog
root@suse-server:~# netstat -nltp | grep 514
[sudo] password for root:
tcp        0      0 0.0.0.0:514          0.0.0.0:*          LISTEN
2263/rsyslogd
tcp6       0      0 :::514              :::*                LISTEN
2263/rsyslogd
```

Ensuite, nous devons ouvrir les ports dans le pare-feu et recharger la configuration :

```
root@suse-server:~# firewall-cmd --permanent --add-port 514/tcp
success
root@suse-server:~# firewall-cmd --reload
success
```

NOTE

Avec l'avènement d'openSUSE Leap 15.0, `firewalld` a remplacé complètement le classique `SuSEFirewall2`.

Modèles et conditions de filtrage

Par défaut, les journaux du client sont envoyés vers le fichier `/var/log/messages` du serveur — avec ceux du serveur lui-même. Or, nous allons créer un *modèle* et une *condition de filtrage* pour que les journaux de nos clients soient stockés dans des répertoires séparés qui leur sont propres. Pour ce faire, nous allons ajouter ce qui suit à `/etc/rsyslog.conf` (ou `/etc/rsyslog.d/remote.conf`) :

```
$template RemoteLogs, "/var/log/remotehosts/%HOSTNAME%/%$NOW%.%syslogseverity-text%.log"
if $FROMHOST-IP=='192.168.1.4' then ?RemoteLogs
& stop
```

Modèle

Le modèle correspond à la première ligne et vous permet de spécifier un format pour les noms de journaux en utilisant la génération dynamique de noms de fichiers. Un modèle est constitué de :

- Directive du modèle (`$template`)
- Nom du modèle (`RemoteLogs`)
- Texte du modèle (`"/var/log/remotehosts/%HOSTNAME%/%$NOW%.%syslogseverity-text%.log"`)
- Options (optionnelles)

Notre modèle s'appelle `RemoteLogs` et son texte consiste en un chemin dans `/var/log`. Tous les journaux de notre hôte distant iront dans le répertoire `remotehosts`, où un sous-répertoire sera créé en fonction du nom d'hôte de la machine (`%HOSTNAME%`). Chaque nom de fichier dans ce répertoire sera composé de la date (`%$NOW%`), de la sévérité (ou priorité) du message au format texte (`%syslogseverity-text%`) et du suffixe `.log`. Les mots entre les signes de pourcentage sont des *propriétés* et vous permettent d'accéder au contenu du message (date, priorité, etc.). Un

message `syslog` a un certain nombre de propriétés bien définies qui peuvent être utilisées dans les modèles. Ces propriétés sont accessibles — et peuvent être modifiées — par ce que l'on appelle le *remplaçant de propriétés* qui consiste à les écrire entre des signes de pourcentages.

Condition de filtrage

Les deux lignes restantes correspondent à la condition de filtrage et à l'action associée :

- Filtre basé sur une expression (`if $FROMHOST-IP== '192.168.1.4'`)
- Action (`then ?RemoteLogs, & stop`)

La première ligne vérifie l'adresse IP de l'hôte distant qui envoie le journal et — si elle est égale à celle de notre client Debian — elle applique le modèle `RemoteLogs`. La dernière ligne (`& stop`) assure que les messages ne sont pas envoyés simultanément à `/var/log/messages` (mais seulement vers les fichiers du répertoire `/var/log/remotehosts`).

NOTE

Pour en savoir plus sur les modèles, les propriétés et les règles, vous pouvez consulter la page de manuel de `rsyslog.conf`.

Avec la configuration mise à jour, nous allons redémarrer `rsyslog` et vérifier qu'il n'y a pas encore de répertoire `remotehosts` dans `/var/log` :

```
root@suse-server:~# systemctl restart rsyslog
root@suse-server:~# ls /var/log/
acpid          chrony         localmessages  pbl.log        Xorg.0.log
alternatives.log cups           mail           pk_backend_zypp Xorg.0.log.old
apparmor       firebird       mail.err       samba          YaST2
audit          firewall       mail.info      snapper.log    zypp
boot.log        firewalld      mail.warn      tallylog       zypper.log
boot.msg        krb5           messages       tuned
boot.omsg       lastlog        mysql          warn
btmpt           lightdm        NetworkManager wtmp
```

Le serveur est maintenant configuré. À présent, nous allons configurer le client.

Là aussi, nous devons nous assurer que `rsyslog` est installé et fonctionne :

```
root@debian:~# sudo systemctl status rsyslog
rsyslog.service - System Logging Service
Loaded: loaded (/lib/systemd/system/rsyslog.service; enabled; vendor preset:
Active: active (running) since Thu 2019-09-17 18:47:54 CEST; 7min ago
Docs: man:rsyslogd(8)
```



```

http://www.rsyslog.com/doc/
Main PID: 351 (rsyslogd)
  Tasks: 4 (limit: 4915)
  CGroup: /system.slice/rsyslog.service
          └─351 /usr/sbin/rsyslogd -n

```

Dans notre environnement de test, nous avons intégré la résolution de noms sur le client en ajoutant la ligne `192.168.1.6 suse-server` à `/etc/hosts`. Ainsi, nous pouvons nous référer au serveur soit par son nom (`suse-server`) soit par son adresse IP (`192.168.1.6`).

Notre client Debian n'a pas de fichier `remote.conf` dans `/etc/rsyslog.d/`, nous appliquerons donc nos configurations dans `/etc/rsyslog.conf`. Nous écrirons la ligne suivante à la fin du fichier :

```
*.* @@suse-server:514
```

Enfin, nous redémarrons `rsyslog`.

```
root@debian:~# systemctl restart rsyslog
```

Maintenant, revenons à notre machine `suse-server` et vérifions l'existence de `remotehosts` dans `/var/log` :

```

root@suse-server:~# ls /var/log/remotehosts/debian/
2019-09-17.info.log  2019-09-17.notice.log

```

Nous avons déjà deux logs dans `/var/log/remotehosts` comme décrit dans notre modèle. Pour terminer cette partie, nous lançons `tail -f 2019-09-17.notice.log` sur `suse-server` pendant que nous envoyons *manuellement* un journal depuis notre client Debian pour confirmer que les messages sont ajoutés au fichier de journalisation comme prévu (l'option `-t` fournit une balise pour notre message) :

```

root@suse-server:~# tail -f /var/log/remotehosts/debian/2019-09-17.notice.log
2019-09-17T20:57:42+02:00 debian dbus[323]: [system] Successfully activated service
'org.freedesktop.nm_dispatcher'
2019-09-17T21:01:41+02:00 debian anacron[1766]: Anacron 2.3 started on 2019-09-17
2019-09-17T21:01:41+02:00 debian anacron[1766]: Normal exit (0 jobs run)

```

```
carol@debian:~$ logger -t DEBIAN-CLIENT Hi from 192.168.1.4
```

```
root@suse-server:~# tail -f /var/log/remotehosts/debian/2019-09-17.notice.log
2019-09-17T20:57:42+02:00 debian dbus[323]: [system] Successfully activated service
'org.freedesktop.nm_dispatcher'
2019-09-17T21:01:41+02:00 debian anacron[1766]: Anacron 2.3 started on 2019-09-17
2019-09-17T21:01:41+02:00 debian anacron[1766]: Normal exit (0 jobs run)
2019-09-17T21:04:21+02:00 debian DEBIAN-CLIENT: Hi from 192.168.1.4
```

Mécanisme de rotation des journaux

Les fichiers de journalisation font l'objet d'une rotation régulière, pour deux raisons principalement :

- Empêcher les anciens fichiers de journalisation d'utiliser plus d'espace disque que nécessaire.
- Les journaux doivent garder une taille raisonnable pour faciliter leur consultation.

L'outil en charge de la rotation (ou du cyclage) des journaux est `logrotate` et son rôle comprend des actions comme le renommage des fichiers de journalisation, leur archivage et/ou leur compression, leur envoi par email à l'administrateur du système et finalement leur suppression au fur et à mesure qu'ils vieillissent. Il existe plusieurs conventions pour nommer ces fichiers de journalisation pivotés (en ajoutant un suffixe avec la date au nom du fichier, par exemple) ; cependant, l'ajout d'un suffixe avec un nombre entier est la pratique courante :

```
root@debian:~# ls /var/log/messages*
/var/log/messages /var/log/messages.1 /var/log/messages.2.gz /var/log/messages.3.gz
/var/log/messages.4.gz
```

Voyons maintenant ce qui va se passer lors de la prochaine rotation des journaux :

1. `messages.4.gz` sera supprimé et perdu.
2. Le contenu de `messages.3.gz` sera déplacé vers `messages.4.gz`.
3. Le contenu de `messages.2.gz` sera déplacé vers `messages.3.gz`.
4. Le contenu de `messages.1` sera déplacé vers `messages.2.gz`.
5. Le contenu de `messages` sera déplacé vers `messages.1` et `messages` sera vide et prêt à enregistrer de nouvelles entrées du journal.

Notez comment — selon les directives `logrotate` que vous verrez bientôt — les trois fichiers logs

les plus anciens sont compressés, alors que les deux plus récents ne le sont pas. Par ailleurs, nous conservons les logs des 4 à 5 dernières semaines. Pour lire les messages datant de la semaine dernière, nous consulterons `messages.1` (et ainsi de suite).

`logrotate` est exécuté quotidiennement comme une tâche automatique ou un *cronjob* à travers le script `/etc/cron.daily/logrotate` et lit le fichier de configuration `/etc/logrotate.conf`. Ce fichier inclut quelques options globales et il est bien commenté, chaque option étant introduite par une brève explication de son utilité :

```
carol@debian:~$ sudo less /etc/logrotate.conf
# see "man logrotate" for details
# rotate log files weekly
weekly

# keep 4 weeks worth of backlogs
rotate 4

# create new (empty) log files after rotating old ones
create

# uncomment this if you want your log files compressed
#compress

# packages drop log rotation information into this directory
include /etc/logrotate.d

(...)
```

Comme vous pouvez le voir, les fichiers de configuration dans `/etc/logrotate.d` pour des paquets spécifiques sont également inclus. Ces fichiers contiennent—pour la plupart—des définitions locales et spécifient les fichiers de journalisation à faire tourner (rappelez-vous que les définitions locales sont prioritaires par rapport aux définitions globales et que les définitions ultérieures ont la priorité sur les précédentes). Ce qui suit est un extrait d’une définition dans `/etc/logrotate.d/rsyslog`:

```
/var/log/messages
{
    rotate 4
    weekly
    missingok
    notifempty
    compress
```

```
    delaycompress
    sharedscripts
    postrotate
        invoke-rs.d rsyslog rotate > /dev/null
    endscript
}
```

Comme vous pouvez le voir, chaque directive est séparée de sa valeur par un espace et/ou un signe égal optionnel (=). Les lignes entre `postrotate` et `endscript` doivent cependant apparaître sur des lignes séparées. Voici une explication :

rotate 4

Conserver les journaux des 4 dernières semaines.

weekly

Effectuer une rotation hebdomadaire des fichiers de journalisation.

missingok

Ne pas afficher de message d'erreur si le fichier de journalisation est manquant et passer au suivant.

notifempty

Ne pas faire tourner le fichier de journalisation s'il est vide.

compress

Compresser les fichiers de journalisation avec `gzip` (par défaut).

delaycompress

Reporte la compression du fichier de journalisation précédent au prochain cycle de rotation (efficace uniquement en combinaison avec `compress`). C'est utile lorsqu'on ne peut pas demander à un programme de fermer son fichier de journalisation et qu'il peut donc continuer à écrire dans le fichier de journalisation précédent pendant un certain temps.

sharedscripts

Lié aux scripts `prerotate` et `postrotate`. Pour éviter qu'un script ne soit exécuté plusieurs fois, lancez les scripts une seule fois, quel que soit le nombre de fichiers de journalisation qui correspondent à un motif donné (par exemple `/var/log/mail/*`). De plus, si les scripts se terminent avec des erreurs, les actions restantes ne seront pas exécutées pour tous les fichiers de journalisation.

postrotate

Indique le début d'un script *postrotate*.

invoke-rc.d rsyslog rotate > /dev/null

Utiliser `/bin/sh` pour lancer `invoke-rc.d rsyslog rotate > /dev/null` après la rotation des fichiers de journalisation.

endscript

Indique la fin du script *postrotate*.

NOTE

Pour la liste complète des directives et des explications, consultez la page de manuel de `logrotate.conf`.

La mémoire tampon circulaire du noyau

Puisque le noyau génère plusieurs messages avant que `rsyslogd` ne devienne disponible au démarrage, un mécanisme pour enregistrer ces messages devient nécessaire. C'est là que la mémoire tampon circulaire du noyau (*kernel ring buffer*) entre en jeu. C'est une structure de données de taille fixe et — par conséquent — au fur et à mesure qu'elle se remplit de nouveaux messages, les plus anciens disparaissent.

La commande `dmesg` affiche la mémoire tampon circulaire du noyau. En raison de la taille du tampon, cette commande est normalement utilisée en combinaison avec l'outil de filtrage de texte `grep`. Par exemple, pour rechercher les messages relatifs aux périphériques USB :

```
root@debian:~# dmesg | grep "usb"
[ 1.241182] usbcore: registered new interface driver usbfs
[ 1.241188] usbcore: registered new interface driver hub
[ 1.250968] usbcore: registered new device driver usb
[ 1.339754] usb usb1: New USB device found, idVendor=1d6b, idProduct=0001, bcdDevice=4.19
[ 1.339756] usb usb1: New USB device strings: Mfr=3, Product=2, SerialNumber=1
(...)
```

Exercices guidés

1. Quels outils/commandes utiliseriez-vous dans ces cas de figure :

Objectif et fichier de journalisation	Outil
Lire <code>/var/log/syslog.7.gz</code>	
Lire <code>/var/log/syslog</code>	
Chercher le mot <code>renewal</code> dans <code>/var/log/syslog</code>	
Lire <code>/var/log/faillog</code>	
Lire <code>/var/log/syslog</code> dynamiquement	

2. Réorganiser les entrées de journal suivantes de manière à ce qu'elles représentent un message de journal valide avec la structure appropriée :

- `debian-server`
- `sshd`
- `[515]:`
- `Sep 13 21:47:56`
- `Server listening on 0.0.0.0 port 22`

La séquence correcte est :

3. Quelles règles ajouteriez-vous à `/etc/rsyslog.conf` afin d'accomplir chacune des actions ci-dessous :

- Envoyer tous les messages en provenance de la ressource `mail` avec un niveau de priorité/sévérité `crit` (et plus) vers `/var/log/mail.crit` :

- Envoyer tous les messages en provenance de la ressource `mail` avec les niveaux de priorité `alert` et `emergency` vers `/var/log/mail.urgent` :

- À l'exception de ceux en provenance des ressources `cron` et `ntp`, envoyer tous les messages—indépendamment de leur ressource et de leur niveau de priorité—vers `/var/log/allmessages` :

- Une fois que tous les paramètres requis sont correctement configurés, envoyer tous les messages de la ressource mail vers un hôte distant dont l'adresse IP est 192.168.1.88 en utilisant TCP et en spécifiant le port par défaut :

- Indépendamment de leur ressource, envoyer tous les messages avec un niveau de priorité warning (*uniquement avec la priorité warning*) vers `/var/log/warnings` de manière à éviter des écritures excessives sur le disque :

4. Examinez cette stance de `/etc/logrotate.d/samba` et expliquez les différentes options :

```
carol@debian:~$ sudo head -n 11 /etc/logrotate.d/samba
/var/log/samba/log.smbd {
    weekly
    missingok
    rotate 7
    postrotate
        [ ! -f /var/run/samba/smbd.pid ] || /etc/init.d/smbd reload > /dev/null
    endscript
    compress
    delaycompress
    notifempty
}
```

Option	Signification
weekly	
missingok	
rotate 7	
postrotate	
endscript	
compress	
delaycompress	
notifyempty	

Exercices d'approfondissement

1. Dans la section “Modèles et conditions de filtrage” nous avons utilisé un *filtre basé sur une expression* comme condition de filtrage. Les filtres basés sur les propriétés sont un autre type de filtre propre à `rsyslogd`. Traduisez notre *filtre basé sur une expression* en un *filtre basé sur une propriété* :

Filtre basé sur une expression	Filtre basé sur une propriété
<pre>if \$FROMHOST-IP== '192.168.1.4' then ?RemoteLogs</pre>	

2. `omusrmsg` est un module intégré à `rsyslog` qui facilite les notifications aux utilisateurs (il envoie des messages de journalisation dans le terminal des utilisateurs). Écrivez une règle pour envoyer tous les messages d'urgence de toutes les ressources à la fois vers `root` et vers l'utilisateur normal `carol`.

Résumé

Voici ce que nous avons appris dans cette leçon :

- La journalisation est essentielle pour l'administration système.
- `rsyslogd` est l'outil chargé de garder les journaux propres et bien organisés.
- Certains services gèrent leurs propres journaux.
- En gros, les journaux peuvent être classés en journaux système et en journaux de services ou de programmes.
- Il existe un certain nombre d'outils pratiques pour la lecture des journaux : `less`, `more`, `zless`, `zmore`, `grep`, `head` et `tail`.
- La plupart des journaux sont des fichiers au format texte simple, mais il existe un petit nombre de journaux au format binaire.
- Pour ce qui est des journaux, `rsyslogd` reçoit les informations pertinentes via des fichiers spéciaux (sockets et mémoires tampon) avant de les traiter.
- Pour catégoriser les journaux, `rsyslogd` utilise les règles dans `/etc/rsyslog.conf` ou `/etc/rsyslog.d/*`.
- Tout utilisateur peut écrire manuellement ses propres messages dans les journaux du système à l'aide de l'outil `logger`.
- `rsyslog` vous permet de conserver tous les journaux à travers des réseaux IP dans un serveur de journaux centralisé.
- Les Modèles sont très utiles pour formater les noms des fichiers de journalisation de manière dynamique.
- La rotation des journaux se fait pour deux raisons : empêcher les anciens journaux d'utiliser un espace disque excessif et rendre la consultation des journaux plus facile à gérer.

Réponses aux exercices guidés

1. Quels outils/commandes utiliseriez-vous dans ces cas de figure :

Objectif et fichier de journalisation	Outil
Lire <code>/var/log/syslog.7.gz</code>	<code>zmore</code> ou <code>zless</code>
Lire <code>/var/log/syslog</code>	<code>more</code> ou <code>less</code>
Chercher le mot <code>renewal</code> dans <code>/var/log/syslog</code>	<code>grep</code>
Lire <code>/var/log/faillog</code>	<code>faillog -a</code>
Lire <code>/var/log/syslog</code> dynamiquement	<code>tail -f</code>

2. Réorganiser les entrées de journal suivantes de manière à ce qu'elles représentent un message de journal valide avec la structure appropriée :

- `debian-server`
- `sshd`
- `[515]:`
- `Sep 13 21:47:56`
- `Server listening on 0.0.0.0 port 22`

La séquence correcte est :

```
Sep 13 21:47:56 debian-server sshd[515]: Server listening on 0.0.0.0 port 22
```

3. Quelles règles ajouteriez-vous à `/etc/rsyslog.conf` afin d'accomplir chacune des actions ci-dessous :

- Envoyer tous les messages en provenance de la ressource `mail` avec un niveau de priorité/sévérité `crit` (et plus) vers `/var/log/mail.crit` :

```
mail.crit                /var/log/mail.crit
```

- Envoyer tous les messages en provenance de la ressource `mail` avec les niveaux de priorité `alert` et `emergency` vers `/var/log/mail.urgent` :

```
mail.alert                                /var/log/mail.urgent
```

- À l'exception de ceux en provenance des ressources `cron` et `ntp`, envoyer tous les messages—indépendamment de leur ressource et de leur niveau de priorité—vers `/var/log/allmessages` :

```
*.*;cron.none;ntp.none                  /var/log/allmessages
```

- Une fois que tous les paramètres requis sont correctement configurés, envoyer tous les messages de la ressource `mail` vers un hôte distant dont l'adresse IP est `192.168.1.88` en utilisant TCP et en spécifiant le port par défaut :

```
mail.* @@192.168.1.88:514
```

- Indépendamment de leur ressource, envoyer tous les messages avec un niveau de priorité `warning` (*uniquement avec la priorité `warning`*) vers `/var/log/warnings` de manière à éviter des écritures excessives sur le disque :

```
*.=warning                               -/var/log/warnings
```

4. Examinez cette stance de `/etc/logrotate.d/samba` et expliquez les différentes options :

```
carol@debian:~$ sudo head -n 11 /etc/logrotate.d/samba
/var/log/samba/log.smbd {
    weekly
    missingok
    rotate 7
    postrotate
        [ ! -f /var/run/samba/smbd.pid ] || /etc/init.d/smbd reload > /dev/null
    endscrip
    compress
    delaycompress
    notifempty
}
```

Option	Signification
<code>weekly</code>	Effectuer une rotation hebdomadaire des fichiers de journalisation.
<code>missingok</code>	Ne pas afficher de message d'erreur si le fichier de journalisation est manquant et passer au suivant.
<code>rotate 7</code>	Conserver les journaux des 7 dernières semaines.
<code>postrotate</code>	Exécuter le script sur la ligne suivante après la rotation des journaux.
<code>endscript</code>	Indique la fin du script <i>postrotate</i> .
<code>compress</code>	Compresser les fichiers de journalisation avec <code>gzip</code> .
<code>delaycompress</code>	De concert avec <code>compress</code> , reporter la compression au prochain cycle de rotation des journaux.
<code>notifyempty</code>	Ne pas faire tourner le fichier de journalisation s'il est vide.

Réponses aux exercices d'approfondissement

1. Dans la section “Modèles et conditions de filtrage” nous avons utilisé un *filtre basé sur une expression* comme condition de filtrage. Les filtres basés sur les propriétés sont un autre type de filtre propre à `rsyslogd`. Traduisez notre *filtre basé sur une expression* en un *filtre basé sur une propriété* :

Filtre basé sur une expression	Filtre basé sur une propriété
if \$FROMHOST-IP=='192.168.1.4' then ?RemoteLogs	:fromhost-ip, isequal, "192.168.1.4" ?RemoteLogs

2. `omusrmsg` est un module intégré à `rsyslog` qui facilite les notifications aux utilisateurs (il envoie des messages de journalisation dans le terminal des utilisateurs). Écrivez une règle pour envoyer tous les messages d'urgence de toutes les ressources à la fois vers `root` et vers l'utilisateur normal `carol`.

```
*.emerg                                :omusrmsg:root,carol
```



108.2 Leçon 2

Certification :	LPIC-1
Version :	5.0
Thème :	108 Services Systèmes Essentiels
Objectif :	108.2 Journalisation du système
Leçon :	2 sur 2

Introduction

Avec l'adoption massive de `systemd` par toutes les principales distributions, le démon de journalisation (`systemd-journald`) est devenu le service de journalisation par défaut. Dans cette leçon, nous allons discuter son fonctionnement et son utilisation pour faire un certain nombre de choses : le consulter, filtrer les informations selon différents critères, configurer le stockage et la taille, effacer les données périmées, récupérer les données depuis un système de secours ou une copie du système de fichiers et — enfin et surtout — comprendre son interaction avec `rsyslogd`.

Bases de `systemd`

Présenté initialement sous Fedora, `systemd` a progressivement remplacé SysV Init en tant que gestionnaire de système et de services dans la plupart des grandes distributions Linux. Voici quelques-uns de ses principaux atouts :

- Facilité de configuration : fichiers d'unité par opposition aux scripts SysV Init.
- Gestion polyvalente : en dehors des démons et des processus, il gère également les périphériques, les sockets et les points de montage.

- Rétrocompatibilité avec SysV Init et Upstart.
- Chargement parallèle à l'amorçage : les services sont chargés en parallèle, alors que Sysv Init les charge l'un après l'autre.
- Il dispose d'un service de journalisation nommé *journal* avec les avantages suivants :
 - Tous les journaux sont centralisés en un seul endroit.
 - Il n'a pas besoin de rotation des journaux.
 - Les journaux peuvent être désactivés, chargés dans la RAM ou rendus persistants.

Unités et cibles

`systemd` fonctionne sur des *unités* (*units*). Une unité est une ressource que `systemd` peut gérer (par exemple le réseau, le Bluetooth, etc.). Les unités — à leur tour — sont gouvernées par des *fichiers d'unité*. Ce sont des fichiers texte rangés dans `/lib/systemd/system` et qui incluent les paramètres de configuration — sous forme de *sections* et de *directives* — pour une ressource particulière à gérer. Il existe un certain nombre de types d'unités : `service`, `mount`, `automount`, `swap`, `timer`, `device`, `socket`, `path`, `timer`, `snapshot`, `slice`, `scope` et `target`. Ainsi, chaque nom de fichier d'unité suit le schéma `<ressource>.<unité>` (par exemple `reboot.service`).

Une *cible* (*target*) est un type d'unité spécial qui ressemble aux niveaux d'exécution (*runlevels*) classiques de SysV Init. C'est parce qu'une *unité cible* (*target unit*) rassemble différentes ressources pour représenter un état particulier du système (par exemple, `graphical.target` est similaire à `runlevel 5`, etc). Pour vérifier la cible en cours dans votre système, utilisez la commande `systemctl get-default` :

```
carol@debian:~$ systemctl get-default
graphical.target
```

D'autre part, les cibles et les niveaux d'exécution diffèrent en ce que les premières sont mutuellement inclusives, alors que les seconds ne le sont pas. Ainsi, une cible peut lancer d'autres cibles, ce qui n'est pas possible avec les niveaux d'exécution.

NOTE

L'explication du fonctionnement des unités `systemd` dépasse le cadre de cette leçon.

Le journal du système : `systemd-journald`

`systemd-journald` est le service système qui se charge de recevoir les informations de journalisation depuis plusieurs sources : les messages du noyau, les journaux simples et structurés

du système, la sortie standard et l'erreur standard des services ainsi que les enregistrements d'audit du sous-système d'audit du noyau (pour plus de détails, voir la page de manuel de `systemd-journald`). Sa mission consiste à créer et à maintenir un journal structuré et indexé.

Son fichier de configuration est `/etc/systemd/journald.conf` et—comme pour tout autre service—vous pouvez utiliser la commande `systemctl` pour le démarrer, le redémarrer, l'arrêter ou—simplement—vérifier son état :

```
root@debian:~# systemctl status systemd-journald
systemd-journald.service - Journal Service
   Loaded: loaded (/lib/systemd/system/systemd-journald.service; static; vendor preset:
   enabled)
   Active: active (running) since Sat 2019-10-12 13:43:06 CEST; 5min ago
     Docs: man:systemd-journald.service(8)
           man:journald.conf(5)
  Main PID: 178 (systemd-journal)
    Status: "Processing requests..."
     Tasks: 1 (limit: 4915)
    CGroup: /system.slice/systemd-journald.service
            └─178 /lib/systemd/systemd-journald
 (...)

```

Les fichiers de configuration du type `journal.conf.d/*.conf`—qui peuvent inclure des configurations spécifiques à un paquet—sont également possibles (consultez la page de manuel de `journald.conf` pour en savoir plus).

Si le journal est activé, il peut être stocké de manière persistante sur le disque ou de manière volatile sur un système de fichiers en mémoire vive. Le journal n'est pas un fichier texte, c'est un fichier binaire. Ainsi, vous ne pourrez pas utiliser des outils textuels tels que `less` ou `more` pour lire son contenu ; la commande `journalctl` sera utilisée à la place.

Interroger le contenu du journal

`journalctl` est l'outil que vous utiliserez pour interroger le journal `systemd`. Vous devez être root ou utiliser `sudo` pour l'invoquer. S'il est invoqué sans options, il affichera l'intégralité du journal par ordre chronologique (les entrées les plus anciennes sont affichées en premier) :

```
root@debian:~# journalctl
-- Logs begin at Sat 2019-10-12 13:43:06 CEST, end at Sat 2019-10-12 14:19:46 CEST. --
Oct 12 13:43:06 debian kernel: Linux version 4.9.0-9-amd64 (debian-kernel@lists.debian.org)
 (...)

```



```
Oct 12 13:43:06 debian kernel: Command line: BOOT_IMAGE=/boot/vmlinuz-4.9.0-9-amd64
root=UUID=b6be6117-5226-4a8a-bade-2db35ccf4cf4 ro qu
(...)
```

Vous pouvez effectuer des requêtes plus spécifiques en utilisant un certain nombre d'options :

-r

Les messages du journal seront affichés dans l'ordre inverse :

```
root@debian:~# journalctl -r
-- Logs begin at Sat 2019-10-12 13:43:06 CEST, end at Sat 2019-10-12 14:30:30 CEST. --
Oct 12 14:30:30 debian sudo[1356]: pam_unix(sudo:session): session opened for user root
by carol(uid=0)
Oct 12 14:30:30 debian sudo[1356]:    carol : TTY=pts/0 ; PWD=/home/carol ; USER=root ;
COMMAND=/bin/journalctl -r
Oct 12 14:19:53 debian sudo[1348]: pam_unix(sudo:session): session closed for user root
(...)
```

-f

Cette option va afficher les messages les plus récents du journal et continuer à afficher les nouvelles entrées au fur et à mesure qu'elles s'y ajoutent — un peu comme `tail -f` :

```
root@debian:~# journalctl -f
-- Logs begin at Sat 2019-10-12 13:43:06 CEST. --
(...)
Oct 12 14:44:42 debian sudo[1356]: pam_unix(sudo:session): session closed for user root
Oct 12 14:44:44 debian sudo[1375]:    carol : TTY=pts/0 ; PWD=/home/carol ; USER=root ;
COMMAND=/bin/journalctl -f
Oct 12 14:44:44 debian sudo[1375]: pam_unix(sudo:session): session opened for user root
by carol(uid=0)

(...)
```

-e

Cette option va aller à la fin du journal, de sorte que les dernières entrées soient visibles dans le pager :

```
root@debian:~# journalctl -e
(...)
Oct 12 14:44:44 debian sudo[1375]:    carol : TTY=pts/0 ; PWD=/home/carol ; USER=root ;
```

```
COMMAND=/bin/journalctl -f
Oct 12 14:44:44 debian sudo[1375]: pam_unix(sudo:session): session opened for user root
by carol(uid=0)
Oct 12 14:45:57 debian sudo[1375]: pam_unix(sudo:session): session closed for user root
Oct 12 14:48:39 debian sudo[1378]:    carol : TTY=pts/0 ; PWD=/home/carol ; USER=root ;
COMMAND=/bin/journalctl -e
Oct 12 14:48:39 debian sudo[1378]: pam_unix(sudo:session): session opened for user root
by carol(uid=0)
```

-n <valeur>, --lines=<valeur>

Cette option va afficher les *valeur* dernières lignes (si <valeur> n'est pas spécifiée, elle vaut 10 par défaut) :

```
root@debian:~# journalctl -n 5
(...)
Oct 12 14:44:44 debian sudo[1375]:    carol : TTY=pts/0 ; PWD=/home/carol ; USER=root ;
COMMAND=/bin/journalctl -f
Oct 12 14:44:44 debian sudo[1375]: pam_unix(sudo:session): session opened for user root
by carol(uid=0)
Oct 12 14:45:57 debian sudo[1375]: pam_unix(sudo:session): session closed for user root
Oct 12 14:48:39 debian sudo[1378]:    carol : TTY=pts/0 ; PWD=/home/carol ; USER=root ;
COMMAND=/bin/journalctl -e
Oct 12 14:48:39 debian sudo[1378]: pam_unix(sudo:session): session opened for user root
by carol(uid=0)
```

-k, --dmesg

Équivaut à la commande `dmesg` :

```
root@debian:~# journalctl -k
-- Logs begin at Sat 2019-10-12 13:43:06 CEST, end at Sat 2019-10-12 14:53:20 CEST. --
Oct 12 13:43:06 debian kernel: Linux version 4.9.0-9-amd64 (debian-
kernel@lists.debian.org) (gcc version 6.3.0 20170516 (Debian 6.3.0-18
Oct 12 13:43:06 debian kernel: Command line: BOOT_IMAGE=/boot/vmlinuz-4.9.0-9-amd64
root=UUID=b6be6117-5226-4a8a-bade-2db35ccf4cf4 ro qu
Oct 12 13:43:06 debian kernel: x86/fpu: Supporting XSAVE feature 0x001: 'x87 floating
point registers'
Oct 12 13:43:06 debian kernel: x86/fpu: Supporting XSAVE feature 0x002: 'SSE registers'
(...)
```

Navigation et recherche dans le journal

Vous pouvez naviguer dans les résultats du journal avec :

- PageHaut, PageBas et les touches fléchées pour vous déplacer vers le haut, le bas, la gauche et la droite.
-  pour aller à la fin du résultat.
-  pour aller au début du résultat.

Vous pouvez rechercher des chaînes de caractères en aval et en amont de votre position actuelle :

- Recherche en aval : Appuyez sur  et entrez la chaîne à rechercher, puis appuyez sur Entrée.
- Recherche en amont : Appuyez sur  et entrez la chaîne à rechercher, puis appuyez sur Entrée.

Pour naviguer dans les résultats de la recherche, utilisez  pour passer à l'occurrence suivante d'une correspondance et  +  pour passer à l'occurrence précédente.

Filtrer les données du journal

Le journal vous permet de filtrer les données des fichiers de journalisation selon un certain nombre de critères :

Numéro d'amorçage

--list-boots

Cette option affiche tous les amorçages disponibles. Le résultat consiste en trois colonnes ; la première spécifie le numéro d'amorçage (0 se réfère à l'amorçage actuel, -1 est l'amorçage précédent, -2 celui qui précède l'amorçage précédent et ainsi de suite) ; la deuxième colonne est l'ID de l'amorçage ; la troisième affiche les horodatages :

```
root@debian:~# journalctl --list-boots
0 83df3e8653474ea5aed19b41cdb45b78 Sat 2019-10-12 18:55:41 CEST-Sat 2019-10-12
19:02:24 CEST
```

-b, --boot

Cette option affiche tous les messages de l'amorçage en cours. Pour voir les messages des amorçages précédents, il suffit d'ajouter un paramètre de décalage comme expliqué ci-dessus. Par exemple, pour afficher les messages de l'amorçage précédent, vous devrez taper `journalctl -b -1`. Ceci étant dit, rappelez-vous que pour récupérer les informations des journaux précédents, la persistance du journal doit être activée (vous apprendrez comment le faire dans la section suivante) :

```
root@debian:~# journalctl -b -1
Specifying boot ID has no effect, no persistent journal was found
```

Priorité

-p

Noter que vous pouvez également filtrer par niveau de sévérité/priorité avec l'option `-p` :

```
root@debian:~# journalctl -b -0 -p err
-- No entries --
```

Le journal nous informe que — jusqu'ici du moins — il n'y a pas eu de messages avec une priorité `error` (ou plus) depuis l'amorçage en cours. Remarque : `-b -0` peut être omis lorsqu'il s'agit de l'amorçage en cours.

NOTE

Reportez-vous à la leçon précédente pour une liste complète de tous les niveaux de sévérité (ou priorités) de `syslog`.

Intervalle de temps

Vous pouvez demander à `journalctl` de n'afficher que les messages enregistrés dans un intervalle de temps donné avec les options `--since` et `--until`. La spécification de la date doit suivre le format `YYYY-MM-DD HH:MM:SS`. Si le temps n'est pas spécifié, c'est minuit qui sera pris en compte. De même, si la date est omise, c'est la date du jour qui est prise en compte. Par exemple, pour voir les messages enregistrés entre 19h00 et 19h01, vous devez taper :

```
root@debian:~# journalctl --since "19:00:00" --until "19:01:00"
-- Logs begin at Sat 2019-10-12 18:55:41 CEST, end at Sat 2019-10-12 20:10:50 CEST. --
Oct 12 19:00:14 debian systemd[1]: Started Run anacron jobs.
Oct 12 19:00:14 debian anacron[1057]: Anacron 2.3 started on 2019-10-12
Oct 12 19:00:14 debian anacron[1057]: Normal exit (0 jobs run)
Oct 12 19:00:14 debian systemd[1]: anacron.timer: Adding 2min 47.988096s random time.
```

De même, vous pouvez utiliser une spécification de temps légèrement différente (un nombre entier avec l'unité du temps et le mot *ago*) : `"integer time-unit ago"`. Ainsi, pour voir les messages enregistrés il y a deux minutes, vous devrez taper `sudo journalctl --since "2 minutes ago"`. Il est également possible d'utiliser `+` et `-` pour spécifier des temps relatifs au temps actuel, ainsi `--since "-2 minutes"` et `--since "2 minutes ago"` sont équivalents.

En dehors des expressions numériques, vous pouvez spécifier un certain nombre de mots-clés :

yesterday

À partir de minuit le jour précédant la journée en cours.

today

À partir de minuit la journée en cours.

tomorrow

À partir de minuit après la journée en cours.

now

Maintenant.

Voyons tous les messages depuis minuit jusqu'à 21 heures aujourd'hui :

```
root@debian:~# journalctl --since "today" --until "21:00:00"
-- Logs begin at Sat 2019-10-12 20:45:29 CEST, end at Sat 2019-10-12 21:06:15 CEST. --
Oct 12 20:45:29 debian sudo[1416]:    carol : TTY=pts/0 ; PWD=/home/carol ;
USER=root ; COMMAND=/bin/systemctl r
Oct 12 20:45:29 debian sudo[1416]: pam_unix(sudo:session): session opened for user
root by carol(uid=0)
Oct 12 20:45:29 debian systemd[1]: Stopped Flush Journal to Persistent Storage.
(...)
```

NOTE

Pour en savoir plus sur les différentes syntaxes des spécifications de temps, consultez la page de manuel `systemd.time`.

Programme

Pour voir les messages du journal relatifs à un exécutable donné, la syntaxe suivante est utilisée : `journalctl /path/to/executable` :

```
root@debian:~# journalctl /usr/sbin/sshd
-- Logs begin at Sat 2019-10-12 20:45:29 CEST, end at Sat 2019-10-12 21:54:49 CEST. --
Oct 12 21:16:28 debian sshd[1569]: Accepted password for carol from 192.168.1.65 port
34050 ssh2
Oct 12 21:16:28 debian sshd[1569]: pam_unix(sshd:session): session opened for user carol
by (uid=0)
Oct 12 21:16:54 debian sshd[1590]: Accepted password for carol from 192.168.1.65 port
34052 ssh2
Oct 12 21:16:54 debian sshd[1590]: pam_unix(sshd:session): session opened for user carol
by (uid=0)
```

Unité

Une unité est une ressource gérée par `systemd` et vous pouvez également filtrer par unité.

-u

Cette option affiche les messages relatifs à une unité donnée :

```
root@debian:~# journalctl -u ssh.service
-- Logs begin at Sun 2019-10-13 10:50:59 CEST, end at Sun 2019-10-13 12:22:59 CEST. --
Oct 13 10:51:00 debian systemd[1]: Starting OpenBSD Secure Shell server...
Oct 13 10:51:00 debian sshd[409]: Server listening on 0.0.0.0 port 22.
Oct 13 10:51:00 debian sshd[409]: Server listening on :: port 22.
(...)
```

NOTE

Pour afficher toutes les unités chargées et actives, utilisez `systemctl list-units` ; pour voir tous les fichiers d'unités installés, utilisez `systemctl list-unit-files`.

Champs

Le journal peut également être filtré en fonction de certains *champs* grâce à l'une des syntaxes suivantes :

- `<field-name>=<value>`
- `__<field-name>=<value>_`
- `___<field-name>=<value>`

PRIORITY=

Une des huit valeurs possibles de la priorité `syslog` formatée comme une chaîne décimale :

```
root@debian:~# journalctl PRIORITY=3
-- Logs begin at Sun 2019-10-13 10:50:59 CEST, end at Sun 2019-10-13 14:30:50 CEST. --
Oct 13 10:51:00 debian avahi-daemon[314]: chroot.c: open() failed: No such file or directory
```

Notez que vous auriez pu obtenir le même résultat en utilisant la commande `sudo journalctl -p err` que nous avons vue plus haut.

SYSLOG_FACILITY=

Tout code de ressource possible formaté sous forme de chaîne décimale. Par exemple, pour voir tous les messages du niveau des utilisateurs :

```
root@debian:~# journalctl SYSLOG_FACILITY=1
-- Logs begin at Sun 2019-10-13 10:50:59 CEST, end at Sun 2019-10-13 14:42:52 CEST.
--
Oct 13 10:50:59 debian mtp-probe[227]: checking bus 1, device 2:
"/sys/devices/pci0000:00/0000:00:06.0/usb1/1-1"
Oct 13 10:50:59 debian mtp-probe[227]: bus: 1, device: 2 was not an MTP device
Oct 13 10:50:59 debian mtp-probe[238]: checking bus 1, device 2:
"/sys/devices/pci0000:00/0000:00:06.0/usb1/1-1"
Oct 13 10:50:59 debian mtp-probe[238]: bus: 1, device: 2 was not an MTP device
```

_PID=

Afficher les messages produits par un processus donné. Pour voir tous les messages générés par `systemd`, vous devez taper :

```
root@debian:~# journalctl _PID=1
-- Logs begin at Sun 2019-10-13 10:50:59 CEST, end at Sun 2019-10-13 14:50:15 CEST.
--
Oct 13 10:50:59 debian systemd[1]: Mounted Debug File System.
Oct 13 10:50:59 debian systemd[1]: Mounted POSIX Message Queue File System.
Oct 13 10:50:59 debian systemd[1]: Mounted Huge Pages File System.
Oct 13 10:50:59 debian systemd[1]: Started Remount Root and Kernel File Systems.
Oct 13 10:50:59 debian systemd[1]: Starting Flush Journal to Persistent Storage...
(...)
```

_BOOT_ID

En vous basant sur l'identifiant d'amorçage, vous pouvez isoler les messages provenant d'un amorçage spécifique, par exemple : `sudo journalctl _BOOT_ID=83df3e8653474ea5aed19b41cdb45b78`.

_TRANSPORT

Afficher les messages reçus d'un sous-système spécifique. Les valeurs possibles sont : `audit` (sous-système d'audit du noyau), `driver` (généré en interne), `syslog` (socket `syslog`), `journal` (protocole natif du journal), `stdout` (sortie standard des services ou erreur standard), `kernel` (mémoire tampon circulaire du noyau — la même que `dmesg`, `journalctl -k` ou `journalctl --dmesg`) :

```

root@debian:~# journalctl _TRANSPORT=journal
-- Logs begin at Sun 2019-10-13 20:19:58 CEST, end at Sun 2019-10-13 20:46:36 CEST.
--
Oct 13 20:19:58 debian systemd[1]: Started Create list of required static device
nodes for the current kernel.
Oct 13 20:19:58 debian systemd[1]: Starting Create Static Device Nodes in /dev...
Oct 13 20:19:58 debian systemd[1]: Started Create Static Device Nodes in /dev.
Oct 13 20:19:58 debian systemd[1]: Starting udev Kernel Device Manager...
(...)

```

Combiner les champs

Les champs ne s'excluent pas mutuellement et vous pouvez donc en utiliser plusieurs dans la même requête. En revanche, seuls les messages qui correspondent simultanément à la valeur des deux champs seront affichés :

```

root@debian:~# journalctl PRIORITY=3 SYSLOG_FACILITY=0
-- No entries --
root@debian:~# journalctl PRIORITY=4 SYSLOG_FACILITY=0
-- Logs begin at Sun 2019-10-13 20:19:58 CEST, end at Sun 2019-10-13 20:21:55 CEST. --
Oct 13 20:19:58 debian kernel: acpi PNP0A03:00: fail to add MMCONFIG information, can't
access extended PCI configuration (...)

```

Sauf si vous utilisez le séparateur `+` pour combiner deux expressions à la manière d'un *OU* logique :

```

root@debian:~# journalctl PRIORITY=3 + SYSLOG_FACILITY=0
-- Logs begin at Sun 2019-10-13 20:19:58 CEST, end at Sun 2019-10-13 20:24:02 CEST. --
Oct 13 20:19:58 debian kernel: Linux version 4.9.0-9-amd64 (debian-kernel@lists.debian.org)
(...)
Oct 13 20:19:58 debian kernel: Command line: BOOT_IMAGE=/boot/vmlinuz-4.9.0-9-amd64
root=UUID= (...)
(...)

```

D'autre part, vous pouvez fournir deux valeurs pour le même champ et toutes les entrées correspondant à l'une ou l'autre valeur seront affichées :

```

root@debian:~# journalctl PRIORITY=1
-- Logs begin at Sun 2019-10-13 17:16:24 CEST, end at Sun 2019-10-13 17:30:14 CEST. --
-- No entries --

```



```
root@debian:~# journalctl PRIORITY=1 PRIORITY=3
-- Logs begin at Sun 2019-10-13 17:16:24 CEST, end at Sun 2019-10-13 17:32:12 CEST. --
Oct 13 17:16:27 debian connmand[459]: __connman_inet_get_pnp_nameservers: Cannot read /pro
Oct 13 17:16:27 debian connmand[459]: The name net.connman.vpn was not provided by any .se
```

NOTE

Les champs du journal appartiennent à l'une des catégories suivantes : “Champs du journal de l'utilisateur”, “Champs du journal de confiance”, “Champs du journal du noyau”, “Champs pour le compte d'un programme différent” et “Champs d'adresse”. Pour plus d'informations sur ce sujet — y compris une liste complète des champs — voir la page `man pour systemd.journal-fields(7)`.

Entrées manuelles dans le journal du système : `systemd-cat`

Tout comme la commande `logger` est utilisée pour envoyer des messages de la ligne de commande vers le journal du système (comme nous l'avons vu dans la leçon précédente), la commande `systemd-cat` sert un but similaire — mais plus étoffé — avec le journal du système. Elle nous permet d'envoyer l'entrée standard (*stdin*), la sortie standard (*stdout*) et l'erreur standard (*stderr*) au journal.

Invocé sans paramètres, elle enverra tout ce qu'elle lit dans *stdin* vers le journal. Une fois que vous avez terminé, appuyez sur `Ctrl + C`.

```
carol@debian:~$ systemd-cat
This line goes into the journal.
^C
```

Si le résultat d'une commande lui est transmise par un *pipe*, elle sera également envoyée au journal :

```
carol@debian:~$ echo "And so does this line." | systemd-cat
```

Si elle est suivie d'une commande, la sortie de cette commande sera également envoyée au journal — ainsi que *stderr* (le cas échéant) :

```
carol@debian:~$ systemd-cat echo "And so does this line too."
```

Vous pouvez également spécifier un niveau de priorité avec l'option `-p` :

```
carol@debian:~$ systemd-cat -p emerg echo "This is not a real emergency."
```

Lisez la page de manuel de `systemd-cat` pour en savoir plus sur les autres options.

Pour voir les quatre dernières lignes du journal :

```
carol@debian:~$ journalctl -n 4
(...)
-- Logs begin at Sun 2019-10-20 13:43:54 CEST. --
Nov 13 23:14:39 debian cat[1997]: This line goes into the journal.
Nov 13 23:19:16 debian cat[2027]: And so does this line.
Nov 13 23:23:21 debian echo[2030]: And so does this line too.
Nov 13 23:26:48 debian echo[2034]: This is not a real emergency.
```

NOTE

Les entrées de journal dont le niveau de priorité est *emergency* s'affichent en gros caractères rouges sur la plupart des systèmes.

Stockage persistant du journal

Comme nous l'avons déjà dit, vous avez trois choix pour l'emplacement du journal :

- La journalisation peut être complètement désactivée (la redirection vers d'autres ressources comme la console est toujours possible).
- Le garder en mémoire — ce qui le rend volatile — et vous débarrasser des journaux à chaque réamorçage du système. Dans ce scénario, le répertoire `/run/log/journal` sera créé et utilisé.
- Le rendre persistant de manière à ce que les journaux soient écrits sur le disque. Dans ce cas, les messages de journalisation iront dans le répertoire `/var/log/journal`.

Voici le comportement par défaut : si `/var/log/journal/` n'existe pas, les journaux seront sauvegardés de manière éphémère dans un répertoire sous `/run/log/journal/` et—par conséquent—perdus au réamorçage. Le nom du répertoire—le `/etc/machine-id`—est une chaîne de 32 caractères hexadécimaux en minuscules, terminée par un retour chariot :

```
carol@debian:~$ ls /run/log/journal/8821e1fdf176445697223244d1dfbd73/
system.journal
```

Si vous essayez de le lire avec la commande `less`, vous obtiendrez un avertissement. Utilisez plutôt la commande `journalctl` :

```

root@debian:~# less /run/log/journal/9a32ba45ce44423a97d6397918de1fa5/system.journal
"/run/log/journal/9a32ba45ce44423a97d6397918de1fa5/system.journal" may be a binary file.
See it anyway?
root@debian:~# journalctl
-- Logs begin at Sat 2019-10-05 21:26:38 CEST, end at Sat 2019-10-05 21:31:27 CEST. --
(...)
Oct 05 21:26:44 debian systemd-journald[1712]: Runtime journal
(/run/log/journal/9a32ba45ce44423a97d6397918de1fa5) is 4.9M, max 39.5M, 34.6M free.
Oct 05 21:26:44 debian systemd[1]: Started Journal Service.
(...)
```

Si `/var/log/journal/` existe, les journaux y seront stockés de manière persistante. Si ce répertoire est supprimé, `systemd-journald` ne va pas le recréer mais plutôt écrire dans `/run/log/journal`. Dès que nous recréons `/var/log/journal/` et que nous redémarrons le démon, l'enregistrement persistant sera rétabli :

```

root@debian:~# mkdir /var/log/journal/
root@debian:~# systemctl restart systemd-journald
root@debian:~# journalctl
(...)
Oct 05 21:33:49 debian systemd-journald[1712]: Received SIGTERM from PID 1 (systemd).
Oct 05 21:33:49 debian systemd[1]: Stopped Journal Service.
Oct 05 21:33:49 debian systemd[1]: Starting Journal Service...
Oct 05 21:33:49 debian systemd-journald[1768]: Journal started
Oct 05 21:33:49 debian systemd-journald[1768]: System journal
(/var/log/journal/9a32ba45ce44423a97d6397918de1fa5) is 8.0M, max 1.1G, 1.1G free.
Oct 05 21:33:49 debian systemd[1]: Started Journal Service.
Oct 05 21:33:49 debian systemd[1]: Starting Flush Journal to Persistent Storage...
(...)
```

NOTE

Par défaut, vous aurez des fichiers journaux spécifiques pour chaque utilisateur connecté, situés dans `/var/log/journal/`, donc—en plus des fichiers `system.journal`—vous trouverez également des fichiers du genre `user-1000.journal`.

En plus de ce que nous venons de voir, la façon dont le démon de journalisation gère le stockage des journaux peut être modifiée après l'installation en adaptant son fichier de configuration : `/etc/systemd/journald.conf`. L'option clé est `Storage=` et elle peut prendre les valeurs suivantes :

Storage=volatile

Les données du journal seront stockées en mémoire exclusivement — sous `/run/log/journal/`. S'il n'est pas présent, le répertoire sera créé.

Storage=persistent

Par défaut, les données du journal seront stockées sur le disque — sous `/var/log/journal/` — avec un basculement vers la mémoire (`/run/log/journal/`) pendant les premières phases d'amorçage et si le disque n'est pas accessible en écriture. Les deux répertoires seront créés si nécessaire.

Storage=auto

auto fonctionne comme persistent, mais le répertoire `/var/log/journal` n'est pas créé en cas de besoin. C'est la valeur par défaut.

Storage=none

Toutes les données de journalisation seront rejetées. Le transfert vers d'autres ressources comme la console, la mémoire tampon circulaire du noyau ou une socket `syslog` est toujours possible.

Par exemple, pour que `systemd-journald` crée `/var/log/journal/` et passe à un stockage persistant, vous devez éditer `/etc/systemd/journald.conf` et mettre `Storage=persistent`, sauvegarder le fichier et redémarrer le démon avec `sudo systemctl restart systemd-journald`. Pour vous assurer que le redémarrage s'est déroulé sans problème, vous pouvez toujours vérifier l'état du service :

```
root@debian:~# systemctl status systemd-journald
systemd-journald.service - Journal Service
  Loaded: loaded (/lib/systemd/system/systemd-journald.service; static; vendor preset:
enabled)
  Active: active (running) since Wed 2019-10-09 10:03:40 CEST; 2s ago
    Docs: man:systemd-journald.service(8)
          man:journald.conf(5)
 Main PID: 1872 (systemd-journal)
  Status: "Processing requests..."
   Tasks: 1 (limit: 3558)
  Memory: 1.1M
   CGroup: /system.slice/systemd-journald.service
           └─1872 /lib/systemd/systemd-journald

Oct 09 10:03:40 debian10 systemd-journald[1872]: Journal started
Oct 09 10:03:40 debian10 systemd-journald[1872]: System journal
```

```
(/var/log/journal/9a32ba45ce44423a97d6397918de1fa5) is 8.0M, max 1.2G, 1.2G free.
```

NOTE

Les fichiers journaux dans `/var/log/journal/<machine-id>/` ou `/run/log/journal/<machine-id>/` ont le suffixe `.journal` (par exemple `system.journal`). Cependant, s'ils sont corrompus ou si le démon est arrêté de façon irrégulière, ils seront renommés en ajoutant `~` (par exemple `system.journal~`) et le démon commencera à écrire dans un nouveau fichier propre.

Supprimer les anciennes données du journal : taille du journal

Les journaux sont enregistrés dans des *fichiers journaux* dont les noms se terminent par `.journal` ou `.journal~` et situés dans le répertoire approprié (`/run/log/journal` ou `/var/log/journal` selon la configuration). Pour vérifier l'espace disque occupé par les fichiers journaux (archivés et actifs), utilisez l'option `--disk-usage` :

```
root@debian:~# journalctl --disk-usage
Archived and active journals take up 24.0M in the filesystem.
```

Les journaux `systemd` sont limités par défaut à 10% de la taille du système de fichiers dans lequel ils sont enregistrés. Par exemple, sur un système de fichiers de 1 Go, ils ne prendront pas plus de 100 Mo. Une fois cette limite atteinte, les anciens journaux commenceront à disparaître pour rester proche de cette valeur.

En revanche, on peut limiter la taille des fichiers journaux stockés en adaptant une série d'options de configuration dans le fichier `/etc/systemd/journald.conf`. Ces options se déclinent en deux catégories selon le type de système de fichiers utilisé : persistant (`/var/log/journal`) ou en mémoire (`/run/log/journal`). La première catégorie utilise des options préfixées par le mot `System` et ne s'applique que si la journalisation persistante est correctement activée et une fois que le système est entièrement amorcé. Les noms des options de la seconde commencent par le mot `Runtime` et vont s'appliquer dans les scénarios suivants :

SystemMaxUse=, RuntimeMaxUse=

Ces options déterminent la quantité d'espace disque qui peut être occupée par le journal. La valeur par défaut est de 10% de la taille du système de fichiers, mais elle peut être modifiée (par exemple `SystemMaxUse=500M`) tant qu'elle ne dépasse pas un maximum de 4 Gio.

SystemKeepFree=, RuntimeKeepFree=

Ces options déterminent la quantité d'espace disque qui doit être laissée libre pour les autres utilisateurs. Elle est fixée par défaut à 15% de la taille du système de fichiers mais peut être

modifiée (par exemple `SystemKeepFree=500M`) tant qu'elle ne dépasse pas un maximum de 4 Gio.

En ce qui concerne la priorité de `*MaxUse` et `*KeepFree`, `systemd-journald` va satisfaire les deux en utilisant la plus petite des deux valeurs. De même, gardez à l'esprit que seuls les fichiers journaux archivés (par opposition aux fichiers actifs) seront supprimés.

SystemMaxFileSize=, RuntimeMaxFileSize=

Ces options déterminent la taille maximale que peuvent atteindre les fichiers individuels du journal. La valeur par défaut est 1/8 de `*MaxUse`. La réduction de taille s'effectue de manière synchrone et les valeurs peuvent être spécifiées en octets ou en utilisant K, M, G, T, P, E pour kibioctet, mébioctet, gibioctet, tébioctet, pébioctet et exbioctet, respectivement.

SystemMaxFiles=, RuntimeMaxFiles=

Ces options déterminent le nombre maximum de fichiers journaux individuels et archivés à stocker (les fichiers journaux actifs ne sont pas affectés). La valeur par défaut est 100.

En dehors de la suppression et de la rotation des messages de journalisation basées sur la taille, `systemd-journald` permet également des critères basés sur le temps en utilisant les deux options suivantes : `MaxRetentionSec=` et `MaxFileSec=`. Reportez-vous à la page de manuel de `journald.conf` pour plus d'informations sur ces options et bien d'autres.

NOTE

Chaque fois que vous modifiez le comportement par défaut de `systemd-journald` en décommentant et en éditant des options dans `/etc/systemd/journald.conf`, vous devez redémarrer le démon pour que les changements prennent effet.

Faire le ménage dans le journal

Vous pouvez à tout moment nettoyer manuellement les fichiers journaux archivés à l'aide de l'une des trois options suivantes :

--vacuum-time=

Cette option basée sur le temps supprime tous les messages des fichiers journaux dont l'horodatage est antérieur à la période spécifiée. Les valeurs devront être écrites avec l'un des suffixes suivants : s, m, h, days (ou d), months, weeks (ou w) et years (ou y). Par exemple, pour vous débarrasser de tous les messages dans les fichiers journaux archivés qui datent de plus d'un mois :

```
root@debian:~# journalctl --vacuum-time=1months
Deleted archived journal
/var/log/journal/7203088f20394d9c8b252b64a0171e08/system@27dd08376f71405a91794e632ede97ed
```

```
-0000000000000001-00059475764d46d6.journal (16.0M).
Deleted archived journal /var/log/journal/7203088f20394d9c8b252b64a0171e08/user-
1000@e7020d80d3af42f0bc31592b39647e9c-000000000000008e-00059479df9677c8.journal (8.0M).
```

--vacuum-size=

Cette option basée sur la taille supprimera les fichiers journaux archivés jusqu'à ce qu'ils occupent une valeur inférieure à la taille spécifiée. Les valeurs devront être écrites avec l'un des suffixes suivants : K, M, G ou T. Par exemple, pour éliminer les fichiers journaux archivés jusqu'à ce qu'ils soient inférieurs à 100 mébiotets :

```
root@debian:~# journalctl --vacuum-size=100M
Vacuuming done, freed 0B of archived journals from
/run/log/journal/9a32ba45ce44423a97d6397918de1fa5.
```

--vacuum-files=

Cette option veille à ce qu'il ne reste pas plus de fichiers journaux archivés que le nombre spécifié. La valeur spécifiée est un nombre entier. Par exemple, pour limiter le nombre de fichiers journaux archivés à 10 :

```
root@debian:~# journalctl --vacuum-files=10
Vacuuming done, freed 0B of archived journals from
/run/log/journal/9a32ba45ce44423a97d6397918de1fa5.
```

Le nettoyage supprime uniquement les fichiers journaux archivés. Si vous souhaitez vous débarrasser de tout (y compris des fichiers journaux actifs), vous devrez utiliser un signal (SIGUSR2) qui demande la rotation immédiate des fichiers journaux avec l'option `--rotate`. D'autres signaux importants peuvent être invoqués avec les options suivantes :

--flush (SIGUSR1)

Cette option lance le transfert des fichiers du journal de `/run/` vers `/var/` pour rendre le journal persistant. Elle nécessite l'activation de la journalisation persistante et le montage de `/var/`.

--sync (SIGRTMIN+1)

Cette option est utilisée pour déclencher l'écriture de toutes les données non enregistrées du journal sur le disque.

NOTE

Pour vérifier la cohérence interne du fichier journal, utilisez `journalctl` avec l'option `--verify`. Vous verrez une barre de progression au fur et à mesure de la

vérification et les problèmes éventuels seront affichés.

Récupérer les données d'un journal à partir d'un système de secours

En tant qu'administrateur système, vous pouvez vous retrouver dans une situation où vous devez accéder à des fichiers journaux sur le disque dur d'une machine défectueuse par le biais d'un système de secours (un CD amorçable ou une clé USB contenant une distribution Linux live).

`journalctl` recherche les fichiers journaux dans `/var/log/journal/<machine-id>/`. Étant donné que les identifiants de machine sur le système de secours et le système défectueux seront différents, vous devrez utiliser l'option suivante :

-D </path/to/dir>, --directory=</path/to/dir>

Avec cette option, nous spécifions le chemin du répertoire dans lequel `journalctl` cherchera les fichiers journaux au lieu des emplacements par défaut du système en cours d'exécution.

Il est donc nécessaire de monter la racine du système défectueux (`/dev/sda1`) sur le système de fichiers de secours pour y lire les fichiers journaux comme ceci :

```
root@debian:~# journalctl -D /media/carol/faulty.system/var/log/journal/
-- Logs begin at Sun 2019-10-20 12:30:45 CEST, end at Sun 2019-10-20 12:32:57 CEST. --
oct 20 12:30:45 suse-server kernel: Linux version 4.12.14-lp151.28.16-default
(geeko@buildhost) (...)
oct 20 12:30:45 suse-server kernel: Command line: BOOT_IMAGE=/boot/vmlinuz-4.12.14-
lp151.28.16-default root=UUID=7570f67f-4a08-448e-aa09-168769cb9289 splash=>
oct 20 12:30:45 suse-server kernel: x86/fpu: Supporting XSAVE feature 0x001: 'x87 floating
point registers'
oct 20 12:30:45 suse-server kernel: x86/fpu: Supporting XSAVE feature 0x002: 'SSE registers'
(...)
```

D'autres options peuvent être utiles dans ce contexte :

-m, --merge

Cette option fusionne les entrées de tous les journaux disponibles dans `/var/log/journal`, y compris les journaux distants.

--file

Cette option va afficher les entrées dans un fichier donné, par exemple : `journalctl --file /var/log/journal/64319965bda04dfa81d3bc4e7919814a/user-1000.journal`.

--root

Un chemin signifiant que le répertoire racine est passé en argument. `journalctl` y recherchera les fichiers journaux (par exemple `journalctl --root /faulty.system/`).

Consultez la page de manuel `journalctl` pour plus d'informations.

Transférer les données de journalisation vers un démon syslog traditionnel

Les données du journal peuvent être mises à disposition d'un démon `syslog` traditionnel :

- En transférant les messages vers le fichier socket `/run/systemd/journal/syslog` pour que `syslogd` les lise. Cette fonctionnalité est activée avec l'option `ForwardToSyslog=yes`.
- Avec un démon `syslog` qui se comporte comme `journalctl` et qui lit les messages de journalisation directement depuis les fichiers journaux. Dans ce cas, l'option pertinente est celle de `Storage` ; elle doit avoir une valeur autre que `none`.

NOTE

De même, vous pouvez transférer les messages de journalisation vers d'autres destinations avec les options suivantes : `ForwardToKMsg` (mémoire tampon du journal du noyau *kernel log buffer* — `kmsg`), `ForwardToConsole` (la console du système) ou `ForwardToWall` (tous les utilisateurs connectés via `wall`). Pour plus d'informations, consultez la page de manuel de `journald.conf`.

Exercices guidés

1. En supposant que vous soyez `root`, complétez le tableau avec les commandes `journalctl` appropriées :

Objectif	Commande
Afficher les messages du noyau	
Afficher les messages du deuxième amorçage depuis le début du journal	
Afficher les messages du deuxième amorçage depuis la fin du journal	
Afficher les messages de journal les plus récents et continuer à surveiller les nouveaux messages	
Afficher uniquement les nouveaux messages depuis maintenant et mettre à jour les résultats en permanence	
Afficher les messages de l'amorçage précédent avec un niveau de priorité <code>warning</code> et dans l'ordre inverse	

2. Le comportement du démon de journalisation en ce qui concerne le stockage est principalement contrôlé par la valeur de l'option `Storage` dans `/etc/systemd/journald.conf`. Indiquez quel comportement est lié à quelle valeur dans le tableau suivant :

Comportement	<code>Storage=auto</code>	<code>Storage=none</code>	<code>Storage=persistent</code>	<code>Storage=volatile</code>
Les données du journal sont jetées, mais le transfert est possible.				

Comportement	Storage=auto	Storage=none	Storage=persistent	Storage=volatile
Une fois le système amorcé, les données du journal seront stockées dans <code>/var/log/journal</code> . S'il n'est pas déjà présent, le répertoire sera créé.				
Une fois le système amorcé, les données du journal seront stockées dans <code>/var/log/journal</code> . S'il n'est pas déjà présent, le répertoire ne sera pas créé.				
Les données du journal seront stockées dans <code>/var/run/journal</code> , mais ne seront pas conservées après un réamorçage.				

3. Comme vous l'avez appris, le journal peut être vidé manuellement en fonction du temps, de la taille et du nombre de fichiers. Effectuez les tâches suivantes en utilisant `journalctl` avec les options appropriées :

- Vérifiez l'espace disque occupé par les fichiers du journal :

- Réduisez l'espace réservé aux fichiers journaux archivés et fixez-le à 200 Mio :

- Vérifiez à nouveau l'espace disque et expliquez le résultat :

Exercices d'approfondissement

1. Quelles options devez-vous modifier dans `/etc/systemd/journald.conf` pour que les messages soient transférés vers `/dev/tty5` ? Quelles sont les valeurs à donner à ces options ?

2. Fournissez le filtre `journalctl` correct pour afficher ce qui suit :

Objectif	Filtre + Valeur
Afficher les messages appartenant à un utilisateur spécifique	
Afficher les messages provenant d'un hôte nommé <code>debian</code>	
Afficher les messages appartenant à un groupe spécifique	
Afficher les messages appartenant à <code>root</code>	
En fonction du chemin d'accès de l'exécutable, afficher les messages <code>sudo</code>	
En fonction du nom de la commande, afficher les messages <code>sudo</code>	

3. Lors du filtrage par priorité, les journaux dont la priorité est supérieure à celle indiquée seront également inclus dans la liste ; par exemple `journalctl -p err` affichera les messages *error*, *critical*, *alert* et *emergency*. Cependant, vous pouvez demander à `journalctl` de n'afficher qu'une plage spécifique. Quelle commande utiliseriez-vous pour que `journalctl` n'affiche que les messages des niveaux de priorité *warning*, *error* et *critical* ?

4. Les niveaux de priorité peuvent également être spécifiés de manière numérique. Réécrivez la commande de l'exercice précédent en utilisant la notation numérique des niveaux de priorité :

Résumé

Voici ce que nous avons vu dans cette leçon :

- Les avantages liés au choix de `systemd` comme gestionnaire de services.
- Les fondamentaux des unités et des cibles `systemd`.
- D'où `systemd-journald` obtient les données de journalisation.
- Les options que vous pouvez passer à `systemctl` pour contrôler `systemd-journald` : `start`, `status`, `restart` et `stop`.
- L'emplacement du fichier de configuration du journal — `/etc/systemd/journald.conf` — et ses principales options.
- Comment interroger le journal d'une manière générale et pour des données spécifiques avec des filtres.
- Comment naviguer dans le journal et effectuer des recherches.
- Comment gérer le stockage des fichiers journaux : en mémoire ou sur le disque.
- Comment désactiver complètement la journalisation.
- Comment vérifier l'espace disque occupé par le journal, appliquer des limites de taille aux fichiers journaux stockés et nettoyer manuellement les fichiers journaux archivés (*vacuuming*).
- Comment récupérer les données du journal à partir d'un système de secours.
- Comment transférer les données de journalisation à un démon `syslog` traditionnel.

Voici les commandes utilisées dans cette leçon :

`systemctl`

Contrôle le gestionnaire de système et de services `systemd`.

`journalctl`

Interroge le journal de `systemd`.

`ls`

Affiche le contenu d'un répertoire.

`less`

Affiche le contenu d'un fichier.

mkdir

Crée un répertoire.

Réponses aux exercices guidés

1. En supposant que vous soyez `root`, complétez le tableau avec les commandes `journalctl` appropriées :

Objectif	Commande
Afficher les messages du noyau	<code>journalctl -k</code> ou <code>journalctl --dmesg</code>
Afficher les messages du deuxième amorçage depuis le début du journal	<code>journalctl -b 2</code>
Afficher les messages du deuxième amorçage depuis la fin du journal	<code>journalctl -b -2 -r</code> or <code>journalctl -r -b -2</code>
Afficher les messages de journal les plus récents et continuer à surveiller les nouveaux messages	<code>journalctl -f</code>
Afficher uniquement les nouveaux messages depuis maintenant et mettre à jour les résultats en permanence	<code>journalctl --since "now" -f</code>
Afficher les messages de l'amorçage précédent avec un niveau de priorité <code>warning</code> et dans l'ordre inverse	<code>journalctl -b -1 -p warning -r</code>

2. Le comportement du démon de journalisation en ce qui concerne le stockage est principalement contrôlé par la valeur de l'option `Storage` dans `/etc/systemd/journald.conf`. Indiquez quel comportement est lié à quelle valeur dans le tableau suivant :

Comportement	<code>Storage=auto</code>	<code>Storage=none</code>	<code>Storage=persistent</code>	<code>Storage=volatile</code>
Les données du journal sont jetées, mais le transfert est possible		X		

Comportement	Storage=auto	Storage=none	Storage=persistent	Storage=volatile
Une fois le système amorcé, les données du journal seront stockées dans <code>/var/log/journal</code> . S'il n'est pas déjà présent, le répertoire sera créé			x	
Une fois le système amorcé, les données du journal seront stockées dans <code>/var/log/journal</code> . S'il n'est pas déjà présent, le répertoire ne sera pas créé	x			
Les données du journal seront stockées dans <code>/var/run/journal</code> , mais ne seront pas conservées après un réamorçage				x

3. Comme vous l'avez appris, le journal peut être vidé manuellement en fonction du temps, de la taille et du nombre de fichiers. Effectuez les tâches suivantes en utilisant `journalctl` avec les options appropriées :

- Vérifiez l'espace disque occupé par les fichiers du journal :

```
journalctl --disk-usage
```

- Réduisez l'espace réservé aux fichiers journaux archivés et fixez-le à 200 Mio :

```
journalctl --vacuum-size=200M
```

- Vérifiez à nouveau l'espace disque et expliquez le résultat :

```
journalctl --disk-usage
```

Il n'y a pas de corrélation car `--disk-usage` affiche l'espace occupé par les fichiers journaux actifs et archivés alors que `--vacuum-size` ne s'applique qu'aux seuls fichiers archivés.

Réponses aux exercices d'approfondissement

1. Quelles options devez-vous modifier dans `/etc/systemd/journald.conf` pour que les messages soient transférés vers `/dev/tty5` ? Quelles sont les valeurs à donner à ces options ?

```
ForwardToConsole=yes
TTYPath=/dev/tty5
```

2. Fournissez le filtre `journalctl` correct pour afficher ce qui suit :

Objectif	Filtre + Valeur
Afficher les messages appartenant à un utilisateur spécifique	<code>_ID=<user-id></code>
Afficher les messages provenant d'un hôte nommé <code>debian</code>	<code>_HOSTNAME=debian</code>
Afficher les messages appartenant à un groupe spécifique	<code>_GID=<group-id></code>
Afficher les messages appartenant à <code>root</code>	<code>_UID=0</code>
En fonction du chemin d'accès de l'exécutable, afficher les messages <code>sudo</code>	<code>_EXE=/usr/bin/sudo</code>
En fonction du nom de la commande, afficher les messages <code>sudo</code>	<code>_COMM=sudo</code>

3. Lors du filtrage par priorité, les journaux dont la priorité est supérieure à celle indiquée seront également inclus dans la liste ; par exemple `journalctl -p err` affichera les messages *error*, *critical*, *alert* et *emergency*. Cependant, vous pouvez demander à `journalctl` de n'afficher qu'une plage spécifique. Quelle commande utiliseriez-vous pour que `journalctl` n'affiche que les messages des niveaux de priorité *warning*, *error* et *critical* ?

```
journalctl -p warning..crit
```

4. Les niveaux de priorité peuvent également être spécifiés de manière numérique. Réécrivez la commande de l'exercice précédent en utilisant la notation numérique des niveaux de priorité :

```
journalctl -p 4..2
```



108.3 Bases sur l'agent de transfert de courrier (MTA)

Référence aux objectifs de LPI

LPIC-1 version 5.0, Exam 102, Objective 108.3

Valeur

3

Domaines de connaissance les plus importants

- Création des alias de courriel.
- Transfert de courriel (forward).
- Connaissance des principaux serveurs SMTP (postfix, sendmail, et exim) (sans configuration).

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

- `~/ .forward`
- couche d'émulation des commandes sendmail
- `newaliases`
- `mail`
- `mailq`
- `postfix`
- `sendmail`
- `exim`



108.3 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	108 Services Systèmes Essentiels
Objectif :	108.3 Fonctionnement de l'agent de transfert de courrier (MTA)
Leçon :	1 sur 1

Introduction

Dans les systèmes d'exploitation de type Unix, comme Linux, chaque utilisateur possède sa propre boîte de réception (*inbox*) : un emplacement particulier du système de fichiers, inaccessible aux autres utilisateurs non root, et où sont stockés les messages électroniques personnels de l'utilisateur. Les nouveaux messages entrants sont ajoutés à la boîte de réception de l'utilisateur par l'agent de transfert de courrier (MTA ou *Mail Transfer Agent*). Le MTA est un programme qui fonctionne en tant que service système et qui collecte les messages envoyés par d'autres comptes locaux ainsi que les messages reçus depuis le réseau, envoyés par des comptes d'utilisateurs distants.

Le même MTA est également chargé d'envoyer des messages vers le réseau si l'adresse de destination se réfère à un compte distant. Pour ce faire, il utilise un emplacement du système de fichiers comme boîte d'envoi de courrier électronique (*outbox*) pour tous les utilisateurs du système : dès qu'un utilisateur place un nouveau message dans cette boîte d'envoi, le MTA identifie le nœud du réseau cible à partir du nom de domaine fourni par l'adresse électronique de destination — la partie située après le signe @ — puis il essaie de transférer le message au MTA distant à l'aide du protocole simple de transfert de courrier (SMTP ou *Simple Mail Transfer*

Protocol). Le protocole SMTP a été conçu en gardant à l'esprit les réseaux qui ne sont pas fiables, de sorte qu'il tente d'établir des itinéraires de livraison alternatifs si jamais le nœud principal de destination du courrier n'est pas accessible.

MTA local et distant

Les comptes utilisateurs traditionnels sur des machines en réseau constituent le scénario le plus basique pour l'échange de courrier électronique, dans lequel chaque nœud du réseau exécute son propre démon MTA. Aucun logiciel autre que le MTA n'est nécessaire pour envoyer et recevoir des messages électroniques. En pratique, il est cependant plus courant d'utiliser un compte de courrier électronique distant sans disposer d'un service MTA local actif (c'est-à-dire en utilisant plutôt une application client de courrier électronique pour accéder au compte distant).

Contrairement aux comptes locaux, un compte de messagerie distant — également appelé *boîte aux lettres distante* — nécessite l'authentification de l'utilisateur pour lui permettre d'accéder à sa boîte aux lettres et au MTA distant (dans ce cas, simplement appelé *serveur SMTP*). Alors que l'utilisateur qui interagit avec une boîte de réception locale et un MTA est déjà identifié par le système, un système distant doit vérifier l'identité de l'utilisateur avant de traiter ses messages via IMAP ou POP3.

NOTE

De nos jours, la méthode la plus courante pour envoyer et recevoir du courrier électronique consiste à utiliser un compte hébergé sur un serveur distant, par exemple le serveur de courrier électronique centralisé d'une entreprise qui héberge tous les comptes des employés ou un service de courrier électronique personnel, comme *Gmail* de Google. Au lieu de collecter les messages livrés localement, l'application client de messagerie se connecte à la boîte aux lettres distante et récupère les messages à partir de là. Les protocoles POP3 et IMAP sont généralement utilisés pour récupérer les messages sur le serveur distant, mais d'autres protocoles propriétaires peuvent également être utilisés.

Lorsqu'un démon MTA fonctionne sur le système local, les utilisateurs locaux peuvent envoyer leur courrier électronique à d'autres utilisateurs locaux ou à des utilisateurs sur une machine distante, à condition que leur système dispose également d'un service MTA qui accepte les connexions réseau. Le port TCP 25 est le port standard pour la communication SMTP, mais d'autres ports peuvent également être utilisés, en fonction du schéma d'authentification et/ou de chiffrement utilisé (le cas échéant).

Si l'on fait abstraction des topologies qui permettent d'accéder à des boîtes aux lettres distantes, il est possible de mettre en place un réseau d'échange de courrier électronique entre des comptes utilisateurs Linux ordinaires, à condition que tous les nœuds du réseau disposent d'un MTA actif capable d'effectuer les tâches suivantes :

- Gérer la file d'attente de la boîte d'envoi des messages à envoyer. Pour chaque message dans la file d'attente, le MTA local évalue le MTA de destination à partir de l'adresse du destinataire.
- Communiquer avec des démons MTA distants à l'aide du protocole SMTP. Le MTA local doit pouvoir utiliser le protocole SMTP (*Simple Mail Transfer Protocol*) via la pile TCP/IP pour recevoir, envoyer et rediriger des messages depuis/vers d'autres démons MTA distants.
- Maintenir une boîte de réception individuelle pour chaque compte local. Le MTA stocke généralement les messages au format *mbox* : un fichier texte unique qui contient tous les messages électroniques dans l'ordre.

En temps normal, les adresses de courrier électronique spécifient un nom de domaine comme emplacement, par exemple `lpi.org` dans `info@lpi.org`. Dans ce cas, le MTA de l'expéditeur interroge le service DNS pour obtenir l'enregistrement MX correspondant. L'enregistrement DNS MX contient l'adresse IP du MTA qui traite le courrier électronique pour ce domaine. Si le même domaine a plus d'un enregistrement MX spécifié dans le DNS, le MTA doit essayer de les contacter en fonction de leurs valeurs de priorité. Si l'adresse du destinataire ne spécifie pas de nom de domaine ou si le domaine n'a pas d'enregistrement MX, la partie après le symbole `@` sera traitée comme l'hôte du MTA de destination.

La sécurité doit être prise en compte si les hôtes MTA sont visibles sur Internet. Par exemple, il est possible pour un utilisateur inconnu d'utiliser le MTA local pour se faire passer pour un autre utilisateur et envoyer des courriels potentiellement malveillants. Un MTA qui relaie à l'aveugle un courrier électronique est connu sous le nom de « relais ouvert » (*open relay*) lorsqu'il peut être utilisé comme intermédiaire pour dissimuler l'expéditeur réel du message. Pour éviter ce genre d'abus, il est recommandé d'accepter uniquement les connexions provenant de domaines autorisés et de mettre en œuvre un schéma d'authentification sécurisé.

Par ailleurs, il existe toute une série d'implémentations de MTA pour Linux, chacune se concentrant sur des aspects spécifiques comme la compatibilité, les performances, la sécurité, etc. Ceci étant dit, tous les MTA respectent les mêmes principes de base et offrent des fonctionnalités similaires.

MTAs pour Linux

Le MTA traditionnel pour les systèmes Linux est *Sendmail*, un MTA polyvalent très flexible utilisé par la plupart des systèmes d'exploitation de type Unix. D'autres MTA populaires sont *Postfix*, *qmail* et *Exim*. La principale raison de choisir un autre MTA est la possibilité d'implémenter plus facilement des fonctionnalités avancées, étant donné que la configuration de serveurs de messagerie personnalisés dans *Sendmail* peut s'avérer complexe. Par ailleurs, chaque distribution peut avoir son MTA préféré, avec des paramètres prédéfinis adaptés à la plupart des configurations courantes. Tous les MTA ont vocation à remplacer *Sendmail*, de sorte que toutes les

applications compatibles avec Sendmail devraient fonctionner indépendamment du MTA utilisé.

Si le MTA fonctionne mais qu'il n'accepte pas les connexions réseau, il ne pourra délivrer les messages que sur la machine locale. Pour le MTA `sendmail`, le fichier `/etc/mail/sendmail.mc` devra être modifié pour accepter les connexions non-locales. Pour ce faire, l'entrée

```
DAEMON_OPTIONS(`Port=smtp,Addr=127.0.0.1, Name=MTA')dnl
```

devra être modifiée pour indiquer l'adresse réseau correcte et le service devra être redémarré. Certaines distributions Linux, comme Debian, peuvent fournir des outils de configuration pour aider à lancer le serveur de messagerie avec un ensemble prédéfini de fonctionnalités couramment utilisées.

TIP

Pour des raisons de sécurité, la plupart des distributions Linux n'installent pas de MTA par défaut. Pour tester les exemples donnés dans cette leçon, assurez-vous qu'il y a un MTA en cours d'exécution sur chaque machine et qu'ils acceptent les connexions sur le port TCP 25. Par mesure de sécurité, ces systèmes ne devront pas être exposés à des connexions entrantes en provenance de l'internet public pendant les tests.

Une fois que le MTA fonctionne et accepte les connexions sur le réseau, les nouveaux messages électroniques lui sont transmis par des commandes SMTP envoyées par le biais d'une connexion TCP. La commande `nc` — un utilitaire qui lit et écrit des données génériques sur le réseau — pourra être utilisée pour envoyer des commandes SMTP directement au MTA. Si la commande `nc` n'est pas disponible, elle peut être installée avec le paquet `ncat` ou `nmap-ncat`, selon le système de gestion de paquets utilisé. Envoyer des commandes SMTP directement au MTA vous aidera à mieux comprendre le protocole et d'autres concepts généraux du courrier électronique, mais cela peut aussi vous aider à diagnostiquer des problèmes dans le processus de distribution du courrier.

Si, par exemple, l'utilisatrice `emma` sur l'hôte `lab1.campus` veut envoyer un message à l'utilisateur `dave` sur l'hôte `lab2.campus`, elle pourra utiliser la commande `nc` pour se connecter directement au MTA `lab2.campus`, en supposant qu'il écoute sur le port TCP 25 :

```
$ nc lab2.campus 25
220 lab2.campus ESMTP Sendmail 8.15.2/8.15.2; Sat, 16 Nov 2019 00:16:07 GMT
HELO lab1.campus
250 lab2.campus Hello lab1.campus [10.0.3.134], pleased to meet you
MAIL FROM: emma@lab1.campus
250 2.1.0 emma@lab1.campus... Sender ok
RCPT TO: dave@lab2.campus
```



```

250 2.1.5 dave@lab2.campus... Recipient ok
DATA
354 Enter mail, end with "." on a line by itself
Subject: Recipient MTA Test

Hi Dave, this is a test for your MTA.
.
250 2.0.0 xAG0G7Y0000595 Message accepted for delivery
QUIT
221 2.0.0 lab2.campus closing connection

```

Une fois la connexion établie, le MTA distant s'identifie et peut recevoir des commandes SMTP. La première commande SMTP de l'exemple, `HELO lab1.campus`, indique que `lab1.campus` a initié l'échange. Les deux commandes suivantes, `MAIL FROM : emma@lab1.campus` et `RCPT TO : dave@lab2.campus`, indiquent l'expéditeur et le destinataire. Un message électronique commence après la commande `DATA` et se termine par un point sur une ligne séparée. Pour ajouter un champ `Sujet` à l'e-mail, il doit se trouver sur la première ligne après la commande `DATA`, comme indiqué dans l'exemple. Lorsque le champ `Sujet` est utilisé, il doit y avoir une ligne vide qui le sépare du contenu du message. La commande `QUIT` ferme la connexion avec le MTA sur l'hôte `lab2.campus`.

Sur l'hôte `lab2.campus`, l'utilisateur `dave` recevra un message du type `You have new mail in /var/spool/mail/dave` dès qu'il ouvrira une session shell. Ce fichier contiendra le message brut envoyé par `emma` ainsi que les en-têtes ajoutés par le MTA :

```

$ cat /var/spool/mail/dave
From emma@lab1.campus Sat Nov 16 00:19:13 2019
Return-Path: <emma@lab1.campus>
Received: from lab1.campus (lab1.campus [10.0.3.134])
    by lab2.campus (8.15.2/8.15.2) with SMTP id xAG0G7Y0000595
    for dave@lab2.campus; Sat, 16 Nov 2019 00:17:06 GMT
Date: Sat, 16 Nov 2019 00:16:07 GMT
From: emma@lab1.campus
Message-Id: <201911160017.xAG0G7Y0000595@lab2.campus>
Subject: Recipient MTA Test

Hi Dave, this is a test for your MTA.

```

L'en-tête `Received:` montre que le message de `lab1.campus` a été reçu directement par `lab2.campus`. Par défaut, les MTA n'acceptent que les messages destinés à des utilisateurs locaux. L'erreur suivante se produira probablement si l'utilisatrice `emma` essaie d'envoyer un e-mail à

l'utilisateur `henry` sur l'hôte `lab3.campus`, mais en utilisant le MTA `lab2.campus` au lieu du MTA `lab3.campus` approprié :

```
$ nc lab2.campus 25
220 lab2.campus ESMTP Sendmail 8.15.2/8.15.2; Sat, 16 Nov 2019 00:31:44 GMT
HELO lab1.campus
250 lab2.campus Hello lab1.campus [10.0.3.134], pleased to meet you
MAIL FROM: emma@lab1.campus
250 2.1.0 emma@lab1.campus... Sender ok
RCPT TO: henry@lab3.campus
550 5.7.1 henry@lab3.campus... Relaying denied
```

Les codes de réponse SMTP qui commencent par 5, comme le message `Relaying denied`, indiquent une erreur. Il existe des situations légitimes où le relais est souhaitable, par exemple lorsque les hôtes qui envoient et reçoivent des courriels ne sont pas connectés en permanence : un MTA intermédiaire pourra être configuré pour accepter les e-mails destinés à d'autres hôtes, agissant comme un serveur SMTP relais capable de transférer les messages entre les MTA.

La possibilité d'acheminer le trafic de courrier électronique via des serveurs SMTP intermédiaires décourage les tentatives de connexion directe à l'hôte indiqué par l'adresse électronique du destinataire, comme le montrent les exemples précédents. Par ailleurs, les adresses e-mail comportent souvent un nom de domaine comme emplacement (après le @), de sorte que le nom réel de l'hôte MTA correspondant devra être récupéré par le biais du DNS. Il est donc recommandé de déléguer la tâche d'identification de l'hôte de destination approprié au MTA local ou au serveur SMTP distant, lorsqu'on utilise des boîtes aux lettres distantes.

Sendmail fournit la commande `sendmail` pour effectuer bon nombre d'opérations liées au courrier électronique, y compris l'aide à la composition de nouveaux messages. L'utilisateur devra également saisir à la main les en-têtes du message, mais d'une manière plus conviviale que s'il utilisait directement les commandes SMTP. Ainsi, une méthode plus adéquate pour que l'utilisatrice `emma@lab1.campus` envoie un message électronique à `dave@lab2.campus` serait la suivante :

```
$ sendmail dave@lab2.campus
From: emma@lab1.campus
To: dave@lab2.campus
Subject: Sender MTA Test

Hi Dave, this is a test for my MTA.
.
```

Là encore, le point seul sur une ligne termine le message. Celui-ci devrait être immédiatement envoyé au destinataire, à moins que le MTA local n'ait pas réussi à contacter le MTA distant. La commande `mailq` exécutée par `root` affichera tous les messages non délivrés. Si par exemple le MTA de `lab2.campus` n'a pas répondu, la commande `mailq` affichera le message non délivré et la cause de l'échec :

```
# mailq
      /var/spool/mqueue (1 request)
-----Q-ID----- --Size--  -----Q-Time-----  -----Sender/Recipient-----
xAIK3D9S000453      36 Mon Nov 18 20:03 <emma@lab1.campus>
                        (Deferred: Connection refused by lab2.campus.)
                        <dave@lab2.campus>
Total requests: 1
```

L'emplacement par défaut de la file d'attente de la boîte d'envoi est `/var/spool/mqueue/`, mais les différents MTA peuvent utiliser des emplacements différents dans le répertoire `/var/spool/`. Postfix, par exemple, va créer une arborescence de répertoires en-dessous de `/var/spool/postfix/` pour gérer la file d'attente. La commande `mailq` équivaut à `sendmail -bp`, et ces commandes devraient être présentes indépendamment du MTA installé sur le système. Pour assurer une compatibilité ascendante, la plupart des MTA fournissent ces commandes traditionnelles pour administrer le courrier.

Si l'hôte primaire de destination du courrier électronique—tel qu'il est fourni par un enregistrement DNS MX pour le domaine—est injoignable, le MTA essaiera de contacter les entrées avec une priorité plus faible (s'il y en a). Si aucune d'entre elles n'est joignable, le message reste dans la file d'attente de la boîte d'envoi locale pour être envoyé ultérieurement. S'il est configuré à cet effet, le MTA peut vérifier périodiquement la disponibilité des hôtes distants pour effectuer une nouvelle tentative de livraison. Si vous utilisez un MTA compatible avec Sendmail, une nouvelle tentative aura lieu immédiatement avec la commande `sendmail -q`.

Sendmail enregistre les messages entrants dans un fichier nommé d'après le propriétaire de la boîte de réception correspondante, par exemple `/var/spool/mail/dave`. D'autres MTA comme Postfix pourront conserver les messages entrants dans des emplacements comme `/var/mail/dave`, mais le contenu du fichier sera le même. Dans l'exemple, la commande `sendmail` a été utilisée dans l'hôte de l'expéditeur pour composer le message, de sorte que les entêtes bruts du message montrent que le courrier électronique a suivi des étapes supplémentaires avant d'atteindre sa destination finale :

```
$ cat /var/spool/mail/dave
From emma@lab1.campus  Mon Nov 18 20:07:39 2019
```

```

Return-Path: <emma@lab1.campus>
Received: from lab1.campus (lab1.campus [10.0.3.134])
    by lab2.campus (8.15.2/8.15.2) with ESMTPS id xAIK7c1C000432
    (version=TLSv1.3 cipher=TLS_AES_256_GCM_SHA384 bits=256 verify=NOT)
    for <dave@lab2.campus>; Mon, 18 Nov 2019 20:07:38 GMT
Received: from lab1.campus (localhost [127.0.0.1])
    by lab1.campus (8.15.2/8.15.2) with ESMTPS id xAIK3D9S000453
    (version=TLSv1.3 cipher=TLS_AES_256_GCM_SHA384 bits=256 verify=NOT)
    for <dave@lab2.campus>; Mon, 18 Nov 2019 20:03:13 GMT
Received: (from emma@localhost)
    by lab1.campus (8.15.2/8.15.2/Submit) id xAIK0doL000449
    for dave@lab2.campus; Mon, 18 Nov 2019 20:00:39 GMT
Date: Mon, 18 Nov 2019 20:00:39 GMT
Message-Id: <201911182000.xAIK0doL000449@lab1.campus>
From: emma@lab1.campus
To: dave@lab2.campus
Subject: Sender MTA Test

Hi Dave, this is a test for my MTA.

```

De bas en haut, les lignes qui commencent par `Received:` indiquent le chemin emprunté par le message. Le message a été soumis par l'utilisatrice `emma` avec la commande `sendmail dave@lab2.campus` émise sur `lab1.campus`, comme l'indique le premier en-tête `Received:`. Ensuite, toujours sur `lab1.campus`, le MTA utilise ESMTPS — une surcouche du SMTP qui ajoute des extensions de chiffrement — pour envoyer le message au MTA sur `lab2.campus`, comme l'indique le dernier en-tête `Received:` (en haut).

Le MTA conclut son travail une fois que le message est enregistré dans la boîte de réception de l'utilisateur. Il est courant de filtrer le courrier électronique, par exemple pour bloquer les spams ou appliquer des règles de filtrage définies par l'utilisateur. Ces tâches sont exécutées par des applications tierces conjointement avec le MTA. Le MTA pourra par exemple appeler l'utilitaire *SpamAssassin* pour marquer les messages suspects à l'aide de ses fonctions d'analyse de texte.

Même si cela est possible, il n'est pas pratique de lire directement le fichier de la boîte de réception. Il vaut mieux utiliser un client de messagerie (comme Thunderbird, Evolution ou KMail) qui va analyser le fichier et gérer les messages de manière appropriée. Ces programmes offrent également des fonctions supplémentaires comme des raccourcis pour les actions courantes, des sous-répertoires de la boîte de réception, etc.

La commande `mail` et les applications clientes

On peut très bien écrire un message électronique directement dans son format brut, mais dans la

pratique, il vaut mieux utiliser une application cliente — également appelée MUA (*Mail User Agent*) — pour accélérer le processus et éviter les erreurs. Le MUA gère le travail sous le capot, c'est-à-dire que le client de messagerie présente et organise les messages reçus et s'occupe de la communication avec le MTA une fois que l'utilisateur a rédigé son message.

Il existe plusieurs types distincts de clients de messagerie. Les applications de bureau comme *Mozilla Thunderbird* et *Evolution* de Gnome prennent en charge les comptes de messagerie locaux et distants. Même les interfaces webmail peuvent être considérées comme un type de MUA, puisqu'elles servent d'intermédiaire entre l'utilisateur et le MTA sous-jacent. Pourtant, les clients de messagerie ne se limitent pas aux interfaces graphiques : les clients en mode console sont largement utilisés pour accéder aux boîtes aux lettres non intégrées dans une interface graphique et pour automatiser les tâches liées à la messagerie dans des scripts shell.

À l'origine, la commande `mail` d'Unix ne servait qu'à partager des messages entre les utilisateurs d'un système local (la première commande `mail` remonte à la première version d'Unix, publiée en 1971). Au fur et à mesure que les échanges de courrier électronique en réseau se sont développés, d'autres programmes ont été créés pour gérer le nouveau système de livraison en remplacement de l'ancien programme `mail`.

De nos jours, la commande `mail` la plus utilisée est fournie par le paquet *mailx*, qui est compatible avec toutes les fonctionnalités modernes du courrier électronique. Dans la plupart des distributions Linux, la commande `mail` constitue simplement un lien symbolique vers la commande `mailx`. D'autres implémentations, comme le paquet *GNU Mailutils*, fournissent en gros les mêmes fonctionnalités que `mailx`. Il y a tout de même quelques petites différences entre elles, en particulier en ce qui concerne les options en ligne de commande.

Indépendamment de leur implémentation, toutes les variantes modernes de la commande `mail` utilisent deux modes : le *mode normal* et le *mode d'envoi*. Si une adresse e-mail est fournie comme argument à la commande `mail`, celle-ci passera en mode envoi, sinon elle restera en mode normal (lecture). En mode normal, les messages reçus sont répertoriés avec un index numérique pour chacun d'entre eux, de sorte que l'utilisateur peut s'y référer individuellement lorsqu'il tape des commandes dans l'invite interactive. La commande `print 1` permet d'afficher le contenu du message numéro 1, par exemple. Les commandes interactives peuvent être abrégées. Ainsi les commandes comme `print`, `delete` ou `reply` pourront être remplacées par `p`, `d` ou `r`, respectivement. La commande `mail` retiendra toujours le dernier message reçu ou le dernier message vu lorsqu'on omet le numéro d'index du message. La commande `quit` ou `q` permet de quitter le programme.

Le *mode d'envoi* est très pratique pour envoyer des messages électroniques automatisés. Il permet, par exemple, d'envoyer un e-mail à l'administrateur du système lorsqu'un script de maintenance programmée n'a pas réussi à accomplir sa tâche. En mode d'envoi, `mail` utilisera le contenu de

l'entrée standard comme corps du message :

```
$ mail -s "Maintenance fail" henry@lab3.campus <<<"The maintenance script failed at `date`"
```

Dans cet exemple, l'option `-s` a été ajoutée pour inclure un champ sujet dans le message. Le corps du message a été fourni par la redirection *Hereline* vers l'entrée standard, mais le contenu d'un fichier ou la sortie d'une commande peuvent également être transmis à l'entrée standard du programme. Si aucun contenu n'est fourni par une redirection vers l'entrée standard, le programme va attendre que l'utilisateur saisisse le corps du message. Dans ce cas, la touche `Ctrl + D` mettra fin au message. La commande `mail` se termine immédiatement une fois que le message a été ajouté à la file d'attente de la boîte d'envoi.

Personnalisation de la livraison

Par défaut, les comptes de messagerie d'un système Linux sont associés aux comptes système classiques. Par exemple, si l'utilisatrice Carol a l'identifiant `carol` sur l'hôte `lab2.campus`, son adresse de courrier électronique sera `carol@lab2.campus`. Cette association directe entre les comptes système et les boîtes aux lettres peut être étendue par des méthodes classiques fournies par la plupart des distributions Linux, en particulier le mécanisme de routage des e-mails fourni par le fichier `/etc/aliases`.

Un alias est un destinataire "virtuel" dont les messages reçus seront redirigés vers des boîtes aux lettres locales existantes ou vers d'autres types de destinations de stockage ou de traitement des messages. Les alias servent par exemple à déposer les messages envoyés à `postmaster@lab2.campus` dans la boîte aux lettres de Carol, une boîte aux lettres locale ordinaire dans le système `lab2.campus`. Pour ce faire, la ligne `postmaster: carol` devra être ajoutée au fichier `/etc/aliases` dans `lab2.campus`. Une fois le fichier `/etc/aliases` modifié, la commande `newaliases` devra être exécutée pour mettre à jour la base de données des alias du MTA et rendre effectifs les changements. Les commandes `sendmail -bi` ou `sendmail -I` peuvent également être utilisées pour mettre à jour la base de données des alias.

Les alias sont définis à raison d'un par ligne, dans le format `<alias>: <destination>`. En dehors des boîtes aux lettres locales habituelles indiquées par le nom d'utilisateur correspondant, d'autres types de destinations sont possibles :

- Un chemin complet (commençant par `/`) vers un fichier. Les messages envoyés à l'alias correspondant seront ajoutés à ce fichier.
- Une commande pour traiter le message. La `<destination>` doit commencer par un caractère pipe. Si la commande contient des caractères spéciaux (comme des espaces vides), elle devra être placée entre guillemets. Par exemple, l'alias `subscribe: |subscribe.sh` dans

lab2.campus va transférer tous les messages envoyés à subscribe@lab2.campus vers l'entrée standard de la commande subscribe.sh. Si sendmail fonctionne en mode *restricted shell*, les commandes autorisées —ou les liens vers celles-ci—devraient figurer dans /etc/smrsh/.

- Un fichier d'inclusion. Un alias peut avoir plusieurs destinations (séparées par des virgules), il peut donc être plus pratique de les conserver dans un fichier externe. Le mot-clé `:include:` doit indiquer le chemin du fichier, comme dans `:include:/var/local/destinations`
- Une adresse externe. Les alias peuvent également transférer les messages vers des adresses électroniques externes.
- Un autre alias.

Un utilisateur local non privilégié peut définir des alias pour son propre courrier électronique en éditant le fichier `.forward` dans son répertoire personnel. Comme les alias ne peuvent affecter que leur propre boîte aux lettres, seule la partie `<destination>` est nécessaire. Pour transférer tous les e-mails entrants vers une adresse externe, par exemple, l'utilisateur dave dans lab2.campus pourra créer le fichier `~/ .forward` suivant :

```
$ cat ~/.forward
emma@lab1.campus
```

Il va rediriger tous les messages envoyés à dave@lab2.campus vers emma@lab1.campus. Comme pour le fichier /etc/aliases, d'autres règles de redirection pourront être ajoutées à `.forward`, une par ligne. Cela dit, le fichier `.forward` doit être accessible en écriture par son propriétaire uniquement et il n'est pas nécessaire d'exécuter la commande `newaliases` après l'avoir modifié. Les fichiers dont le nom commence par un point n'apparaissent pas dans l'affichage normal des fichiers, ce qui peut empêcher l'utilisateur d'être au courant de l'existence d'alias actifs. Il est donc important de vérifier l'existence de ce fichier pour diagnostiquer les problèmes de distribution du courrier électronique.

Exercices guidés

1. Invoquée sans autres options ou arguments, la commande `mail henry@lab3.campus` passe en mode saisie pour que l'utilisateur puisse taper le message à `henry@lab3.campus`. Une fois le message terminé, quelle combinaison de touches fermera le mode saisie pour envoyer l'e-mail ?

2. Quelle commande l'utilisateur root peut-il exécuter pour afficher la liste des messages non distribués en provenance du système local ?

3. Comment un utilisateur non privilégié peut-il utiliser la méthode MTA classique pour transférer automatiquement tous ses e-mails entrants vers l'adresse `dave@lab2.campus` ?

Exercices d'approfondissement

1. En utilisant la commande `mail` fournie par `mailx`, quelle commande va envoyer un message à `emma@lab1.campus` avec le fichier `logs.tar.gz` comme pièce jointe et le résultat de la commande `uname -a` comme corps du message ?
2. L'administrateur d'un service de messagerie électronique doit surveiller les transferts d'e-mails sur le réseau, mais il ne veut pas encombrer sa boîte de réception avec des messages de test. Comment peut-il configurer un alias de messagerie au niveau du système pour rediriger tous les messages envoyés à l'utilisateur `test` vers le fichier `/dev/null` ?
3. Quelle commande à part `newaliases` pourrait être utilisée pour mettre à jour la base de données des alias après avoir ajouté un nouvel alias à `/etc/aliases` ?

Résumé

Cette leçon aborde le rôle et l'utilisation des agents de transfert de courrier (MTA) sur les systèmes Linux. Le MTA fournit une méthode classique de communication entre les comptes d'utilisateurs et peut être associé à d'autres applications pour fournir des fonctionnalités supplémentaires. La leçon porte sur les sujets suivants :

- Concepts relatifs aux technologies liées au courrier électronique, boîtes aux lettres et protocoles.
- Comment les MTA Linux échangent des messages sur le réseau.
- Les clients de messagerie en mode console et les MUA (*Mail User Agents*).
- Alias et transfert des e-mails locaux.

Voici les technologies, les commandes et les procédures qui ont été abordées :

- SMTP et protocoles connexes.
- MTA disponibles pour Linux: Sendmail, Postfix, qmail, Exim.
- Commandes MTA et MUA : `sendmail` et `mail`.
- Fichiers et commandes d'administration : `mailq`, `/etc/aliases`, `newaliases`, `~/.forward`.

Réponses aux exercices guidés

1. Invoquée sans autres options ou arguments, la commande `mail henry@lab3.campus` passe en mode saisie pour que l'utilisateur puisse taper le message à `henry@lab3.campus`. Une fois le message terminé, quelle combinaison de touches fermera le mode saisie pour envoyer l'e-mail ?

La combinaison de touches `Ctrl` + `D` entraînera la fermeture du programme et l'envoi de l'e-mail.

2. Quelle commande l'utilisateur root peut-il exécuter pour afficher la liste des messages non distribués en provenance du système local ?

La commande `mailq` ou `sendmail -bp`.

3. Comment un utilisateur non privilégié peut-il utiliser la méthode MTA classique pour transférer automatiquement tous ses e-mails entrants vers l'adresse `dave@lab2.campus` ?

L'utilisateur devra ajouter `dave@lab2.campus` à `~/ .forward`.

Réponses aux exercices d'approfondissement

1. En utilisant la commande `mail` fournie par `mailx`, quelle commande va envoyer un message à `emma@lab1.campus` avec le fichier `logs.tar.gz` comme pièce jointe et le résultat de la commande `uname -a` comme corps du message ?

```
uname -a | mail -a logs.tar.gz emma@lab1.campus
```

2. L'administrateur d'un service de messagerie électronique doit surveiller les transferts d'e-mails sur le réseau, mais il ne veut pas encombrer sa boîte de réception avec des messages de test. Comment peut-il configurer un alias de messagerie au niveau du système pour rediriger tous les messages envoyés à l'utilisateur `test` vers le fichier `/dev/null` ?

La ligne `test: /dev/null` dans `/etc/aliases` va rediriger tous les messages envoyés à la boîte aux lettres locale `test` vers le fichier `/dev/null`.

3. Quelle commande à part `newaliases` pourrait être utilisée pour mettre à jour la base de données des alias après avoir ajouté un nouvel alias à `/etc/aliases` ?

La commande `sendmail -bi` ou `sendmail -I`.



Linux
Professional
Institute

108.4 Gestion des imprimantes et de l'impression

Référence aux objectifs de LPI

LPIC-1 version 5.0, Exam 102, Objective 108.4

Valeur

2

Domaines de connaissance les plus importants

- Configuration élémentaire de CUPS (pour les imprimantes locales et distantes).
- Gestion des files d'attente des utilisateurs.
- Résolution des problèmes d'impression.
- Ajout et suppression des travaux des files d'impression configurées.

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

- Fichiers, outils et utilitaires de configuration de CUPS
- `/etc/cups/`
- Interface héritée de lpd (`lpr`, `lprm`, `lpq`)



108.4 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	108 Services Systèmes Essentiels
Objectif :	108.4 Gestion des imprimantes et de l'impression
Leçon :	1 sur 1

Introduction

Les promesses d'une "société sans papier" résultant de l'avènement de l'informatique se sont révélées fausses jusqu'à présent. Nombre d'organisations s'appuient encore sur des pages imprimées, ou "copies papier", d'informations. Dans ce contexte, on comprend l'importance pour l'utilisateur d'un ordinateur de savoir comment imprimer à partir d'un système, ainsi que pour l'administrateur de savoir comment maintenir la capacité d'un ordinateur à fonctionner avec des imprimantes.

Sous Linux ainsi que dans d'autres systèmes d'exploitation, la pile logicielle *Common Unix Printing System* (CUPS) permet d'imprimer et de gérer les imprimantes à partir d'un ordinateur. Voici un aperçu très simplifié de la manière dont un fichier est imprimé sous Linux avec CUPS :

1. Un utilisateur soumet un fichier pour être imprimé.
2. Le démon CUPS, `cupsd`, met la tâche d'impression en attente (*spool*). CUPS attribue un numéro à cette tâche ainsi que des informations sur la file d'attente dans laquelle elle se trouve et sur le nom du document à imprimer.

3. CUPS utilise les *filtres* installés sur le système pour générer un fichier formaté que l'imprimante pourra utiliser.
4. CUPS envoie alors le fichier reformaté à l'imprimante pour qu'elle l'imprime.

Nous allons voir ces étapes plus en détail, ainsi que l'installation et la gestion d'une imprimante sous Linux.

Le service CUPS

La plupart des postes de travail Linux disposeront déjà des paquets CUPS. Sur les installations Linux minimales, les paquets CUPS peuvent ne pas être installés, en fonction de la distribution. Une installation de base de CUPS pourra être effectuée sur un système Debian avec la commande suivante :

```
$ sudo apt install cups
```

Sur les systèmes Fedora, la procédure d'installation est tout aussi simple. Vous devrez démarrer le service CUPS manuellement après l'installation sur Fedora et d'autres distributions basées sur Red Hat :

```
$ sudo dnf install cups
...
$ sudo systemctl start cups.service
```

Une fois l'installation terminée, vous pouvez vérifier que le service CUPS fonctionne à l'aide de la commande `systemctl` :

```
$ systemctl status cups.service
• cups.service - CUPS Scheduler
  Loaded: loaded (/lib/systemd/system/cups.service; enabled; vendor preset: enabled)
  Active: active (running) since Thu 2020-06-25 14:35:47 EDT; 41min ago
    Docs: man:cupsd(8)
 Main PID: 3136 (cupsd)
   Tasks: 2 (limit: 1119)
  Memory: 3.2M
   CGroup: /system.slice/cups.service
           └─3136 /usr/sbin/cupsd -l
           └─3175 /usr/lib/cups/notifier/dbus dbus://
```

Comme la plupart des services sous Linux, CUPS s'appuie sur une série de fichiers de configuration pour son fonctionnement. Voici les principaux fichiers susceptibles d'intéresser l'administrateur du système :

/etc/cups/cupsd.conf

Ce fichier contient les paramètres de configuration du service CUPS lui-même. Si vous connaissez un tant soit peu le fichier de configuration du serveur web Apache, celui de CUPS vous semblera assez familier dans la mesure où il utilise la même syntaxe. Le fichier `cupsd.conf` contient des paramètres comme le contrôle d'accès aux différentes files d'attente d'impression du système, l'activation de l'interface web de CUPS ainsi que le niveau de journalisation utilisé par le service.

/etc/printcap

Il s'agit là du fichier utilisé par le protocole LPD (*Line Printer Daemon*) avant l'avènement de CUPS. CUPS crée toujours ce fichier sur les systèmes pour des raisons de rétro-compatibilité. Il s'agit souvent d'un lien symbolique vers `/run/cups/printcap`. Chaque ligne de ce fichier contient une imprimante à laquelle le système aura accès.

/etc/cups/printers.conf

Ce fichier contient chaque imprimante configurée pour être utilisée par le système CUPS. Chaque imprimante et sa file d'attente associée dans ce fichier est comprise dans une stance `<Printer></Printer>`. Ce fichier fournit les différentes imprimantes que l'on trouve dans `/etc/printcap`.

WARNING

Le fichier `/etc/cups/printers.conf` ne doit pas être modifié manuellement lorsque le service CUPS est en cours d'exécution.

/etc/cups/ppd/

Ce n'est pas un fichier de configuration mais un répertoire qui contient les fichiers PPD (*PostScript Printer Description*) pour les imprimantes qui les utilisent. Les fonctionnalités de chaque imprimante sont stockées dans un fichier PPD (avec l'extension `.ppd`). Ce sont des fichiers au format texte simple qui respectent un format spécifique.

Le service CUPS utilise la journalisation de la même manière que le service Apache 2. Les journaux sont stockés dans `/var/log/cups/` et contiennent les fichiers `access_log`, `page_log` et `error_log`. Le fichier `access_log` enregistre les accès à l'interface web de CUPS ainsi que les actions qui y sont effectuées, comme la gestion des imprimantes. Le fichier `page_log` garde la trace des tâches d'impression envoyées aux files d'attente d'impression gérées par CUPS. Le fichier `error_log` contient des messages sur les tâches d'impression qui ont échoué et d'autres erreurs enregistrées par l'interface web.

Nous allons maintenant découvrir les outils et les commandes utilisés pour gérer le service CUPS.

Utiliser l'interface Web

Comme nous l'avons dit plus haut, le fichier de configuration `/etc/cups/cupsd.conf` détermine si l'interface web du système CUPS est activée. L'option de configuration ressemble à ceci :

```
# Web interface setting...
WebInterface Yes
```

Si l'interface web est activée, CUPS peut être géré depuis un navigateur web à l'URL par défaut `http://localhost:631`. Dans la configuration par défaut, un utilisateur du système peut afficher les imprimantes et les files d'attente d'impression, mais toute forme de modification de la configuration nécessite un accès root avec une authentification auprès du service web. La stance de configuration dans le fichier `/etc/cups/cupsd.conf` pour restreindre l'accès aux capacités administratives ressemblera à ceci :

```
# All administration operations require an administrator to authenticate...
<Limit CUPS-Add-Modify-Printer CUPS-Delete-Printer CUPS-Add-Modify-Class CUPS-Delete-Class
CUPS-Set-Default>
  AuthType Default
  Require user @SYSTEM
  Order deny,allow
</Limit>
```

Voici un aperçu de ces options :

AuthType Default

affiche une invite d'authentification de base lorsqu'une action nécessite un accès root.

Require user @SYSTEM

un utilisateur avec des privilèges d'administration sera requis pour l'opération. Cela peut être changé en `@groupname` où les membres de `groupname` pourront administrer le service CUPS. Alternativement, des utilisateurs individuels pourraient être fournis sous forme de liste comme dans `Require user carol, tim`.

Order deny,allow

fonctionne comme l'option de configuration d'Apache 2 où l'action est refusée par défaut à moins qu'un utilisateur (ou un membre d'un groupe) ne soit authentifié.

L'interface web de CUPS peut être désactivée en arrêtant d'abord le service CUPS, puis en changeant l'option `WebInterface` de `Yes` à `No`, et enfin en redémarrant le service CUPS.

L'interface web de CUPS est organisée comme un site web ordinaire, avec des onglets de navigation pour les différentes sections du système CUPS. Voici les onglets de l'interface web :

Home

La page d'accueil indique la version actuellement installée de CUPS. Elle répartit CUPS en plusieurs sections, telles que :

CUPS for Users

Fournit une description de CUPS, des options en ligne de commande pour travailler avec les imprimantes et les files d'attente d'impression, et un lien vers le forum des utilisateurs de CUPS.

CUPS for Administrators

Fournit des liens dans l'interface pour installer et gérer les imprimantes ainsi que des liens vers les informations relatives à l'utilisation des imprimantes en réseau.

CUPS for Developers

Fournit des liens pour développer CUPS ainsi que pour créer des fichiers PPD pour les imprimantes.

Administration

La page d'administration est également organisée en sections :

Printers

Ici, un administrateur peut ajouter de nouvelles imprimantes au système, trouver les imprimantes connectées au système et gérer les imprimantes déjà installées.

Classes

Les classes sont un mécanisme qui permet d'ajouter des imprimantes à des groupes avec des politiques spécifiques. Par exemple, une classe peut contenir un groupe d'imprimantes appartenant à un étage du bâtiment et sur lesquelles seuls les utilisateurs d'un service donné pourront imprimer. Une autre classe peut limiter le nombre de pages qu'un utilisateur a le droit d'imprimer. Les classes ne sont pas créées par défaut sur une installation CUPS et doivent être définies par un administrateur. C'est dans cette section de l'interface web de CUPS que de nouvelles classes peuvent être créées et gérées.

Jobs

C'est ici qu'un administrateur peut voir toutes les tâches d'impression qui sont actuellement

dans la file d'attente pour toutes les imprimantes gérées par cette installation CUPS.

Server

C'est ici qu'un administrateur peut apporter des modifications au fichier `/etc/cups/cupsd.conf`. D'autres options de configuration sont également disponibles via des cases à cocher, telles que l'autorisation de partager sur un réseau les imprimantes connectées à cette installation de CUPS, l'authentification avancée et l'autorisation pour l'administration à distance des imprimantes.

Classes

Si les classes d'imprimantes sont configurées sur le système, elles seront répertoriées sur cette page. Chaque classe d'imprimantes dispose d'options permettant de gérer toutes les imprimantes de la classe en même temps, ainsi que de visualiser toutes les tâches en attente pour les imprimantes de cette classe.

Help

Cet onglet fournit des liens vers toute la documentation disponible pour CUPS qui est installé sur le système.

Jobs

L'onglet Jobs permet de rechercher des tâches d'impression individuelles et d'afficher la liste de toutes les tâches d'impression en cours gérées par le serveur.

Printers

L'onglet Printers (Imprimantes) répertorie toutes les imprimantes actuellement gérées par le système et donne un aperçu rapide de l'état de chacune d'entre elles. Chaque imprimante affichée est cliquable et l'administrateur sera redirigé vers une page où l'imprimante individuelle pourra être gérée de manière plus détaillée. Les informations concernant les imprimantes de cet onglet proviennent du fichier `/etc/cups/printers.conf`.

Installer une imprimante

L'ajout d'une imprimante au système est une procédure triviale dans l'interface web CUPS :

1. Cliquez sur l'onglet **Administration** puis sur le bouton **Add Printer**.
2. La page suivante propose différentes options en fonction de la manière dont votre imprimante est connectée au système. S'il s'agit d'une imprimante locale, sélectionnez l'option la plus pertinente, par exemple le port auquel l'imprimante est connectée ou le logiciel d'impression tiers éventuellement installé. CUPS essaiera également de détecter les imprimantes connectées au réseau et les affichera ici. Vous pouvez également choisir une option de connexion directe à

une imprimante réseau en fonction des protocoles d'impression réseau pris en charge par l'imprimante. Sélectionnez l'option appropriée et cliquez sur le bouton **Continuer**.

3. La page suivante vous permet de fournir un nom, une description et un emplacement (comme "bureau" ou "accueil" etc.) pour l'imprimante. Si vous souhaitez partager cette imprimante sur le réseau, vous pouvez également cocher la case correspondant à cette option sur cette page. Une fois les paramètres définis, cliquez sur le bouton **Continuer**.
4. La page suivante permet de sélectionner la marque et le modèle de l'imprimante. Cela permet à CUPS de rechercher dans sa base de données locale les pilotes et les fichiers PPD les plus adaptés à l'imprimante. Si vous disposez d'un fichier PPD fourni par le fabricant de l'imprimante, recherchez son emplacement et sélectionnez-le pour l'utiliser ici. Une fois que c'est fait, cliquez sur le bouton **Add Printer**.
5. La dernière page permet de définir les options par défaut, notamment la taille de papier que l'imprimante va utiliser et la résolution des caractères sur la page. Cliquez sur le bouton **Set Default Options** et votre imprimante est désormais installée sur votre système.

NOTE

La plupart des postes de travail Linux disposent de plusieurs outils pour installer une imprimante. Les environnements de bureau GNOME et KDE ont leurs propres applications intégrées pour installer et gérer les imprimantes. Certaines distributions fournissent également des applications distinctes pour gérer les imprimantes. Cela dit, lorsqu'il s'agit d'une installation de serveur où de nombreux utilisateurs vont imprimer, l'interface web CUPS peut s'avérer l'outil le plus adapté à cette tâche.

La file d'attente d'une imprimante peut également être installée à l'aide des commandes historiques LPD/LPR. Voici un exemple qui utilise la commande `lpadmin` :

```
$ sudo lpadmin -p ENVY-4510 -L "office" -v socket://192.168.150.25 -m everywhere
```

Voici le détail des options utilisées pour cette commande :

- Puisque l'ajout d'une imprimante au système nécessite un utilisateur avec des privilèges d'administration, nous faisons précéder la commande `lpadmin` par `sudo`.
- L'option `-p` est la destination de vos travaux d'impression. Il s'agit essentiellement d'un nom convivial permettant à l'utilisateur de savoir où vont atterrir les travaux d'impression. Typiquement, vous pouvez fournir le nom de l'imprimante.
- L'option `-L` est l'emplacement de l'imprimante. Cette option est facultative mais utile si vous devez gérer un certain nombre d'imprimantes à différents endroits.
- L'option `-v` correspond à l'URI du périphérique d'impression. L'URI du périphérique est ce qu'il

faut à la file d'attente d'impression de CUPS pour envoyer des travaux d'impression rendus vers une imprimante spécifique. Dans notre exemple, nous utilisons un emplacement réseau en utilisant l'adresse IP fournie.

- La dernière option, `-m`, est définie à “everywhere”. Elle indique le modèle de l'imprimante pour que CUPS puisse déterminer le fichier PPD à utiliser. Dans les versions modernes de CUPS, il est préférable d'utiliser “everywhere” afin que CUPS puisse vérifier l'URI du périphérique (défini avec l'option `-v` précédente) pour déterminer automatiquement le bon fichier PPD à utiliser pour l'imprimante. Dans les situations modernes, CUPS utilisera simplement IPP comme expliqué ci-dessous.

Comme nous l'avons dit plus haut, il vaut mieux laisser CUPS choisir automatiquement le fichier PPD à utiliser pour une file d'attente d'impression particulière. Cependant, la commande `lpinfo` peut être utilisée pour interroger les fichiers PPD installés localement pour voir ce qui est disponible. Fournissez simplement l'option `--make-and-model` pour l'imprimante que vous souhaitez installer avec l'option `-m` :

```
$ lpinfo --make-and-model "HP Envy 4510" -m
hplip:0/ppd/hplip/HP/hp-envy_4510_series-hpijs.ppd HP Envy 4510 Series hpijs, 3.17.10
hplip:1/ppd/hplip/HP/hp-envy_4510_series-hpijs.ppd HP Envy 4510 Series hpijs, 3.17.10
hplip:2/ppd/hplip/HP/hp-envy_4510_series-hpijs.ppd HP Envy 4510 Series hpijs, 3.17.10
drv:///hpcups.crv/hp-envy_4510_series.ppd HP Envy 4510 Series, hpcups 3.17.10
everywhere IPP Everywhere
```

Notez que la commande `lpinfo` est obsolète. Elle est présentée ici comme un exemple pour afficher les fichiers du pilote d'impression qu'une imprimante pourrait utiliser.

WARNING

Les versions ultérieures de CUPS déprécient les pilotes et se concentrent sur l'utilisation du protocole IPP (*Internet Printing Protocol*) et des formats de fichiers standard. Le résultat de la commande précédente illustre cela avec la capacité d'impression `everywhere IPP Everywhere`. Le protocole IPP peut effectuer les mêmes tâches qu'un pilote d'impression. Tout comme l'interface web de CUPS, IPP utilise le port réseau 631 avec le protocole TCP.

Une imprimante par défaut peut être définie en utilisant la commande `lpoptions`. Ainsi, si la majorité (ou la totalité) des tâches d'impression sont envoyées vers une imprimante particulière, celle spécifiée par la commande `lpoptions` sera l'imprimante par défaut. Il suffit de spécifier cette imprimante avec l'option `-d` :

```
$ lpoptions -d ENVY-4510
```

Gérer les imprimantes

Une fois qu'une imprimante a été installée, un administrateur peut utiliser l'interface web pour gérer les options disponibles pour l'imprimante. Une approche plus directe de la gestion d'une imprimante consiste à utiliser la commande `lpadmin`.

Une option consiste à permettre le partage en réseau d'une imprimante. Ceci peut être réalisé avec l'option `printer-is-shared`, et en spécifiant l'imprimante avec l'option `-p` :

```
$ sudo lpadmin -p FRONT-DESK -o printer-is-shared=true
```

Un administrateur peut également configurer une file d'attente d'impression pour qu'elle n'accepte que les tâches d'impression d'utilisateurs spécifiques, en séparant chaque utilisateur par une virgule :

```
$ sudo lpadmin -p FRONT-DESK -u allow:carol,frank,grace
```

Inversement, certains utilisateurs peuvent se voir refuser l'accès à une file d'attente d'impression spécifique :

```
$ sudo lpadmin -p FRONT-DESK -u deny:dave
```

Les groupes d'utilisateurs peuvent également être utilisés pour autoriser ou refuser l'accès à la file d'attente d'une imprimante, à condition que le nom du groupe soit précédé du caractère arobase (@) :

```
$ sudo lpadmin -p FRONT-DESK -u deny:@sales,@marketing
```

Une file d'attente d'impression peut également avoir une politique d'erreur si elle rencontre des problèmes lors de l'impression d'une tâche. Avec l'utilisation de politiques, une tâche d'impression peut être abandonnée (`abort-job`) ou une autre tentative d'impression peut avoir lieu plus tard (`retry-job`). D'autres politiques incluent la possibilité d'arrêter l'imprimante immédiatement en cas d'erreur (`stop-printer`) ainsi que la possibilité de relancer la tâche immédiatement après la détection d'un échec (`retry-current-job`). Voici un exemple où la politique d'impression est définie pour interrompre la tâche d'impression si une erreur se produit sur l'imprimante `FRONT-DESK` :

```
$ sudo lpadmin -p FRONT-DESK -o printer-error-policy=abort-job
```

Assurez-vous de consulter les pages de manuel de la commande `lpadmin` située à `lpadmin(8)` pour plus de détails sur l'utilisation de cette commande.

Soumettre des tâches d'impression

La plupart des applications de bureau vous permettront d'envoyer des tâches d'impression à partir d'un élément de menu ou en utilisant le raccourci clavier `Ctrl + p`. Si vous vous retrouvez dans un système Linux sans environnement de bureau, vous pouvez toujours envoyer des fichiers vers une imprimante grâce aux commandes historiques LPD/LPR.

La commande `lpr` ("line printer remote") est utilisée pour envoyer une tâche d'impression vers la file d'attente d'une imprimante. Dans la forme la plus basique de la commande, un nom de fichier en argument de la commande `lpr` est tout ce qu'il faut :

```
$ lpr report.txt
```

La commande ci-dessus va envoyer le fichier `report.txt` vers la file d'attente d'impression par défaut du système (définie par le fichier `/etc/cups/printers.conf`).

Si plusieurs imprimantes sont disponibles dans une installation CUPS, la commande `lpstat` pourra être utilisée pour afficher la liste des imprimantes disponibles en utilisant l'option `-p`. Quant à l'option `-d`, elle indiquera l'imprimante par défaut :

```
$ lpstat -p -d
printer FRONT-DESK is idle.  enabled since Mon 03 Aug 2020 10:33:07 AM EDT
printer PostScript_oc0303387803 disabled since Sat 07 Mar 2020 08:33:11 PM EST -
    reason unknown
printer ENVY-4510 is idle.  enabled since Fri 31 Jul 2020 10:08:31 AM EDT
system default destination: ENVY-4510
```

Dans notre exemple, le fichier `report.txt` sera envoyé à l'imprimante `ENVY-4510` définie par défaut. Si le fichier doit être imprimé sur une autre imprimante, vous pouvez la spécifier avec l'option `-P` :

```
$ lpr -P FRONT-DESK report.txt
```

Lorsqu'une tâche d'impression est soumise à CUPS, le démon déterminera quel est le système d'arrière-plan le mieux adapté pour gérer la tâche. CUPS peut utiliser divers pilotes d'impression, filtres, moniteurs de ports matériels et autres logiciels pour reproduire correctement le document. Il peut arriver qu'un utilisateur qui imprime un document doive apporter des modifications à la manière dont le document devra être imprimé. La plupart des applications graphiques facilitent cette tâche. Il existe également des options en ligne de commande pour modifier le mode d'impression d'un document. Lorsqu'une tâche d'impression est soumise via la ligne de commande, l'option `-o` (pour "options") peut être utilisée avec des arguments spécifiques pour ajuster la mise en page du document en vue de l'impression. Voici une petite liste des options les plus couramment utilisées :

landscape

Le document est imprimé avec une rotation de la page de 90 degrés dans le sens des aiguilles d'une montre. L'option `orientation-requested=4` permet d'obtenir le même résultat.

two-sided-long-edge

L'imprimante va imprimer le document en mode portrait sur les deux côtés du papier, à condition que l'imprimante supporte cette fonctionnalité.

two-sided-short-edge

L'imprimante va imprimer le document en mode paysage sur les deux côtés du papier, à condition que l'imprimante supporte cette fonctionnalité.

media

L'imprimante utilisera le format de support spécifié pour imprimer le document. Les formats disponibles pour une tâche d'impression dépendent de l'imprimante, mais voici une liste des formats les plus courants :

Option de taille	Format du support
A4	ISO A4
Letter	US Letter
Legal	US Legal
DL	Enveloppe ISO DL
COM10	Enveloppe US #10

collate

Assemble le document imprimé. Cette option est utile si vous avez un document de plusieurs pages qui sera imprimé plus d'une fois, étant donné que toutes les pages de chaque document

seront alors imprimées dans l'ordre. Réglez cette option à `true` pour l'activer ou à `false` pour la désactiver.

page-ranges

Cette option peut être utilisée pour sélectionner une seule page à imprimer, ou un ensemble spécifique de pages à imprimer à partir d'un document. Un exemple pourrait ressembler à ceci : `-o page-ranges=5-7,9,15`. Cela va imprimer les pages 5, 6 et 7, puis les pages 9 et 15.

fit-to-page

Imprime le document de manière à ce que le fichier soit mis à l'échelle pour s'adapter au papier. Si aucune information sur la taille de la page n'est fournie par le fichier à imprimer, il est possible que la tâche d'impression soit mise à l'échelle de manière incorrecte et que des parties du document soient mises à l'échelle hors de la page ou que le document soit mis à l'échelle de manière trop petite.

outputorder

Imprime le document dans l'ordre `inverse` ou `normal` pour commencer l'impression à la première page. Si une imprimante imprime ses pages face vers le bas, l'ordre par défaut est `-o outputorder=normal` alors que les imprimantes qui impriment leurs pages face vers le haut imprimeront avec `-o outputorder=reverse`.

En prenant un échantillon des options ci-dessus, l'exemple de commande suivant peut être construit :

```
$ lpr -P ACCOUNTING-LASERJET -o landscape -o media=A4 -o two-sided-short-edge finance-report.pdf
```

Il est possible d'imprimer plus d'une copie d'un document en utilisant l'option `nombre` dans le format suivant : `-#N` où `N` est égal au nombre de copies à imprimer. Voici un exemple avec l'option de regroupement où sept copies d'un rapport doivent être imprimées sur l'imprimante par défaut :

```
$ lpr -#7 -o collate=true status-report.pdf
```

En dehors de la commande `lpr`, la commande `lp` peut également être utilisée. La plupart des options utilisées avec la commande `lpr` peuvent également être utilisées avec la commande `lp`, mais il y a quelques différences. Consultez la page de manuel de `lp(1)` pour plus de détails. Voici comment nous pouvons exécuter la commande `lpr` de l'exemple précédent en utilisant la syntaxe de la commande `lp` tout en spécifiant l'imprimante de destination avec l'option `-d` :

```
$ lp -d ACCOUNTING-LASERJET -n 7 -o collate=true status-report.pdf
```

Gérer les tâches d'impression

Comme nous l'avons dit plus haut, chaque tâche d'impression soumise à la file d'attente reçoit un numéro d'identification de CUPS. Un utilisateur peut afficher les tâches d'impression qu'il a soumises avec la commande `lpq`. En passant l'option `-a`, les files d'attente de toutes les imprimantes gérées par CUPS seront affichées :

```
$ lpq -a
```

Rank	Owner	Job	File(s)	Total Size
1st	carol	20	finance-report.pdf	5072 bytes

La même commande `lpstat` utilisée précédemment possède également une option pour afficher les files d'attente des imprimantes. L'option `-o` seule affichera toutes les files d'attente, ou alors une file d'attente peut être spécifiée par son nom :

```
$ lpstat -o
```

ACCOUNTING-LASERJET-4	carol	19456	Wed 05 Aug 2020 04:29:44 PM EDT
-----------------------	-------	-------	---------------------------------

L'identifiant de la tâche d'impression est précédé du nom de la file d'attente vers laquelle la tâche a été envoyée, puis du nom de l'utilisateur qui a soumis la tâche, de la taille du fichier et de l'heure à laquelle il a été envoyé.

Si une tâche d'impression est bloquée sur une imprimante ou si un utilisateur souhaite annuler sa tâche, utilisez la commande `lprm` avec l'identifiant de la tâche obtenu par la commande `lpq` :

```
$ lprm 20
```

Toutes les tâches d'une file d'attente d'impression peuvent être supprimées à la fois en fournissant simplement un tiret `-` :

```
$ lprm -
```

Alternativement, la commande CUPS `cancel` peut également être utilisée par un utilisateur pour arrêter sa tâche d'impression en cours :

```
$ cancel
```

Une tâche d'impression spécifique peut être annulée à l'aide de son numéro d'identification, précédé du nom de l'imprimante :

```
$ cancel ACCOUNTING-LASERJET-20
```

Une tâche d'impression peut également être déplacée d'une file d'attente vers une autre. Cela peut être utile lorsqu'une imprimante ne répond plus ou que le document à imprimer nécessite des fonctionnalités disponibles sur une autre imprimante. Notez que cette procédure nécessite généralement un utilisateur disposant de privilèges élevés. En utilisant la même tâche d'impression que dans l'exemple précédent, nous pourrions la déplacer vers la file d'attente de l'imprimante FRONT-DESK :

```
$ sudo lpmove ACCOUNTING-LASERJET-20 FRONT-DESK
```

Supprimer des imprimantes

Pour supprimer une imprimante, on peut commencer par répertorier toutes les imprimantes gérées par le service CUPS. Ceci peut être fait avec la commande `lpstat` :

```
$ lpstat -v
device for FRONT-DESK: socket://192.168.150.24
device for ENVY-4510: socket://192.168.150.25
device for PostScript_oc0303387803: ///dev/null
```

L'option `-v` ne répertorie pas seulement les imprimantes mais aussi où (et comment) elles sont connectées. C'est une bonne pratique de commencer par rejeter toute nouvelle tâche destinée à l'imprimante et de fournir une raison pour laquelle l'imprimante n'acceptera pas de nouvelles tâches. Voici ce que vous pouvez faire :

```
$ sudo cupsreject -r "Printer to be removed" FRONT-DESK
```

Notez l'utilisation de `sudo` étant donné que cette tâche nécessite un utilisateur avec des privilèges élevés.

Pour supprimer une imprimante, nous utilisons la commande `lpadmin` avec l'option `-x` :

```
$ sudo lpadmin -x FRONT-DESK
```

Exercices guidés

1. Une nouvelle imprimante vient d'être installée sur un poste de travail local avec le nom `office-mgr`. Quelle commande pourrait être utilisée pour définir cette imprimante comme imprimante par défaut pour ce poste de travail ?
2. Quelle commande et quelle option permettent de déterminer les imprimantes disponibles pour l'impression à partir d'un poste de travail ?
3. En utilisant la commande `cancel`, comment supprimer une tâche d'impression avec l'identifiant 15 qui est bloquée dans la file d'attente de l'imprimante nommée `office-mgr` ?
4. Vous avez une tâche d'impression destinée à une imprimante qui n'a pas assez de papier pour imprimer le fichier complet. Quelle commande pourriez-vous utiliser pour déplacer la tâche d'impression avec l'identifiant 2 en file d'attente sur l'imprimante FRONT-DESK vers la file d'attente de l'imprimante ACCOUNTING-LASERJET ?

Exercices d'approfondissement

En utilisant le gestionnaire de paquets de votre distribution, installez les paquets `cups` et `printer-driver-cups-pdf`. Notez que si vous utilisez une distribution de la famille Red Hat (comme Fedora), le pilote PDF de CUPS s'appelle `cups-pdf`. Nous utiliserons ces paquets pour nous entraîner à gérer une imprimante CUPS sans installer physiquement une véritable imprimante.

1. Vérifiez que le démon CUPS fonctionne et que l'imprimante PDF est activée et définie par défaut.

2. Lancez une commande qui va imprimer le fichier `/etc/services`. Vous devriez maintenant avoir un répertoire nommé `PDF` dans votre répertoire personnel.

3. Utilisez une commande qui désactive seulement l'imprimante, puis exécutez une autre commande qui affiche toutes les informations d'état pour vérifier que l'imprimante PDF est désactivée.

4. Essayez maintenant d'imprimer une copie du fichier `/etc/fstab` sur l'imprimante PDF. Que se passe-t-il ?

5. Annulez la tâche d'impression et supprimez l'imprimante PDF.

Résumé

Le démon CUPS est une plate-forme largement utilisée pour imprimer sur des imprimantes locales et distantes. Même s'il remplace l'ancien protocole LPD, il reste compatible avec ses outils.

Voici les fichiers et les commandes abordés dans cette leçon :

/etc/cups/cupsd.conf

Le fichier de configuration principal pour le service CUPS lui-même. Ce fichier contrôle également l'accès à l'interface web de CUPS.

/etc/printcap

Un fichier historique utilisé par LPD qui contient une ligne pour chaque imprimante connectée au système.

/etc/cups/printers.conf

Le fichier de configuration utilisé par CUPS pour les informations sur les imprimantes.

L'interface web de CUPS qui se trouve à l'adresse `http://localhost:631` dans une installation par défaut. Rappelez-vous que le port réseau par défaut pour l'interface web est 631/TCP.

Les commandes historiques LPD/LPR ont également été abordées :

lpadmin

Utilisé pour installer et supprimer des imprimantes et des classes d'imprimantes.

lpoptions

Permet d'imprimer les options de l'imprimante et de modifier ses paramètres.

lpstat

Utilisé pour afficher les informations d'état des imprimantes connectées à CUPS.

lpr

Utilisé pour soumettre des tâches d'impression à la file d'attente d'une imprimante.

lp

Utilisé pour soumettre des tâches d'impression à la file d'attente d'une imprimante.

lpq

Cette commande affiche la liste des tâches d'impression dans la file d'attente.

lprm

Permet d'annuler des tâches d'impression en fonction de leur identifiant. L'identifiant de la tâche peut être obtenu avec la commande `lpq`.

cancel

Une alternative à la commande `lprm` pour annuler les tâches d'impression par leur identifiant.

Consultez les pages de manuel suivantes pour les différents programmes et outils de CUPS : `lpadmin(8)`, `lpoptions(1)`, `lpr(1)`, `lpq(1)`, `lprm(1)`, `cancel(1)`, `lpstat(1)`, `cupsenable(8)` et `cupsaccept(8)`. Vous pouvez également consulter la documentation en ligne à l'adresse <http://localhost:631/help>.

Réponses aux exercices guidés

1. Une nouvelle imprimante vient d'être installée sur un poste de travail local avec le nom `office-mgr`. Quelle commande pourrait être utilisée pour définir cette imprimante comme imprimante par défaut pour ce poste de travail ?

```
$ lpoptions -d office-mgr
```

2. Quelle commande et quelle option permettent de déterminer les imprimantes disponibles pour l'impression à partir d'un poste de travail ?

```
$ lpstat -p
```

L'option `-p` affiche la liste de toutes les imprimantes disponibles et indique si elles sont disponibles pour l'impression.

3. En utilisant la commande `cancel`, comment supprimer une tâche d'impression avec l'identifiant 15 qui est bloquée dans la file d'attente de l'imprimante nommée `office-mgr` ?

```
$ cancel office-mgr-15
```

4. Vous avez une tâche d'impression destinée à une imprimante qui n'a pas assez de papier pour imprimer le fichier complet. Quelle commande pourriez-vous utiliser pour déplacer la tâche d'impression avec l'identifiant 2 en file d'attente sur l'imprimante `FRONT-DESK` vers la file d'attente de l'imprimante `ACCOUNTING-LASERJET` ?

```
$ sudo lpmove FRONT-DESK-2 ACCOUNTING-LASERJET
```

Réponses aux exercices d'approfondissement

En utilisant le gestionnaire de paquets de votre distribution, installez les paquets `cups` et `printer-driver-cups-pdf`. Notez que si vous utilisez une distribution de la famille Red Hat (comme Fedora), le pilote PDF de CUPS s'appelle `cups-pdf`. Nous utiliserons ces paquets pour nous entraîner à gérer une imprimante CUPS sans installer physiquement une véritable imprimante.

1. Vérifiez que le démon CUPS fonctionne et que l'imprimante PDF est activée et définie par défaut.

Une manière de vérifier la disponibilité et l'état de l'imprimante PDF consisterait à exécuter la commande suivante :

```
$ lpstat -p -d

printer PDF is idle. enabled since Thu 25 Jun 2020 02:36:07 PM EDTi

system default destination: PDF
```

2. Lancez une commande qui va imprimer le fichier `/etc/services`. Vous devriez maintenant avoir un répertoire nommé PDF dans votre répertoire personnel.

```
$ lp -d PDF /etc/services
```

devrait fonctionner. Vous obtiendrez une version PDF de ce fichier dans le répertoire PDF.

3. Utilisez une commande qui désactive seulement l'imprimante, puis exécutez une autre commande qui affiche toutes les informations d'état pour vérifier que l'imprimante PDF est désactivée.

```
$ sudo cupsdisable PDF
```

désactive l'imprimante.

Lancez ensuite la commande `lpstat -t` pour obtenir un rapport complet sur l'état de l'imprimante. Cela devrait ressembler au résultat ci-dessous :

```
$ scheduler is running
```

```
system default destination: PDFi  
  
device for PDF: cups-pdf:/  
  
PDF accepting requests since Wed 05 Aug 2020 04:19:15 PM EDTi  
  
printer PDF disabled since Wed 05 Aug 2020 04:19:15 PM EDT -  
  
Paused
```

4. Essayez maintenant d'imprimer une copie du fichier `/etc/fstab` sur l'imprimante PDF. Que se passe-t-il ?

Après avoir essayé la commande `lp -d PDF /etc/fstab`, vous devriez obtenir un résultat qui montre l'information sur l'identifiant de la tâche. En revanche, si vous vérifiez le dossier PDF dans votre répertoire personnel, le nouveau fichier ne s'y trouve pas. Vous pouvez alors vérifier la file d'attente d'impression avec la commande `lpstat -o`, et vous trouverez votre tâche dans cette file d'attente.

5. Annulez la tâche d'impression et supprimez l'imprimante PDF.

En utilisant le résultat de la commande `lp` ci-dessus, utilisez la commande `cancel` pour supprimer la tâche. Par exemple :

```
$ cancel PDF-4
```

Lancez ensuite la commande `lpstat -o` pour vérifier que la tâche a bien été supprimée.

Supprimez l'imprimante PDF avec la commande suivante : `sudo lpadmin -x PDF`. Vérifiez ensuite que l'imprimante a bien été supprimée : `lpstat -a`.



Thème 109: Notions élémentaires sur les réseaux



109.1 Notions élémentaires sur les protocoles Internet

Référence aux objectifs de LPI

LPIC-1 version 5.0, Exam 102, Objective 109.1

Valeur

4

Domaines de connaissance les plus importants

- Compréhension des masques réseau et de la notation CIDR.
- Connaissance des différences entre les adresses IP privées et publiques.
- Connaissance des ports TCP et UDP les plus courants (20, 21, 22, 23, 25, 53, 80, 110, 123, 139, 143, 161, 162, 389, 443, 465, 514, 636, 993, 995).
- Connaissance des caractéristiques principales et des différences entre UDP, TCP et ICMP.
- Connaissance des différences principales entre IPv4 et IPv6.
- Connaissance des caractéristiques de base d'IPv6.

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

- `/etc/services`
- IPv4, IPv6
- Sous-réseaux
- TCP, UDP, ICMP



109.1 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	109 Principes de base des réseaux
Objectif :	109.1 Fondamentaux des protocoles internet
Leçon :	1 sur 2

Introduction

Le TCP/IP (*Transmission Control Protocol/Internet Protocol*), pour protocole de contrôle de transmissions/protocole internet, est une pile de protocoles utilisés pour établir la communication entre les ordinateurs. En dépit de son nom, la pile comprend plusieurs protocoles comme IP, TCP, UDP, ICMP, DNS, SMTP, ARP etc.

Protocole internet (IP *Internet Protocol*)

L'IP est le protocole en charge de l'adressage logique d'un hôte, et qui permet d'envoyer un paquet d'un hôte à l'autre. Pour ce faire, chaque appareil du réseau se voit attribuer une adresse IP unique, et il est tout à fait possible d'attribuer plus d'une adresse à un même appareil.

Dans la version 4 du protocole IP, généralement appelée IPv4, l'adresse est formée d'un ensemble de 32 bits séparés en 4 groupes de 8 bits, représentés sous forme décimale, appelée "notation décimale pointée" (*dotted quad*). En voici un exemple :

Format binaire (4 groupes de 8 bits)

```
11000000.10101000.00001010.00010100
```

Format décimal

192.168.10.20

En IPv4, les valeurs de chaque octet peuvent aller de 0 à 255, ce qui équivaut à 11111111 en format binaire.

Classes d'adresses

En théorie, les adresses IP sont regroupées en classes, elles-mêmes déterminées par la plage du premier octet, comme le montre le tableau ci-dessous :

Classe	Premier octet	Plage	Exemple
A	1-126	1.0.0.0 – 126.255.255.255	10.25.13.10
B	128-191	128.0.0.0 – 191.255.255.255	141.150.200.1
C	192-223	192.0.0.0 – 223.255.255.255	200.178.12.242

IP publiques et privées

Comme nous l'avons déjà dit, pour qu'il y ait possibilité de communication, chaque appareil sur le réseau doit être associé à une adresse IP unique au moins. En revanche, si chaque appareil connecté à Internet dans le monde disposait d'une adresse IP unique, il n'y aurait pas assez d'adresses IP (v4) pour tout le monde. C'est pour cette raison que les adresses IP *privées* ont été définies.

Les adresses IP privées sont des plages d'adresses IP réservées aux réseaux internes (privés) des entreprises, des institutions, des particuliers, etc. Au sein d'un même réseau, l'utilisation d'une adresse IP reste unique. En contrepartie, la même adresse IP privée pourra être utilisée au sein de n'importe quel réseau privé.

Ainsi, sur Internet, le trafic de données utilise des adresses IP publiques, qui sont reconnaissables et routées sur Internet, tandis qu'au sein des réseaux privés, on utilise des plages d'adresses IP réservées. Le routeur est chargé de convertir le trafic du réseau privé vers le réseau public et vice versa.

Les plages IP privées, séparées par classes, sont indiquées dans le tableau ci-dessous :

Classe	Premier octet	Plage	IP privées
A	1-126	1.0.0.0 – 126.255.255.255	10.0.0.0 – 10.255.255.255
B	128-191	128.0.0.0 – 191.255.255.255	172.16.0.0 – 172.31.255.255
C	192-223	192.0.0.0 – 223.255.255.255	192.168.0.0 – 192.168.255.255

Conversion du format décimal en binaire

Pour les sujets de cette leçon, il convient de savoir convertir les adresses IP en format binaire et en format décimal.

La conversion du format décimal au format binaire s'effectue par des divisions consécutives par 2. A titre d'exemple, nous allons convertir la valeur 105 en suivant les étapes ci-dessous :

1. En divisant la valeur 105 par 2, on obtient :

```
105/2
Quotient = 52
Reste = 1
```

2. Divisez le quotient successivement par 2, jusqu'à ce que le quotient soit égal à 1 :

```
52/2
Reste = 0
Quotient = 26
```

```
26/2
Reste = 0
Quotient = 13
```

```
13/2
Reste = 1
Quotient = 6
```

```
6/2
```


Reste = 0
Quotient = 3

3/2
Reste = 1
Quotient = 1

3. Regroupez le dernier quotient avec le reste de toutes les divisions :

1101001

4. Remplissez de 0 à gauche jusqu'à ce que 8 bits soient complets :

01101001

5. Au final, on obtient que la valeur 105 en décimal est égale à 01101001 en binaire.

Conversion du format binaire en décimal

Dans cet exemple, nous allons utiliser la valeur binaire 10110000.

1. Chaque bit est associé à une valeur dont la base est une puissance de deux. Les puissances commencent à 0 et sont incrémentées de droite à gauche. Dans cet exemple, nous aurons :

1	0	1	1	0	0	0	0
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

2. Lorsque le bit est à 1, nous attribuons la valeur de la puissance correspondante. Lorsque le bit est à 0, le résultat est 0.

1	0	1	1	0	0	0	0
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
128	0	32	16	0	0	0	0

3. Faisons la somme de toutes les valeurs :

$$128 + 32 + 16 = 176$$

4. Ainsi, 10110000 en binaire est égal à 176 en décimal.

Masque de sous-réseau

Le masque de sous-réseau (*netmask*) est utilisé conjointement avec l'adresse IP pour déterminer quelle partie de l'adresse IP représente le réseau et quelle partie représente les hôtes. Il a le même format que l'adresse IP, c'est-à-dire qu'il comporte 32 bits répartis en 4 groupes de 8. Par exemple :

Décimal	Binaire	CIDR
255.0.0.0	11111111.00000000.00000000 0.00000000	8
255.255.0.0	11111111.11111111.00000000 0.00000000	16
255.255.255.0	11111111.11111111.11111111 1.00000000	24

En prenant le masque 255.255.0.0 comme exemple, il indique que dans l'adresse IP qui lui est associée, les 16 premiers bits (2 premières décimales) identifient le réseau/sous-réseau et les 16 derniers bits sont utilisés pour identifier sans équivoque les hôtes au sein du réseau.

Le CIDR (*Classless Inter-Domain Routing*) mentionné ci-dessus est lié à une notation simplifiée du masque, qui indique le nombre de bits (1) associés au réseau/sous-réseau. Cette notation est couramment utilisée pour remplacer le format décimal, par exemple /24 au lieu de 255.255.255.0.

Il est intéressant de noter que chaque classe d'IP a un masque standard, comme suit :

Classe	Premier Octet	Plage	Masque par Défaut
A	1-126	1.0.0.0 – 126.255.255.255	255.0.0.0 / 8
B	128-191	128.0.0.0 – 191.255.255.255	255.255.0.0 / 16
C	192-223	192.0.0.0 – 223.255.255.255	255.255.255.0 / 24

En revanche, ce schéma ne signifie pas qu'il s'agit du masque qui sera forcément utilisé. On peut très bien utiliser n'importe quel masque avec n'importe quelle adresse IP, comme nous allons le voir par la suite.

Voici quelques exemples concrets avec des IP et des masques :

```
192.168.8.12 / 255.255.255.0 / 24
```

Plage

```
192.168.8.0 - 192.168.8.255
```

Adresse réseau

```
192.168.8.0
```

Adresse de diffusion

```
192.168.8.255
```

Hôtes

```
192.168.8.1 - 192.168.8.254
```

Dans ce cas, les 3 premiers chiffres (24 premiers bits) de l'adresse IP définissent le réseau et le dernier chiffre identifie les adresses des hôtes, c'est-à-dire que la plage de ce réseau s'étend de 192.168.8.0 à 192.168.8.255.

Nous voilà confrontés à deux concepts importants : Chaque réseau/sous-réseau a 2 adresses réservées. La première adresse de la plage est appelée *adresse réseau*. Dans ce cas 192.168.8.0, qui est utilisée pour identifier le réseau/sous-réseau lui-même. La dernière adresse de la plage est appelée *adresse de diffusion*, dans ce cas 192.168.8.255. Cette adresse de destination est utilisée pour envoyer le même message (paquet) à tous les hôtes IP de ce réseau/sous-réseau.

L'adresse réseau et l'adresse de diffusion ne peuvent pas être utilisées par les machines du réseau. Par conséquent, la liste des IP qui peuvent effectivement être configurées s'étend de 192.168.8.1 à 192.168.8.254.

Prenons maintenant l'exemple de la même IP, mais avec un masque différent :

```
192.168.8.12 / 255.255.0.0 / 16
```

Plage

```
192.168.0.0 - 192.168.255.255
```

Adresse réseau

```
192.168.0.0
```

Adresse de diffusion

192.168.255.255

Hôtes

192.168.0.1 – 192.168.255.254

Voyez comment le changement de masque modifie la plage d'adresses IP au sein du même réseau/sous-réseau.

La division des réseaux par les masques n'est pas limitée aux valeurs par défaut (8, 16, 24). Nous pouvons créer des subdivisions à notre guise, en ajoutant ou en supprimant des bits dans l'identification du réseau, créant ainsi de nouveaux sous-réseaux.

Par exemple :

11111111.11111111.11111111.00000000 = 255.255.255.0 = 24

Si nous voulons subdiviser le réseau ci-dessus en 2, il suffit d'ajouter un bit supplémentaire à l'identification du réseau dans le masque, comme ceci :

11111111.11111111.11111111.10000000 = 255.255.255.128 = 25

On obtient alors les sous-réseaux suivants :

192.168.8.0 - 192.168.8.127
192.168.8.128 - 192.168.8.255

Si nous allons plus loin dans la subdivision du réseau :

11111111.11111111.11111111.11000000 = 255.255.255.192 = 26

Nous aurons :

192.168.8.0 - 192.168.8.63
192.168.8.64 - 192.168.8.127
192.168.8.128 - 192.168.8.191
192.168.8.192 - 192.168.8.255

Notez que dans chaque sous-réseau, nous réservons l'adresse réseau (la première de la plage) et l'adresse de diffusion (la dernière de la plage), de sorte que plus le réseau est subdivisé, moins les hôtes peuvent utiliser efficacement les adresses IP.

Identifier l'adresse réseau et l'adresse de diffusion

Une adresse IP et un masque permettent d'identifier l'adresse réseau et l'adresse de diffusion, et donc de définir la plage d'adresses IP pour le réseau/sous-réseau.

L'adresse réseau est obtenue en utilisant un "ET logique" entre l'adresse IP et le masque dans leurs formats binaires. Prenons l'exemple de l'adresse IP 192.168.8.12 et du masque 255.255.192.

En convertissant le format décimal en format binaire, comme nous l'avons vu plus haut, nous avons :

```
11000000.10101000.00001000.00001100 (192.168.8.12)
11111111.11111111.11111111.11000000 (255.255.255.192)
```

Avec un "ET logique", nous avons 1 et 1 = 1, 0 et 0 = 0, 1 et 0 = 0, donc :

```
11000000.10101000.00001000.00001100 (192.168.8.12)
11111111.11111111.11111111.11000000 (255.255.255.192)
11000000.10101000.00001000.00000000
```

L'adresse réseau de ce sous-réseau est donc 192.168.8.0.

Pour obtenir l'adresse de diffusion, il faut utiliser l'adresse réseau dont tous les bits relatifs à l'adresse de l'hôte sont à 1 :

```
11000000.10101000.00001000.00000000 (192.168.8.0)
11111111.11111111.11111111.11000000 (255.255.255.192)
11000000.10101000.00001000.00111111
```

L'adresse de diffusion est donc 192.168.8.63.

Au total, nous avons :

```
192.168.8.12 / 255.255.255.192 / 26
```

Plage

192.168.8.0 - 192.168.8.63

Adresse réseau

192.168.8.0

Adresse de diffusion

192.168.8.63

Hôtes

192.168.8.1 – 192.168.8.62

Route par défaut

Comme nous l'avons vu jusqu'à présent, les machines qui se trouvent dans le même réseau/sous-réseau logique peuvent communiquer directement via le protocole IP.

Prenons maintenant l'exemple suivant :

Réseau 1

192.168.10.0/24

Réseau 2

192.168.200.0/24

Dans ce cas, la machine 192.168.10.20 ne peut pas envoyer directement un paquet à la machine 192.168.200.100, étant donné qu'elles se trouvent sur des réseaux logiques distincts.

Pour permettre cette communication, un routeur (ou un ensemble de routeurs) est utilisé. Dans cette configuration, un routeur peut également être appelé "passerelle", vu qu'il sert de passerelle entre deux réseaux. Cette machine a accès aux deux réseaux dans la mesure où elle est configurée avec des adresses IP appartenant aux deux réseaux. Par exemple 192.168.10.1 et 192.168.200.1, et pour cette raison elle parvient à être l'intermédiaire dans cette communication.

Pour ce faire, chaque hôte du réseau doit avoir configuré ce que l'on appelle la route par défaut (*default route*). La route par défaut indique l'IP vers laquelle devront être envoyés tous les paquets dont la destination est une IP qui ne fait pas partie du réseau logique de l'hôte.

Dans l'exemple ci-dessus, la route par défaut pour les machines du réseau 192.168.10.0/24 sera l'IP 192.168.10.1 qui est l'IP du routeur/passerelle, alors que la route par défaut pour les

machines du réseau `192.168.200.0/24` sera `192.168.200.1`.

La route par défaut est également utilisée pour que les machines d'un réseau privé (LAN) aient accès à Internet (WAN) par l'intermédiaire d'un routeur.

Exercices guidés

1. En utilisant l'IP 172.16.30.230 et le masque de réseau 255.255.255.224, identifiez :

La notation CIDR pour le masque de réseau	
L'adresse réseau	
L'adresse de diffusion	
Le nombre d'IPs utilisables pour les hôtes de ce sous-réseau	

2. Quel paramètre est requis sur un hôte pour permettre une communication IP avec un hôte situé dans un réseau logique différent ?

Exercices d'approfondissement

1. Pourquoi les plages d'adresses IP commençant par 127 et la plage après 224 ne sont-elles pas comprises dans les classes d'adresses IP A, B ou C ?

2. Le TTL (*Time To Live*) est l'un des champs les plus importants d'un paquet IP. Quelle est la fonction de ce champ et comment fonctionne-t-il ?

3. Expliquez la fonction du NAT et son utilisation.

Résumé

Cette leçon a abordé les principaux concepts du protocole IPv4, qui permet la communication entre les hôtes d'un réseau.

Les principales opérations qu'un professionnel doit connaître pour convertir les IP dans différents formats et pour pouvoir analyser et effectuer les configurations logiques sur les réseaux et les sous-réseaux ont également été abordées.

Voici les thèmes que nous avons abordés :

- Classes d'adresses IP
- IP publiques et privées
- Convertir les adresses IP du format décimal au format binaire et vice versa
- Le masque de sous-réseau (*netmask*)
- Identifier l'adresse réseau et l'adresse de diffusion à partir de l'IP et du masque de sous-réseau
- Route par défaut

Réponses aux exercices guidés

1. En utilisant l'IP 172.16.30.230 et le masque de réseau 255.255.255.224, identifiez :

La notation CIDR pour le masque de réseau	27
L'adresse réseau	172.16.30.224
L'adresse de diffusion	172.16.30.255
Le nombre d'IPs utilisables pour les hôtes de ce sous-réseau	30

2. Quel paramètre est requis sur un hôte pour permettre une communication IP avec un hôte situé dans un réseau logique différent ?

La route par défaut

Réponses aux exercices d'approfondissement

1. Pourquoi les plages d'adresses IP commençant par 127 et la plage après 224 ne sont-elles pas comprises dans les classes d'adresses IP A, B ou C ?

La plage qui commence par 127 est réservée aux adresses de rebouclage *loopback*, utilisées pour les tests et la communication interne entre les processus, comme l'adresse 127.0.0.1. En outre, les adresses supérieures à 224 ne sont pas non plus utilisées comme adresses d'hôte, mais pour la multidiffusion *multicast* et à d'autres fins.

2. Le TTL (*Time To Live*) est l'un des champs les plus importants d'un paquet IP. Quelle est la fonction de ce champ et comment fonctionne-t-il ?

Le TTL définit la durée de vie d'un paquet. Il est implémenté par le biais d'un compteur dans lequel la valeur initiale définie à la source est décrémentée à chaque passerelle/routeur par lequel le paquet passe, ce que l'on appelle également un "saut" (*hop*). Si ce compteur atteint 0, le paquet est supprimé.

3. Expliquez la fonction du NAT et son utilisation.

La fonctionnalité de traduction d'adresse réseau NAT (*Network Address Translation*) permet aux hôtes d'un réseau local qui utilise des IP privées d'avoir accès à Internet comme s'ils y étaient directement connectés, avec l'IP publique utilisée sur la passerelle.



109.1 Leçon 2

Certification :	LPIC-1
Version :	5.0
Thème :	109 Principes de base des réseaux
Objectif :	109.1 Fondamentaux des protocoles internet
Leçon :	2 sur 2

Introduction

Au début de cette thématique, nous avons vu que la pile TCP/IP est composée d'une série de protocoles différents. Pour l'instant, nous avons étudié le protocole IP qui permet la communication entre les machines par le biais d'adresses IP, de masques, de routes, etc.

Pour qu'un hôte puisse accéder à un service disponible sur un autre hôte, en dehors du protocole d'adressage IP au niveau de la couche réseau, il faudra utiliser un protocole au niveau de la couche transport tel que les protocoles TCP et UDP.

Ces protocoles assurent la communication par l'intermédiaire de ports réseau. Ainsi, en plus de la définition d'une IP source et d'une IP destination, on utilisera des ports source et destination pour accéder à un service.

Le port est identifié par un champ de 16 bits, ce qui limite le nombre de ports possibles à 65 535. Les services (destination) utilisent les ports 1 à 1023, que l'on appelle *ports privilégiés* parce qu'ils ont un accès root au système. L'origine de la connexion utilisera la plage de ports comprise entre 1024 et 65 535, appelés *ports non privilégiés* ou ports socket.

Les ports utilisés par chaque type de service sont standardisés et contrôlés par l'IANA (*Internet*

Assigned Numbers Authority). Cela signifie que sur n'importe quel système, le port 22 sera utilisé par le service SSH, le port 80 par le service HTTP et ainsi de suite.

Le tableau ci-dessous présente les principaux services et leurs ports respectifs.

Port	Service
20	FTP (données)
21	FTP (contrôle)
22	SSH (<i>Secure Socket Shell</i>)
23	Telnet (Connexion à distance sans chiffrement)
25	SMTP (<i>Simple Mail Transfer Protocol</i>), Envoi d'e-mails
53	DNS (Système de noms de domaine)
80	HTTP (<i>Hypertext Transfer Protocol</i>)
110	POP3 (<i>Post Office Protocol</i>), Réception d'e-mails
123	NTP (<i>Network Time Protocol</i>)
139	Netbios
143	IMAP (<i>Internet Message Access Protocol</i>), Accès aux e-mails
161	SNMP (<i>Simple Network Management Protocol</i>)
162	SNMPTRAP, Notifications SNMP
389	LDAP (<i>Lightweight Directory Access Protocol</i>)
443	HTTPS (HTTP sécurisé)
465	SMTPS (SMTP sécurisé)
514	RSH (<i>Remote Shell</i>)
636	LDAPS (LDAP sécurisé)
993	IMAPS (IMAP sécurisé)
995	POP3S (POP3 sécurisé)

Sur un système Linux, les ports standard des services sont répertoriés dans le fichier `/etc/services`.

L'identification du port de destination souhaité dans une connexion se fait en utilisant le

caractère : (deux points) après l'adresse IPv4. Ainsi, pour accéder au service HTTPS géré par l'hôte IP 200.216.10.15, le client doit envoyer sa requête à la destination 200.216.10.15:443.

Les services répertoriés ci-dessus ainsi que tous les autres utilisent un protocole de transport en fonction des caractéristiques requises par le service, parmi lesquels on trouve principalement le TCP et l'UDP.

Transmission Control Protocol (TCP)

TCP est un protocole de transport orienté connexion. Cela signifie qu'une connexion est établie entre le client via le port socket et le service via le port standard du service. Le protocole est chargé de veiller à ce que tous les paquets soient transmis correctement, en vérifiant l'intégrité et l'ordre des paquets, y compris la retransmission des paquets perdus suite à des erreurs de réseau.

L'application n'a donc pas besoin de mettre en œuvre ce contrôle du flux de données dans la mesure où il est déjà garanti par le protocole TCP.

User Datagram Protocol (UDP)

UDP établit une connexion entre le client et le service, mais sans contrôler la transmission des données de cette connexion. En d'autres termes, il ne vérifie pas si des paquets ont été perdus, s'ils ne sont pas dans le bon ordre, etc. C'est l'application qui devra réaliser les contrôles nécessaires.

Comme il y a moins de contrôle, l'UDP permet de meilleures performances au niveau du flux de données, ce qui est important pour certains types de services.

Internet Control Message Protocol (ICMP)

ICMP est un protocole de la couche réseau dans la pile TCP/IP et sa fonction principale consiste à analyser et à contrôler les composants du réseau, ce qui permet par exemple de :

- Contrôler le volume du trafic
- Détecter les destinations inaccessibles
- Rediriger les routes
- Vérifier l'état des hôtes distants

C'est le protocole utilisé par la commande `ping`, que nous étudierons dans un autre volet.

IPv6

Pour l'instant, nous avons vu de près la version 4 du protocole IP, c'est-à-dire l'IPv4. Il s'agit de la version standard utilisée dans tous les réseaux et environnements Internet. Or, elle présente des limites, notamment en ce qui concerne le nombre d'adresses disponibles, et avec le fait que tous les appareils seront d'une manière ou d'une autre connectés à l'internet (voir IoT), il est de plus en plus courant d'utiliser la version 6 du protocole IP, communément appelée IPv6.

L'IPv6 apporte une série de changements, de nouvelles implémentations et fonctionnalités, ainsi qu'une nouvelle représentation de l'adresse elle-même.

Chaque adresse IPv6 comporte 128 bits divisés en 8 groupes de 16 bits et représentés par des valeurs hexadécimales.

Par exemple :

```
2001:0db8:85a3:08d3:1319:8a2e:0370:7344
```

Abréviations

IPv6 prévoit la possibilité de raccourcir les adresses dans certaines situations. Regardons l'adresse suivante :

```
2001:0db8:85a3:0000:0000:0000:0000:7344
```

La première possibilité consiste à réduire les chaînes de `0000` à un simple `0`, ce qui donne :

```
2001:0db8:85a3:0:0:0:0:7344
```

Par ailleurs, les chaînes groupées avec une valeur de `0` peuvent être omises, comme ceci :

```
2001:0db8:85a3::7344
```

En revanche, cette forme d'abréviation ne pourra être utilisée qu'une seule fois dans l'adresse. Voyez l'exemple :

```
2001:0db8:85a3:0000:0000:1319:0000:7344
```



```
2001:0db8:85a3:0:0:1319:0:7344
```

```
2001:0db8:85a3::1319:0:7344
```

Types d'adresses IPv6

L'IPv6 distingue trois types d'adresses :

Unicast

Identifie une seule interface réseau. Par défaut, les 64 bits de gauche identifient le réseau et les 64 bits de droite identifient l'interface.

Multicast

Identifie un ensemble d'interfaces réseau. Un paquet envoyé à une adresse multicast sera envoyé à toutes les interfaces qui appartiennent à ce groupe. Bien que similaire, le multicast ne doit pas être confondu avec l'adresse de diffusion (*broadcast*), qui n'existe pas dans le protocole IPv6.

Anycast

Ceci identifie également un ensemble d'interfaces sur le réseau, mais le paquet transmis à une adresse *anycast* ne sera délivré qu'à une seule adresse de ce groupe, et non pas à tout le monde.

Différences entre IPv4 et IPv6

En dehors de l'adresse, on peut relever d'autres différences entre les versions 4 et 6 de l'IP. En voici quelques-unes :

- Les ports des services respectent les mêmes normes et protocoles (TCP, UDP), la seule différence réside dans la représentation de l'ensemble IP et port. En IPv6, l'adresse IP sera protégée par des `[]` (crochets) :

IPv4

```
200.216.10.15:443
```

IPv6

```
[2001:0db8:85a3:08d3:1319:8a2e:0370:7344]:443
```

- IPv6 n'implémente pas exactement la fonction de diffusion telle qu'elle existe dans IPv4. En revanche, on peut obtenir le même résultat en envoyant le paquet à l'adresse `ff02::1`, ce qui permet d'atteindre tous les hôtes du réseau local. C'est un peu comme l'utilisation de `224.0.0.1` sur IPv4 pour la multidiffusion en tant que destination.

- La fonction SLAAC (*Stateless Address Autoconfiguration*) permet aux hôtes IPv6 de s'auto-configurer.
- Le champ TTL (*Time to Live*) de l'IPv4 a été remplacé par le "Hop Limit" dans l'en-tête de l'IPv6.
- Toutes les interfaces IPv6 ont une adresse locale, appelée adresse link-local, préfixée par `fe80::/10`.
- IPv6 met en œuvre le *Neighbor Discovery Protocol* (NDP), qui ressemble à l'ARP utilisé par IPv4, mais avec beaucoup plus de fonctionnalités.

Exercices guidés

1. Quel est le port par défaut pour le protocole SMTP ?

2. Combien de ports différents sont disponibles dans un système ?

3. Quel protocole de transport garantit que tous les paquets sont transmis correctement, en vérifiant leur intégrité et leur ordre ?

4. Quel type d'adresse IPv6 est utilisé pour envoyer un paquet à toutes les interfaces appartenant à un groupe d'hôtes ?

Exercices d'approfondissement

1. Citez 4 exemples de services qui utilisent le protocole TCP par défaut.

2. Quel est le nom du champ du paquet d'en-tête IPv6 qui met en œuvre la même ressource que le TTL sur IPv4 ?

--

3. Quel type d'information le *Neighbor Discovery Protocol* (NDP) est-il en mesure de découvrir ?

--

Résumé

Cette leçon a abordé les principaux protocoles de transport et les services utilisés avec la pile TCP/IP.

Un autre point important concerne la version 6 du protocole IP, y compris les adresses IPv6 et les principales différences avec IPv4.

Les thèmes suivants ont été abordés :

- La corrélation entre les numéros de ports et les services
- TCP (Transmission Control Protocol)
- UDP (User Datagram Protocol)
- ICMP (Internet Control Message Protocol)
- L'adresse IPv6 et ses abréviations
- Les types d'adresses IPv6
- Principales différences entre IPv4 et IPv6

Réponses aux exercices guidés

1. Quel est le port par défaut pour le protocole SMTP ?

25

2. Combien de ports différents sont disponibles dans un système ?

65535

3. Quel protocole de transport garantit que tous les paquets sont transmis correctement, en vérifiant leur intégrité et leur ordre ?

TCP

4. Quel type d'adresse IPv6 est utilisé pour envoyer un paquet à toutes les interfaces appartenant à un groupe d'hôtes ?

Multicast

Réponses aux exercices d'approfondissement

1. Citez 4 exemples de services qui utilisent le protocole TCP par défaut.

FTP, SMTP, HTTP, POP3, IMAP, SSH

2. Quel est le nom du champ du paquet d'en-tête IPv6 qui met en œuvre la même ressource que le TTL sur IPv4 ?

Hop Limit

3. Quel type d'information le *Neighbor Discovery Protocol* (NDP) est-il en mesure de découvrir ?

Le protocole NDP est capable d'obtenir une série d'informations sur le réseau, notamment sur les autres nœuds, les adresses dupliquées, les routes, les serveurs DNS, les passerelles, etc.



109.2 Configuration réseau persistante

Référence aux objectifs de LPI

LPIC-1 version 5.0, Exam 102, Objective 109.2

Valeur

4

Domaines de connaissance les plus importants

- Compréhension de la configuration TCP/IP élémentaire
- Configuration ethernet et wi-fi network à partir de NetworkManager
- Connaissance de base de systemd-networkd

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

- `/etc/hostname`
- `/etc/hosts`
- `/etc/nsswitch.conf`
- `/etc/resolv.conf`
- `nmcli`
- `hostnamectl`
- `ifup`
- `ifdown`



109.2 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	109 Principes de base des réseaux
Objectif :	109.2 Configuration persistante du réseau
Leçon :	1 sur 2

Introduction

Dans un réseau TCP/IP, chaque nœud doit configurer sa carte réseau pour qu'elle réponde aux exigences du réseau, faute de quoi ils ne pourront pas communiquer entre eux. Par conséquent, l'administrateur du système doit fournir la configuration de base de sorte que le système d'exploitation soit en mesure de configurer l'interface réseau appropriée, ainsi que de s'identifier soi-même et de déterminer les caractéristiques de base du réseau chaque fois qu'il démarre.

Les paramètres du réseau ne dépendent pas des systèmes d'exploitation, mais ces derniers ont chacun leur propre méthode pour enregistrer et appliquer ces paramètres. Les systèmes Linux s'appuient sur des configurations stockées dans des fichiers texte dans le répertoire `/etc` pour activer la connexion au réseau au moment du démarrage. Il est utile de savoir comment ces fichiers sont utilisés pour éviter toute perte de connectivité due à une mauvaise configuration locale.

L'interface réseau

L'*interface réseau* est le terme par lequel le système d'exploitation désigne le canal de communication configuré pour fonctionner avec le matériel réseau attaché au système, comme

un périphérique Ethernet ou Wi-Fi. Une exception à cette règle est l'interface *loopback* (boucle locale), que le système d'exploitation utilise lorsqu'il a besoin d'établir une connexion avec lui-même, mais l'objectif principal d'une interface réseau consiste à fournir une voie par laquelle les données locales peuvent être envoyées et les données distantes peuvent être reçues. Si l'interface réseau n'est pas correctement configurée, le système d'exploitation ne pourra pas communiquer avec les autres machines du réseau.

Dans la plupart des cas, les paramètres corrects de l'interface sont définis par défaut ou personnalisés lors de l'installation du système d'exploitation. Néanmoins, ces paramètres doivent souvent être vérifiés voire modifiés lorsque la communication ne fonctionne pas correctement ou lorsque le comportement de l'interface doit être adapté.

Il existe une série de commandes Linux pour afficher les interfaces réseau présentes sur le système, mais elles ne sont pas toutes disponibles dans toutes les distributions. En revanche, la commande `ip` fait partie des outils réseau de base fournis avec toutes les distributions Linux et peut être utilisée pour afficher ces interfaces. La commande complète pour afficher les interfaces est `ip link show` :

```
$ ip link show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT group
default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: enp3s5: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP mode DEFAULT
group default qlen 1000
    link/ether 00:16:3e:8d:2b:5b brd ff:ff:ff:ff:ff:ff
```

Si elle est disponible, la commande `nmcli device` pourra également être utilisée :

```
$ nmcli device
DEVICE      TYPE      STATE      CONNECTION
enp3s5      ethernet  connected  Gigabit Powerline Adapter
lo          loopback  unmanaged  --
```

Les commandes montrées dans l'exemple ne modifient aucun paramètre du système et peuvent donc être exécutées par un utilisateur non privilégié. Les deux commandes affichent deux interfaces réseau : `lo` (la boucle locale) et `enp3s5` (une interface Ethernet).

Les postes de travail et les ordinateurs portables sous Linux ont généralement deux ou trois interfaces réseau prédéfinies, une pour la boucle locale et les autres attribuées au matériel réseau détecté par le système. En revanche, les serveurs et les équipements réseau sous Linux peuvent

avoir des dizaines d'interfaces réseau, mais les mêmes principes restent valables. L'abstraction fournie par le système d'exploitation permet de configurer les interfaces réseau en utilisant les mêmes méthodes, indépendamment du matériel utilisé.

Connaître les détails du matériel utilisé pour une interface peut néanmoins s'avérer utile pour mieux comprendre ce qui se passe lorsque la communication ne fonctionne pas comme prévu. Dans un système doté d'un certain nombre d'interfaces réseau, il n'est pas toujours évident de savoir laquelle correspond au Wi-Fi et laquelle correspond à l'Ethernet, par exemple. C'est pourquoi Linux utilise une convention de nommage des interfaces qui permet d'identifier quelle interface réseau correspond à quel périphérique et à quel port.

Les noms des interfaces

Les anciennes distributions Linux nommaient les interfaces réseau Ethernet `eth0`, `eth1`, etc. numérotées selon l'ordre dans lequel le noyau identifiait les périphériques. Les interfaces sans fil étaient nommées `wlan0`, `wlan1`, etc. En revanche, cette convention de nommage ne précise pas quel port Ethernet spécifique correspond à l'interface `eth0`, par exemple. Selon la manière dont le matériel était détecté, il était même possible que deux interfaces réseau changent de nom après un redémarrage.

Pour surmonter cette ambiguïté, les systèmes Linux modernes utilisent une convention de dénomination prévisible pour les interfaces réseau, qui établit un rapport plus étroit entre le nom de l'interface et la connexion matérielle correspondante.

Dans les distributions Linux qui utilisent le système d'initialisation `systemd`, tous les noms des interfaces commencent par un préfixe de deux caractères qui indique le type de l'interface :

en

Ethernet

ib

InfiniBand

sl

IP série (slip)

wl

Réseau local sans fil (WLAN)

ww

Réseau étendu sans fil (WWAN)

Les règles suivantes sont utilisées par le système d'exploitation pour nommer et numéroté les interfaces réseau, par ordre de priorité décroissante :

1. Nommer l'interface en fonction de l'index fourni par le BIOS ou par le firmware des périphériques embarqués, par exemple `eno1`.
2. Nommer l'interface en fonction de l'index du port PCI express tel qu'il est fourni par le BIOS ou le firmware, par exemple `ens1`.
3. Nommer l'interface en fonction de son adresse sur le bus correspondant, par exemple `enp3s5`.
4. Nommer l'interface à partir de son adresse MAC, par exemple `enx78e7d1ea46da`.
5. Nommer l'interface en respectant l'ancienne convention, par exemple `eth0`.

On peut supposer par exemple que l'interface réseau `enp3s5` a été nommée ainsi parce qu'elle ne correspondait pas aux deux premières méthodes de dénomination et que son adresse dans le bus et le port correspondants ont été utilisés à la place. L'adresse du périphérique `03:05.0` trouvée dans la sortie de la commande `lspci` révèle le périphérique associé :

```
$ lspci | fgrep Ethernet
```

```
03:05.0 Ethernet controller: Realtek Semiconductor Co., Ltd. RTL-8110SC/8169SC Gigabit Ethernet (rev 10)
```

Les interfaces réseau sont créées par le noyau Linux lui-même, mais il existe une série de commandes pour interagir avec elles. En temps normal, la configuration se fait automatiquement et il n'est pas nécessaire de modifier les paramètres manuellement. En revanche, avec le nom de l'interface, il est possible d'indiquer au noyau comment procéder pour la configurer si c'est nécessaire.

Gestion des interfaces

Au fil des années, une série de programmes ont été développés pour interagir avec les fonctionnalités réseau fournies par le noyau Linux. Même si l'ancienne commande `ifconfig` peut encore être utilisée pour effectuer des configurations et des requêtes simples sur les interfaces, elle est désormais considérée comme obsolète en raison de son support limité des interfaces non-Ethernet. La commande `ifconfig` a été remplacée par la commande `ip` capable de gérer de nombreux autres aspects des interfaces TCP/IP comme les routes et les tunnels.

Les possibilités de la commande `ip` peuvent paraître excessives pour la plupart des tâches ordinaires, c'est pourquoi il existe des commandes auxiliaires pour faciliter l'activation et la configuration des interfaces réseau. Les commandes `ifup` et `ifdown` peuvent être utilisées pour configurer les interfaces réseau basées sur les définitions d'interfaces trouvées dans le fichier

`/etc/network/interfaces`. Même si elles peuvent être invoquées manuellement, ces commandes sont normalement exécutées automatiquement au démarrage du système.

Toutes les interfaces réseau gérées par `ifup` et `ifdown` doivent être répertoriées dans le fichier `/etc/network/interfaces`. Le format utilisé dans ce fichier est simple : les lignes qui commencent par le mot `auto` identifient les interfaces physiques à afficher lorsque `ifup` est exécuté avec l'option `-a`. Le nom de l'interface doit suivre le mot `auto` sur la même ligne. Toutes les interfaces marquées `auto` sont activées au démarrage, dans l'ordre où elles sont inscrites.

WARNING

Les méthodes de configuration réseau utilisées par `ifup` et `ifdown` ne sont pas standardisées dans toutes les distributions Linux. CentOS, par exemple, conserve les paramètres des interfaces dans des fichiers individuels dans le répertoire `/etc/sysconfig/network-scripts/` et le format de configuration utilisé dans ces fichiers est légèrement différent du format utilisé dans `/etc/network/interfaces`.

La configuration de l'interface est inscrite sur une autre ligne. Elle commence par le mot `iface` suivi du nom de l'interface, du nom de la famille d'adresses utilisée par l'interface et du nom de la méthode utilisée pour configurer l'interface. L'exemple ci-dessous montre un fichier de configuration de base pour les interfaces `lo` (loopback) et `enp3s5` :

```
auto lo
iface lo inet loopback

auto enp3s5
iface enp3s5 inet dhcp
```

La famille d'adresses sera `inet` pour les réseaux TCP/IP. Il existe également un support pour les réseaux IPX (`ipx`) et IPv6 (`inet6`). Les interfaces de boucle locale utilisent la méthode de configuration `loopback`. Avec la méthode `dhcp`, l'interface utilisera les paramètres IP fournis par le serveur DHCP du réseau. Les paramètres de l'exemple permettent l'exécution de la commande `ifup` en utilisant le nom de l'interface `enp3s5` comme argument :

```
# ifup enp3s5
Internet Systems Consortium DHCP Client 4.4.1
Copyright 2004-2018 Internet Systems Consortium.
All rights reserved.
For info, please visit https://www.isc.org/software/dhcp/

Listening on LPF/enp3s5/00:16:3e:8d:2b:5b
Sending on   LPF/enp3s5/00:16:3e:8d:2b:5b
```

```

Sending on Socket/fallback
DHCPDISCOVER on enp3s5 to 255.255.255.255 port 67 interval 4
DHCPOFFER of 10.90.170.158 from 10.90.170.1
DHCPREQUEST for 10.90.170.158 on enp3s5 to 255.255.255.255 port 67
DHCPACK of 10.90.170.158 from 10.90.170.1
bound to 10.90.170.158 -- renewal in 1616 seconds.

```

Dans cet exemple, la méthode choisie pour l'interface `enp3s5` était `dhcp`, donc la commande `ifup` a appelé un client DHCP pour obtenir les paramètres IP du serveur DHCP. De même, la commande `ifdown enp3s5` pourra être utilisée pour désactiver l'interface.

Dans les réseaux sans serveur DHCP, la méthode `static` peut être utilisée avec les paramètres IP fournis manuellement dans `/etc/network/interfaces`. Par exemple :

```

iface enp3s5 inet static
    address 192.168.1.2/24
    gateway 192.168.1.1

```

Les interfaces qui utilisent la méthode `static` n'ont pas besoin d'une directive `auto` correspondante, étant donné qu'elles sont activées lors de la détection du matériel réseau.

Si la même interface a plus d'une entrée `iface`, toutes les adresses et les options configurées vont être appliquées lors de son activation. Cela est utile pour configurer les adresses IPv4 et IPv6 sur la même interface, ainsi que pour configurer plusieurs adresses du même type sur une seule interface.

Noms locaux et distants

Une configuration TCP/IP fonctionnelle n'est que la première étape vers un réseau pleinement utilisable. En dehors de l'identification des nœuds du réseau par leur adresse IP, le système doit être capable de les identifier par des noms faciles à comprendre par les êtres humains.

Le nom par lequel le système s'identifie est personnalisable et c'est une bonne pratique de le définir, même si la machine n'est pas censée rejoindre un réseau. Le nom local correspond souvent au nom réseau de la machine, mais ce n'est pas toujours le cas. Si le fichier `/etc/hostname` existe, le système d'exploitation utilisera la première ligne comme nom local, appelé simplement *hostname*. Les lignes qui commencent par `#` dans `/etc/hostname` sont ignorées.

Le fichier `/etc/hostname` peut être édité directement, mais le nom d'hôte de la machine peut également être défini avec la commande `hostnamectl`. Lorsqu'elle est invoquée avec la sous-

commande `set-hostname`, la commande `hostnamectl` va prendre le nom fourni en argument et l'écrire dans `/etc/hostname` :

```
# hostnamectl set-hostname storage
# cat /etc/hostname
storage
```

Le nom d'hôte défini dans `/etc/hostname` est le nom d'hôte *statique*, autrement dit, le nom utilisé pour initialiser le nom d'hôte du système au démarrage. Le nom d'hôte statique peut être une chaîne de caractères quelconque d'une longueur maximale de 64 caractères. En revanche, il est conseillé de le composer uniquement de caractères ASCII en minuscules, sans espaces et sans points. Il doit également se limiter au format autorisé pour les dénominations de domaines DNS, même s'il ne s'agit pas là d'une exigence stricte.

La commande `hostnamectl` peut définir deux autres types de noms d'hôtes en dehors du nom d'hôte statique :

Nom d'hôte amélioré (*pretty hostname*)

Contrairement au nom d'hôte statique, le nom d'hôte amélioré pourra comporter toutes sortes de caractères spéciaux. Il peut être utilisé pour définir un nom plus descriptif pour la machine, comme par exemple "LAN Shared Storage" :

```
# hostnamectl --pretty set-hostname "LAN Shared Storage"
```

Nom d'hôte transitoire (*transient hostname*)

Utilisé lorsque le nom d'hôte statique n'est pas défini ou lorsqu'il s'agit du nom `localhost` par défaut. Le nom d'hôte transitoire est normalement le nom défini avec d'autres configurations automatiques, mais il peut également être modifié par la commande `hostnamectl`, par exemple :

```
# hostnamectl --transient set-hostname generic-host
```

Si les deux options `--pretty` et `--transient` ne sont pas utilisées, alors les trois types de noms d'hôte seront définis avec le nom fourni en argument. Pour définir le nom d'hôte statique mais pas les noms d'hôte amélioré et transitoire, l'option `--static` devra être spécifiée. Dans tous les cas, seul le nom d'hôte statique sera enregistré dans le fichier `/etc/hostname`. La commande `hostnamectl` peut également être utilisée pour afficher une série d'informations descriptives relatives à l'identité du système en cours d'exécution :

```
$ hostnamectl status
  Static hostname: storage
    Pretty hostname: LAN Shared Storage
  Transient hostname: generic-host
    Icon name: computer-server
    Chassis: server
  Machine ID: d91962a957f749bbaf16da3c9c86e093
    Boot ID: 8c11dcab9c3d4f5aa53f4f4e8fdc6318
  Operating System: Debian GNU/Linux 10 (buster)
    Kernel: Linux 4.19.0-8-amd64
  Architecture: x86-64
```

C'est l'action par défaut de la commande `hostnamectl`, de sorte que la sous-commande `status` peut être omise.

En ce qui concerne le nom des nœuds distants du réseau, le système d'exploitation peut utiliser deux méthodes de base pour faire correspondre les noms et les adresses IP : une source locale ou un serveur distant pour traduire les noms en adresses IP et vice-versa. Ces méthodes peuvent être complémentaires et leur ordre de priorité est défini dans le fichier de configuration `/etc/nsswitch.conf` (*Name Service Switch*). Ce fichier est utilisé par le système et les applications pour déterminer non seulement les sources de correspondances entre les noms et les adresses IP, mais aussi les sources à partir desquelles on obtient des informations sur les services de noms dans une série de catégories, appelées *bases de données*.

La base de données `hosts` maintient les correspondances entre les noms d'hôtes et leurs adresses. La ligne dans `/etc/nsswitch.conf` qui commence par `hosts` définit les services chargés de fournir les associations pour cette base de données :

```
hosts: files dns
```

Dans cet exemple, `files` et `dns` sont les noms des services qui spécifient comment le processus de recherche des noms d'hôtes fonctionnera. Tout d'abord, le système va chercher des correspondances dans les fichiers locaux, puis il va demander au service DNS de les trouver.

Le fichier local pour la base de données des hôtes est `/etc/hosts`, un simple fichier texte qui associe les adresses IP aux noms d'hôtes, une ligne par adresse IP, par exemple :

```
127.0.0.1 localhost
```

L'adresse IP `127.0.0.1` est l'adresse par défaut de l'interface *loopback*, d'où son association avec le

nom *localhost*.

On peut également lier des alias éventuels à la même adresse IP. Les alias peuvent fournir des écritures alternatives ou des noms d'hôtes plus courts et doivent être ajoutés à la fin de la ligne, par exemple :

```
192.168.1.10 foo.mydomain.org foo
```

Voici les règles de mise en forme du fichier `/etc/hosts` :

- Les champs sont séparés par un nombre quelconque d'espaces et, ou de tabulations.
- Le texte à partir du caractère `#` jusqu'à la fin de la ligne est un commentaire et doit être ignoré.
- Les noms d'hôtes ne peuvent contenir que des caractères alphanumériques, des signes moins et des points.
- Les noms d'hôtes doivent commencer par un caractère alphabétique et se terminer par un caractère alphanumérique.

Les adresses IPv6 peuvent également être ajoutées à `/etc/hosts`. L'entrée suivante fait référence à l'adresse IPv6 *loopback* :

```
:::1 localhost ip6-localhost ip6-loopback
```

Après la spécification du service `files`, la spécification `dns` indique au système de demander à un service DNS l'association nom/IP désirée. L'ensemble des routines responsables de cette méthode est appelé *resolver* et son fichier de configuration est `/etc/resolv.conf`. L'exemple suivant montre un fichier `/etc/resolv.conf` générique qui contient des entrées pour les serveurs DNS publics de Google :

```
nameserver 8.8.4.4
nameserver 8.8.8.8
```

Comme le montre l'exemple, le mot clé `nameserver` indique l'adresse IP du serveur DNS. Un seul serveur de noms est requis, mais on peut en indiquer jusqu'à trois. Les serveurs supplémentaires seront utilisés comme solution de secours. En l'absence de serveurs de noms, le comportement par défaut consiste à utiliser le serveur de noms de la machine locale.

Le résolveur peut être configuré pour ajouter automatiquement le domaine aux noms avant de les interroger sur le serveur de noms. Par exemple :

```
nameserver 8.8.4.4
nameserver 8.8.8.8
domain mydomain.org
search mydomain.net mydomain.com
```

L'entrée `domain` définit `mydomain.org` comme le nom de domaine local, de sorte que les requêtes pour des noms dans ce domaine seront autorisées à utiliser des noms courts relatifs au domaine local. L'entrée `search` a un but similaire, mais elle accepte une liste de domaines à tester lorsqu'un nom court est fourni. Par défaut, elle ne contient que le nom de domaine local.

Exercices guidés

1. Quelles commandes peuvent être utilisées pour afficher la liste des cartes réseau présentes dans le système ?

2. De quel type est une carte réseau dont le nom d'interface est `wlo1` ?

3. Quel rôle joue le fichier `/etc/network/interfaces` au démarrage ?

4. Quelle entrée dans `/etc/network/interfaces` configure l'interface `eno1` pour qu'elle obtienne ses paramètres IP par DHCP ?

Exercices d'approfondissement

1. Comment la commande `hostnamectl` pourrait-elle être utilisée pour changer uniquement le nom d'hôte statique (*static*) de la machine locale en `firewall` ?

2. Quels détails, autres que les noms d'hôtes, peuvent être modifiés par la commande `hostnamectl` ?

3. Quelle entrée dans `/etc/hosts` associe les deux noms `firewall` et `router` à l'IP `10.8.0.1` ?

4. Comment le fichier `/etc/resolv.conf` pourrait-il être modifié afin d'envoyer toutes les requêtes DNS à `1.1.1.1` ?

Résumé

Cette leçon explique les modifications persistantes à apporter à la configuration du réseau local à l'aide de fichiers et de commandes standard de Linux. Linux s'attend à ce que les paramètres TCP/IP se trouvent à des endroits spécifiques et il peut être nécessaire de les modifier lorsque les paramètres par défaut ne sont pas appropriés. La leçon aborde les sujets suivants :

- Comment Linux identifie les interfaces réseau.
- Activation de l'interface au démarrage et configuration de base de l'IP.
- Comment le système d'exploitation associe les noms aux hôtes.

Voici les concepts, les commandes et les procédures abordés :

- Les conventions de nommage des interfaces.
- Afficher les interfaces réseau avec `ip` et `nmcli`.
- Activer une interface avec `ifup` et `ifdown`.
- La commande `hostnamectl` et le fichier `/etc/hostname`.
- Les fichiers `/etc/nsswitch.conf`, `/etc/hosts` et `/etc/resolv.conf`.

Réponses aux exercices guidés

1. Quelles commandes peuvent être utilisées pour afficher la liste des cartes réseau présentes dans le système ?

Les commandes `ip link show`, `nmcli device` et l'ancienne commande `ifconfig`.

2. De quel type est une carte réseau dont le nom d'interface est `wlo1` ?

Le nom commence par `wl`, il s'agit donc d'un adaptateur sans fil.

3. Quel rôle joue le fichier `/etc/network/interfaces` au démarrage ?

Il contient les configurations utilisées par la commande `ifup` pour activer les interfaces correspondantes pendant le démarrage.

4. Quelle entrée dans `/etc/network/interfaces` configure l'interface `eno1` pour qu'elle obtienne ses paramètres IP par DHCP ?

La ligne `iface eno1 inet dhcp`.

Réponses aux exercices d'approfondissement

1. Comment la commande `hostnamectl` pourrait-elle être utilisée pour changer uniquement le nom d'hôte statique (*static*) de la machine locale en `firewall` ?

Avec l'option `--static` : `hostnamectl --static set-hostname firewall`.

2. Quels détails, autres que les noms d'hôtes, peuvent être modifiés par la commande `hostnamectl` ?

`hostnamectl` peut également définir l'icône par défaut de la machine locale, son type de châssis, l'emplacement et l'environnement de déploiement.

3. Quelle entrée dans `/etc/hosts` associe les deux noms `firewall` et `router` à l'IP `10.8.0.1` ?

La ligne `10.8.0.1 firewall router`.

4. Comment le fichier `/etc/resolv.conf` pourrait-il être modifié afin d'envoyer toutes les requêtes DNS à `1.1.1.1` ?

En utilisant `nameserver 1.1.1.1` comme seule entrée de serveur de noms.



109.2 Leçon 2

Certification :	LPIC-1
Version :	5.0
Thème :	109 Principes de base des réseaux
Objectif :	109.2 Configuration persistante du réseau
Leçon :	2 sur 2

Introduction

Linux prend en charge à peu près toutes les technologies réseau utilisées pour connecter les serveurs, les conteneurs, les machines virtuelles, les ordinateurs de bureau et les appareils mobiles. Les connexions entre tous ces nœuds réseau peuvent être dynamiques et hétérogènes, ce qui nécessite une gestion appropriée par le système d'exploitation qui tourne dessus.

Dans le passé, les distributions avaient élaboré leurs propres solutions personnalisées pour gérer l'infrastructure dynamique du réseau. Aujourd'hui, des outils comme *NetworkManager* et *systemd* fournissent des fonctionnalités plus complètes et intégrées pour répondre à toutes les demandes spécifiques.

NetworkManager

La plupart des distributions Linux adoptent le service *NetworkManager* pour configurer et contrôler les connexions réseau du système. L'objectif de *NetworkManager* est de rendre la configuration du réseau aussi simple et automatique que possible. Lorsqu'on utilise le protocole DHCP, par exemple, *NetworkManager* prend en charge les changements de route, la recherche d'adresses IP et les mises à jour de la liste locale des serveurs DNS, si nécessaire. Si vous avez à la

fois une connexion filaire et une connexion sans fil, NetworkManager donnera la priorité à la connexion filaire par défaut. Dans la mesure du possible, NetworkManager va essayer de garder au moins une connexion active en permanence.

NOTE

Une requête DHCP (*Dynamic Host Configuration Protocol*) est généralement envoyée par la carte réseau dès que le lien avec le réseau est établi. Le serveur DHCP actif sur le réseau répond alors avec les paramètres (adresse IP, masque de réseau, route par défaut, etc.) que le demandeur devra utiliser pour communiquer via le protocole IP.

Par défaut, le service NetworkManager gère les interfaces réseau qui ne sont pas mentionnées dans le fichier `/etc/network/interfaces`. Cela permet de ne pas interférer avec les autres méthodes de configuration éventuellement présentes, et donc de ne modifier que les interfaces qui ne sont pas contrôlées autrement.

Le service NetworkManager tourne en arrière-plan avec les droits root et prend les mesures nécessaires pour garder le système connecté. Les utilisateurs normaux peuvent créer et modifier des connexions réseau à l'aide de logiciels clients qui, même s'ils ne disposent pas eux-mêmes des privilèges root, sont capables de communiquer avec le service sous-jacent pour effectuer les actions requises.

Les logiciels clients pour NetworkManager existent aussi bien en ligne de commande que pour l'environnement graphique. Pour ce dernier, le client est un accessoire de l'environnement de bureau (comme *nm-tray*, *network-manager-gnome*, *nm-applet* ou *plasma-nm*) et il est généralement accessible par une icône dans la zone de notification ou depuis l'outil de configuration du système.

NetworkManager fournit deux clients pour la ligne de commande : `nmcli` et `nmtui`. Les deux programmes offrent les mêmes fonctionnalités de base, mais `nmtui` a une interface basée sur Curses alors que `nmcli` est une commande plus complète qui peut également être utilisée dans des scripts. La commande `nmcli` regroupe toutes les propriétés réseau contrôlées par le NetworkManager dans des catégories appelées *objets* :

general

L'état général et le fonctionnement de NetworkManager.

networking

Gestion globale du réseau.

radio

Commutateurs radio de NetworkManager.

connection

Connexions de NetworkManager.

device

Périphériques gérés par NetworkManager.

agent

Agent des données confidentielles ou polkit de NetworkManager.

monitor

Contrôle des modifications apportées à NetworkManager.

Le nom de l'objet est l'argument principal de la commande `nmcli`. Pour afficher l'état général de la connectivité du système, par exemple, l'objet `general` devra être fourni en argument :

```
$ nmcli general
STATE      CONNECTIVITY  WIFI-HW  WIFI    WWAN-HW  WWAN
connected  full          enabled  enabled  enabled  enabled
```

La colonne `STATE` indique si le système est connecté à un réseau ou non. Si la connexion est limitée en raison d'une mauvaise configuration externe ou une restriction d'accès, alors la colonne `CONNECTIVITY` ne rapportera pas l'état de connectivité `full`. Si `Portal` apparaît dans la colonne `CONNECTIVITY`, cela signifie que des étapes d'authentification supplémentaires (généralement via le navigateur web) sont nécessaires pour achever le processus de connexion. Les autres colonnes indiquent l'état des connexions sans fil (s'il y en a), soit `WIFI`, soit `WWAN` (Wide Wireless Area Network, c'est-à-dire les réseaux cellulaires). Le suffixe `HW` indique que l'état correspond au périphérique réseau plutôt qu'à la connexion réseau du système, c'est-à-dire qu'il indique si le matériel est activé ou désactivé pour économiser de l'énergie.

En dehors de l'argument objet, `nmcli` a également besoin d'un argument commande pour s'exécuter. La commande `status` est utilisée par défaut en l'absence d'argument commande, donc la commande `nmcli general` est en fait interprétée comme `nmcli general status`.

Il n'est guère nécessaire de faire quelque chose lorsque l'adaptateur réseau est connecté directement au point d'accès par le biais de câbles, mais les réseaux sans fil nécessitent une interaction supplémentaire pour accepter de nouveaux utilisateurs. `nmcli` facilite le processus de connexion et enregistre les paramètres pour se connecter automatiquement à l'avenir, ce qui est très pratique pour les ordinateurs portables ou tout autre appareil mobile.

Avant de se connecter au Wi-Fi, il est pratique de commencer par afficher la liste des réseaux disponibles dans la zone locale. Si le système possède un adaptateur Wi-Fi fonctionnel, alors

L'objet `device` l'utilisera pour scanner les réseaux disponibles avec la commande `nmcli device wifi list` :

```
$ nmcli device wifi list
```

IN-USE	BSSID	SSID	MODE	CHAN	RATE	SIGNAL	BARS	SECURITY
	90:F6:52:C5:FA:12	Hypnotoad	Infra	11	130 Mbit/s	67		WPA2
	10:72:23:C7:27:AC	Jumbao	Infra	1	130 Mbit/s	55		WPA2
	00:1F:33:33:E9:BE	NETGEAR	Infra	1	54 Mbit/s	35		WPA1 WPA2
	A4:33:D7:85:6D:B0	AP53	Infra	11	130 Mbit/s	32		WPA1 WPA2
	98:1E:19:1D:CC:3A	Bruma	Infra	1	195 Mbit/s	22		WPA1 WPA2

La plupart des utilisateurs vont probablement se servir du nom dans la colonne `SSID` pour identifier le réseau qui les intéresse. Par exemple, la commande `nmcli` pourra se connecter au réseau `Hypnotoad` en utilisant à nouveau l'objet `device` :

```
$ nmcli device wifi connect Hypnotoad
```

Si la commande est exécutée dans un terminal graphique, une boîte de dialogue s'affichera pour demander le mot de passe du réseau. Si la commande est exécutée dans une console texte, le mot de passe pourra être fourni en même temps que les autres arguments :

```
$ nmcli device wifi connect Hypnotoad password MyPassword
```

Si le réseau Wi-Fi cache son SSID, `nmcli` pourra toujours s'y connecter avec les arguments supplémentaires `hidden yes` :

```
$ nmcli device wifi connect Hypnotoad password MyPassword hidden yes
```

Si le système a plus d'un adaptateur Wi-Fi, celui qui doit être utilisé pourra être indiqué avec `ifname`. Par exemple, pour se connecter en utilisant l'adaptateur `wlo1` :

```
$ nmcli device wifi connect Hypnotoad password MyPassword ifname wlo1
```

Une fois la connexion établie, `NetworkManager` va lui donner le nom du SSID correspondant (s'il s'agit d'une connexion Wi-Fi) et le conserver pour les connexions ultérieures. Les noms des connexions et leurs UUIDs sont affichés par la commande `nmcli connection show` :

```
$ nmcli connection show
```

NAME	UUID	TYPE	DEVICE
Ethernet	53440255-567e-300d-9922-b28f0786f56e	ethernet	enp3s5
tun0	cae685e1-b0c4-405a-8ece-6d424e1fb5f8	tun	tun0
Hypnotoad	6fdec048-bcc5-490a-832b-da83d8cb7915	wifi	wlo1
4G	a2cf4460-0cb7-42e3-8df3-ccb927f2fd88	gsm	--

Le type de chaque connexion est affiché — il peut s’agir de `ethernet`, `wifi`, `tun`, `gsm`, `bridge`, etc. — ainsi que le périphérique auquel elles sont associées respectivement. Pour exécuter une action sur une connexion spécifique, il faut fournir son nom ou son UUID en argument. Pour désactiver la connexion Hypnotoad, par exemple :

```
$ nmcli connection down Hypnotoad
Connection 'Hypnotoad' successfully deactivated
```

De même, la commande `nmcli connection up Hypnotoad` pourra être utilisée pour établir la connexion, étant donné qu’elle est désormais enregistrée par NetworkManager. Le nom de l’interface peut également être utilisé pour se reconnecter, mais dans ce cas, c’est plutôt l’objet `device` qu’il faudra invoquer :

```
$ nmcli device disconnect wlo2
Device 'wlo1' successfully disconnected.
```

Le nom de l’interface peut également être utilisé pour rétablir la connexion :

```
$ nmcli device connect wlo2
Device 'wlo1' successfully activated with '833692de-377e-4f91-a3dc-d9a2b1fcf6cb'.
```

Notez que l’UUID de la connexion change à chaque fois que la connexion est établie. Il vaut donc mieux utiliser le nom pour plus de cohérence.

Si la carte sans fil est disponible mais qu’elle n’est pas utilisée, elle peut être désactivée pour économiser de l’énergie. Dans ce cas, c’est l’objet `radio` qui devra être passé à `nmcli` :

```
$ nmcli radio wifi off
```

Bien évidemment, la connexion sans fil peut être réactivée avec la commande `nmcli radio wifi on`.

Une fois les connexions établies, aucune interaction manuelle ne sera nécessaire à l’avenir, étant

donné que NetworkManager identifie les réseaux connus disponibles et s’y connecte automatiquement. En cas de besoin, NetworkManager dispose de plugins capables d’étendre ses fonctionnalités, comme le plugin pour les connexions VPN.

systemd-networkd

Alternativement, les systèmes basés sur systemd peuvent utiliser les démons intégrés à ce dernier pour gérer la connectivité réseau : `systemd-networkd` pour contrôler les interfaces réseau et `systemd-resolved` pour gérer la résolution des noms locaux. Ces services sont rétrocompatibles avec les anciennes méthodes de configuration de Linux, mais la configuration des interfaces réseau en particulier présente des caractéristiques qui méritent qu’on s’y intéresse.

Les fichiers de configuration utilisés par `systemd-networkd` pour configurer les interfaces réseau se trouvent dans l’un des trois répertoires suivants :

/lib/systemd/network

Le répertoire réseau du système.

/run/systemd/network

Le répertoire d’exécution volatile du réseau.

/etc/systemd/network

Le répertoire d’administration réseau local.

Les fichiers sont traités dans l’ordre lexicographique. Il vaut donc mieux commencer leur nom par des chiffres pour faciliter la lecture et la définition de l’ordre.

Les fichiers dans `/etc` ont la priorité la plus élevée, tandis que les fichiers dans `/run` ont la priorité sur les fichiers du même nom dans `/lib`. Cela veut dire que si des fichiers de configuration dans différents répertoires ont le même nom, alors `systemd-networkd` va ignorer les fichiers de priorité inférieure. Cette séparation des fichiers permet de changer les paramètres de l’interface sans avoir à modifier les fichiers originaux : les modifications peuvent être placées dans `/etc/systemd/network` pour écraser celles de `/lib/systemd/network`.

Le rôle de chaque fichier de configuration dépend de son suffixe. Les noms de fichiers qui se terminent par `.netdev` sont utilisés par `systemd-networkd` pour créer des périphériques réseau virtuels, comme les périphériques *bridge* ou *tun*. Les fichiers qui se terminent par `.link` définissent des configurations de bas niveau pour l’interface réseau correspondante. `systemd-networkd` détecte et configure automatiquement les périphériques réseau au fur et à mesure qu’ils apparaissent — et ignore les périphériques déjà configurés par d’autres moyens — il n’y a donc pas besoin d’ajouter ces fichiers dans la plupart des cas.

Le suffixe le plus important est `.network`. Les fichiers qui utilisent ce suffixe peuvent être utilisés pour configurer des adresses réseau et des routes. Comme pour les autres types de fichiers de configuration, le nom du fichier définit l'ordre dans lequel le fichier sera traité. L'interface réseau à laquelle le fichier de configuration fait référence est définie dans la section `[Match]` à l'intérieur du fichier.

Par exemple, l'interface réseau Ethernet `enp3s5` peut être sélectionnée dans le fichier `/etc/systemd/network/30-lan.network` en utilisant l'entrée `Name=enp3s5` dans la section `[Match]` :

```
[Match]
Name=enp3s5
```

Une liste de noms séparés par des espaces est également possible pour faire correspondre plusieurs interfaces réseau à un seul et même fichier. Les noms peuvent contenir des métacaractères de type *shell*, comme `en*`. D'autres entrées fourniront diverses règles de correspondance, comme la sélection d'un périphérique réseau par son adresse MAC :

```
[Match]
MACAddress=00:16:3e:8d:2b:5b
```

Les paramètres du périphérique se trouvent dans la section `[Network]` du fichier. Une configuration réseau statique simple ne requiert que les entrées `Address` et `Gateway` :

```
[Match]
MACAddress=00:16:3e:8d:2b:5b

[Network]
Address=192.168.0.100/24
Gateway=192.168.0.1
```

Pour utiliser le protocole DHCP au lieu d'adresses IP statiques, on utilisera plutôt l'entrée `DHCP` :

```
[Match]
MACAddress=00:16:3e:8d:2b:5b

[Network]
DHCP=yes
```

Le service `systemd-networkd` essaiera de récupérer les adresses IPv4 et IPv6 pour l'interface réseau. Pour utiliser uniquement IPv4, il faudra utiliser `DHCP=ipv4`. De même, `DHCP=ipv6` va ignorer les paramètres IPv4 et utiliser uniquement l'adresse IPv6 fournie.

Les réseaux sans fil protégés par mot de passe peuvent également être configurés par `systemd-networkd`, mais l'adaptateur réseau devra être déjà authentifié dans le réseau avant que `systemd-networkd` ne puisse le configurer. L'authentification sera effectuée par *WPA supplicant*, un programme dédié à la configuration des adaptateurs réseau pour les réseaux protégés par mot de passe.

La première étape consiste à créer le fichier d'authentification avec la commande `wpa_passphrase` :

```
# wpa_passphrase MyWifi > /etc/wpa_supplicant/wpa_supplicant-wlo1.conf
```

Cette commande va récupérer le mot de passe du réseau sans fil `MyWifi` depuis l'entrée standard et stocker son hachage dans le fichier `/etc/wpa_supplicant/wpa_supplicant-wlo1.conf`. Notez que le nom du fichier devra contenir le nom approprié de l'interface sans fil, d'où le `wlo1` dans le nom du fichier.

Le gestionnaire `systemd` lit les fichiers de mots de passe WPA dans `/etc/wpa_supplicant/` et crée le service correspondant pour exécuter WPA supplicant et activer l'interface. Le fichier de mots de passe créé dans l'exemple aura alors une unité de service correspondante appelée `wpa_supplicant@wlo1.service`. La commande `systemctl start wpa_supplicant@wlo1.service` va associer l'adaptateur sans fil au point d'accès distant. La commande `systemctl enable wpa_supplicant@wlo1.service` rend l'association automatique au démarrage.

Enfin, un fichier `.network` correspondant à l'interface `wlo1` devra être présent dans `/etc/systemd/network/`, vu que `systemd-networkd` va l'utiliser pour configurer l'interface dès que WPA supplicant aura conclu l'association avec le point d'accès.

Exercices guidés

1. Quelle est la signification du mot `Portal` dans la colonne `CONNECTIVITY` dans l'affichage de la commande `nmcli general status` ?
2. Dans un terminal en ligne de commande, comment un utilisateur normal peut-il utiliser la commande `nmcli` pour se connecter au réseau sans fil `MyWifi` protégé par le mot de passe `MyPassword` ?
3. Quelle commande permet d'activer la carte Wi-Fi si elle a été désactivée par le système d'exploitation ?
4. Dans quel répertoire faudra-t-il ranger les fichiers de configuration personnalisés lorsque `systemd-networkd` gère les interfaces réseau ?

Exercices d'approfondissement

1. Comment un utilisateur peut-il utiliser la commande `nmcli` pour supprimer une connexion inutilisée `Hotel Internet` ?
2. NetworkManager scanne régulièrement les réseaux Wi-Fi. La commande `nmcli device wifi list` n'affiche que les points d'accès trouvés lors du dernier scan. Comment utiliser la commande `nmcli` pour demander à NetworkManager de re-scanner immédiatement tous les points d'accès disponibles ?
3. Quelle entrée `name` devra être utilisée dans la section `[Match]` d'un fichier de configuration de `systemd-networkd` pour correspondre à toutes les interfaces Ethernet ?
4. Comment faut-il exécuter la commande `wpa_passphrase` pour utiliser le mot de passe fourni en argument et non pas depuis l'entrée standard ?

Résumé

Cette leçon aborde les outils couramment utilisés sous Linux pour gérer les connexions réseau hétérogènes et dynamiques. Même si la plupart des méthodes de configuration ne nécessitent pas l'intervention de l'utilisateur, elle est parfois nécessaire et des outils comme *NetworkManager* et *systemd-networkd* peuvent réduire la peine. La leçon aborde les sujets suivants :

- Comment NetworkManager et systemd-networkd s'intègrent au système.
- Comment l'utilisateur peut interagir avec NetworkManager et systemd-networkd.
- Configuration de base de l'interface avec NetworkManager et systemd-networkd.

Voici les commandes et les procédures abordées :

- Les clients en ligne de commande de NetworkManager : `nmtui` et `nmcli`.
- Recherche et connexion aux réseaux Wi-Fi avec les commandes `nmcli` appropriées.
- Connexions persistantes au réseau Wi-Fi avec `systemd-networkd`.

Réponses aux exercices guidés

1. Quelle est la signification du mot `Portal` dans la colonne `CONNECTIVITY` dans l'affichage de la commande `nmcli general status` ?

Il signifie que le processus de connexion nécessite une étape d'authentification supplémentaire (généralement par le biais du navigateur Web).

2. Dans un terminal en ligne de commande, comment un utilisateur normal peut-il utiliser la commande `nmcli` pour se connecter au réseau sans fil `MyWifi` protégé par le mot de passe `MyPassword` ?

Dans un terminal en mode texte, la commande serait

```
$ nmcli device wifi connect MyWifi password MyPassword
```

3. Quelle commande permet d'activer la carte Wi-Fi si elle a été désactivée par le système d'exploitation ?

```
$ nmcli radio wifi on
```

4. Dans quel répertoire faudra-t-il ranger les fichiers de configuration personnalisés lorsque `systemd-networkd` gère les interfaces réseau ?

Dans le répertoire local d'administration réseau : `/etc/systemd/network`.

Réponses aux exercices d'approfondissement

1. Comment un utilisateur peut-il utiliser la commande `nmcli` pour supprimer une connexion inutilisée `Hotel Internet` ?

```
$ nmcli connection delete "Hotel Internet"
```

2. NetworkManager scanne régulièrement les réseaux Wi-Fi. La commande `nmcli device wifi list` n'affiche que les points d'accès trouvés lors du dernier scan. Comment utiliser la commande `nmcli` pour demander à NetworkManager de re-scanner immédiatement tous les points d'accès disponibles ?

L'utilisateur `root` peut lancer la commande `nmcli device wifi rescan` pour demander à NetworkManager de re-scanner les points d'accès disponibles.

3. Quelle entrée `name` devra être utilisée dans la section `[Match]` d'un fichier de configuration de `systemd-networkd` pour correspondre à toutes les interfaces Ethernet ?

L'entrée `name=en*`, étant donné que `en` est le préfixe des interfaces Ethernet sous Linux et que `systemd-networkd` accepte les métacaractères du *shell*.

4. Comment faut-il exécuter la commande `wpa_passphrase` pour utiliser le mot de passe fourni en argument et non pas depuis l'entrée standard ?

Le mot de passe devra être indiqué juste après le SSID, comme dans `wpa_passphrase MyWifi MyPassword`.



109.3 Résolution de problèmes réseaux simples

Référence aux objectifs de LPI

LPIC-1 version 5.0, Exam 102, Objective 109.3

Valeur

4

Domaines de connaissance les plus importants

- Configuration manuelle des interfaces réseau, y compris l’affichage et la modification de la configuration des interfaces réseau à partir de `iproute2`.
- Configuration manuelle du routage, y compris l’affichage et la modification, de même que la configuration de la passerelle par défaut à partir de `iproute2`.
- Résolution des problèmes associés à la configuration réseau.
- Connaissance des commandes historiques `net-tools`.

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

- `ip`
- `hostname`
- `ss`
- `ping`
- `ping6`
- `traceroute`
- `traceroute6`
- `tracethat`
- `tracethat6`

- netcat
- ifconfig
- netstat
- route



109.3 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	109 Principes de base des réseaux
Objectif :	109.3 Dépannage réseau de base
Leçon :	1 sur 2

Introduction

Linux a des fonctionnalités réseau flexibles et performantes. En effet, les systèmes d'exploitation basés sur Linux sont souvent utilisés sur des équipements réseau courants, y compris des systèmes commerciaux coûteux. La gestion de réseau sous Linux pourrait faire l'objet d'une certification à elle toute seule. En gardant cela à l'esprit, cette leçon ne couvrira que quelques outils de configuration et de dépannage de base.

Pensez à bien revoir les leçons sur les protocoles Internet et la configuration persistante du réseau avant d'aller plus loin. Dans cette leçon, nous allons aborder les outils qui permettent de configurer et de dépanner les réseaux IPv4 et IPv6.

Même si l'on sort un peu du cadre de nos objectifs officiels, les *renifleurs de paquets* comme `tcpdump` sont des outils de dépannage fort pratiques. Ils vous permettent de visualiser et d'enregistrer les paquets entrants et sortants d'une interface réseau. Des outils comme les *visualiseurs hexadécimaux* ou les *analyseurs de protocoles* peuvent être utilisés pour visualiser ces paquets avec plus de détails qu'un renifleur de paquets ne le permet généralement. C'est déjà une bonne chose de savoir que ces outils existent.

À propos de la commande `ip`

La commande `ip` est un outil relativement récent utilisé pour afficher et configurer à peu près tout ce qui tourne autour du réseau. Cette leçon aborde quelques-unes des sous-commandes les plus utilisées de `ip`, mais elle ne fait qu'effleurer la surface de ce qui existe. Apprenez à lire la documentation et vous serez beaucoup plus efficace.

Chaque sous-commande de `ip` a sa propre page de manuel. La section `SEE ALSO` du manuel en ligne de `ip` vous affiche la liste :

```
$ man ip
...
SEE ALSO
    ip-address(8), ip-addrlabel(8), ip-l2tp(8), ip-link(8), ip-maddress(8),
    ip-monitor(8), ip-mroute(8), ip-neighbour(8), ip-netns(8), ip-
    ntable(8), ip-route(8), ip-rule(8), ip-tcp_metrics(8), ip-token(8), ip-
    tunnel(8), ip-xfrm(8)
    IP Command reference ip-cref.ps
...
```

Au lieu de consulter cette page à chaque fois que vous avez besoin d'une page de manuel, ajoutez simplement `-` et le nom de la sous-commande à `ip`, par exemple `man ip-route`.

La fonction d'aide est une autre source d'information. Pour afficher l'aide intégrée, ajoutez `help` après la sous-commande :

```
$ ip address help
Usage: ip address {add|change|replace} IFADDR dev IFNAME [ LIFETIME ]
                                     [ CONFFLAG-LIST ]

    ip address del IFADDR dev IFNAME [mngtmpaddr]
    ip address {save|flush} [ dev IFNAME ] [ scope SCOPE-ID ]
                                     [ to PREFIX ] [ FLAG-LIST ] [ label LABEL ] [up]
    ip address [ show [ dev IFNAME ] [ scope SCOPE-ID ] [ master DEVICE ]
                                     [ type TYPE ] [ to PREFIX ] [ FLAG-LIST ]
                                     [ label LABEL ] [up] [ vrf NAME ] ]
    ip address {showdump|restore}
IFADDR := PREFIX | ADDR peer PREFIX
...
```


Aperçu du masque de réseau et du routage

IPv4 et IPv6 sont ce que l'on appelle des protocoles routés ou routables. Cela veut dire qu'ils sont conçus de manière à permettre aux architectes réseau de contrôler le flux des communications. Ethernet n'est pas un protocole routable. Autrement dit, si vous connectez un tas de machines entre elles en utilisant uniquement Ethernet, vous ne pourrez pas faire grand-chose pour contrôler le flux du trafic réseau. Toute tentative de contrôle du trafic finirait par ressembler aux protocoles de routage existants.

Les protocoles routables permettent aux concepteurs de réseaux de segmenter ces derniers dans le but de réduire les besoins de calcul du matériel réseau, d'assurer la redondance et de gérer le trafic.

Les adresses IPv4 et IPv6 comportent deux sections. La première série de bits constitue la partie réseau, tandis que la seconde constitue la partie hôte. Le nombre de bits qui composent la partie réseau est déterminé par le *masque de réseau* (également appelé *masque de sous-réseau*). Dans certains cas, on parle aussi de *longueur du préfixe*. Quel que soit son nom, il s'agit du nombre de bits que la machine considère comme la partie réseau de l'adresse. Dans le cas de l'IPv4, cette longueur est parfois spécifiée en notation décimale à point.

Voici un exemple qui utilise l'IPv4. Vous remarquerez que les chiffres binaires conservent leur valeur dans les octets, même lorsqu'ils sont divisés par le masque de réseau.

```
192.168.130.5/20

      192      168      130      5
      11000000 10101000 10000010 00000101

20 bits = 11111111 11111111 11110000 00000000

Network = 192.168.128.0
Host    = 2.5
```

La partie réseau d'une adresse est utilisée par une machine IPv4 ou IPv6 pour rechercher l'interface sur laquelle un paquet doit être envoyé dans sa table de routage. Lorsqu'un hôte IPv4 ou IPv6 pour lequel le routage est activé reçoit un paquet qui ne lui est pas destiné, il tente de faire correspondre la partie réseau de la destination à un réseau de la table de routage. S'il trouve une entrée correspondante, il envoie le paquet à la destination spécifiée dans la table de routage. S'il ne trouve aucune entrée et qu'une route par défaut est configurée, le paquet est envoyé à la route par défaut. S'il n'y a pas d'entrée trouvée et qu'aucune route par défaut n'est configurée, le paquet est rejeté.

Configurer une interface

Nous allons aborder deux outils que vous pouvez utiliser pour configurer une interface réseau : `ifconfig` et `ip`. Le programme `ifconfig`, bien qu'encore largement utilisé, est considéré comme un outil obsolète et peut ne pas être disponible sur les systèmes les plus récents.

TIP Sur les distributions Linux les plus récentes, l'installation du paquet `net-tools` vous fournira les anciennes commandes de gestion du réseau.

Avant de configurer une interface, vous devez d'abord savoir quelles sont les interfaces disponibles. Il y a plusieurs façons de le faire. L'une d'entre elles consiste à utiliser l'option `-a` de `ifconfig`:

```
$ ifconfig -a
```

Une autre façon de procéder consiste à utiliser `ip`. Vous verrez parfois des exemples avec `ip addr`, `ip a`, et d'autres avec `ip address`. Ils sont tous synonymes. La commande officielle est `ip address`. Cela veut dire que si vous voulez afficher la page de manuel, vous devez utiliser `man ip-address` et non pas `man ip-addr`.

La sous-commande `link` de `ip` affiche les liens d'interface disponibles pour la configuration :

```
$ ip link
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT group
default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP mode DEFAULT
group default qlen 1000
    link/ether 08:00:27:54:18:57 brd ff:ff:ff:ff:ff:ff
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP mode DEFAULT
group default qlen 1000
    link/ether 08:00:27:ab:11:3e brd ff:ff:ff:ff:ff:ff
```

En admettant que le système de fichiers `sys` est monté, vous pouvez également afficher le contenu de `/sys/class/net`:

```
$ ls /sys/class/net
enp0s3  enp0s8  lo
```

Pour configurer une interface avec `ifconfig`, vous devez être connecté en tant que `root` ou

utiliser `sudo` pour exécuter la commande avec les privilèges de root. Inspirez-vous de l'exemple ci-dessous :

```
# ifconfig enp1s0 192.168.50.50/24
```

La version Linux de `ifconfig` est flexible quant à la manière de spécifier le masque de sous-réseau :

```
# ifconfig eth2 192.168.50.50 netmask 255.255.255.0
# ifconfig eth2 192.168.50.50 netmask 0xffffffff00
# ifconfig enp0s8 add 2001:db8::10/64
```

Remarquez qu'avec IPv6, le mot-clé `add` a été utilisé. Si vous ne faites pas précéder une adresse IPv6 par `add`, vous obtiendrez un message d'erreur.

La commande ci-dessous permet de configurer une interface avec `ip` :

```
# ip addr add 192.168.5.5/24 dev enp0s8
# ip addr add 2001:db8::10/64 dev enp0s8
```

Avec `ip`, la même commande est utilisée pour IPv4 et IPv6.

Configurer les options de bas niveau

La commande `ip link` est utilisée pour configurer les paramètres d'interface ou de protocole de bas niveau, comme les VLAN, ARP ou MTU, ou pour désactiver une interface.

Une tâche courante pour `ip link` consiste à désactiver ou activer une interface. Cette opération peut également être effectuée avec `ifconfig` :

```
# ip link set dev enp0s8 down
# ip link show dev enp0s8
3: enp0s8: <BROADCAST,MULTICAST> mtu 1500 qdisc pfifo_fast state DOWN mode DEFAULT group
default qlen 1000
    link/ether 08:00:27:ab:11:3e brd ff:ff:ff:ff:ff:ff
# ifconfig enp0s8 up
# ip link show dev enp0s8
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP mode DEFAULT
group default qlen 1000
```

```
link/ether 08:00:27:ab:11:3e brd ff:ff:ff:ff:ff:ff
```

Il se peut que vous ayez besoin d'ajuster le MTU d'une interface. Comme pour l'activation/désactivation des interfaces, cette opération peut être effectuée avec `ifconfig` ou `ip link`:

```
# ip link set enp0s8 mtu 2000
# ip link show dev enp0s3
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 2000 qdisc pfifo_fast state UP mode DEFAULT
group default qlen 1000
    link/ether 08:00:27:54:53:59 brd ff:ff:ff:ff:ff:ff
# ifconfig enp0s3 mtu 1500
# ip link show dev enp0s3
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP mode DEFAULT
group default qlen 1000
    link/ether 08:00:27:54:53:59 brd ff:ff:ff:ff:ff:ff
```

La table de routage

Les commandes `route`, `netstat -r` et `ip route` peuvent toutes être utilisées pour afficher votre table de routage. Si vous souhaitez modifier vos routes, vous devez utiliser `route` ou `ip route`. Voici quelques exemples d'affichage d'une table de routage :

```
$ netstat -r
Kernel IP routing table
Destination      Gateway          Genmask         Flags   MSS Window  irtt Iface
default          10.0.2.2        0.0.0.0         UG        0  0          0 enp0s3
10.0.2.0         0.0.0.0         255.255.255.0   U        0  0          0 enp0s3
192.168.150.0    0.0.0.0         255.255.255.0   U        0  0          0 enp0s8

$ ip route
default via 10.0.2.2 dev enp0s3 proto dhcp metric 100
10.0.2.0/24 dev enp0s3 proto kernel scope link src 10.0.2.15 metric 100
192.168.150.0/24 dev enp0s8 proto kernel scope link src 192.168.150.200

$ route
Kernel IP routing table
Destination      Gateway          Genmask         Flags Metric Ref    Use Iface
default          10.0.2.2        0.0.0.0         UG    100    0      0 enp0s3
10.0.2.0         0.0.0.0         255.255.255.0   U    100    0      0 enp0s3
192.168.150.0    0.0.0.0         255.255.255.0   U     0     0      0 enp0s8
```

Remarquez qu'il n'y a pas de résultat pour l'IPv6. Si vous souhaitez afficher votre table de routage

pour l'IPv6, vous devez utiliser `route -6`, `netstat -6r` et `ip -6 route`.

```
$ route -6
Kernel IPv6 routing table
```

Destination	Next Hop	Flag	Met	Ref	Use	If
2001:db8::/64	:::	U	256	0	0	enp0s8
fe80::/64	:::	U	100	0	0	enp0s3
2002:a00::/24	:::	!n	1024	0	0	lo
:::/0	2001:db8::1	UG	1	0	0	enp0s8
localhost/128	:::	Un	0	2	84	lo
2001:db8::10/128	:::	Un	0	1	0	lo
fe80::a00:27ff:fe54:5359/128	:::	Un	0	1	0	lo
ff00::/8	:::	U	256	1	3	enp0s3
ff00::/8	:::	U	256	1	6	enp0s8

Un exemple pour `netstat -r6` a été omis étant donné que son résultat est identique à `route -6`. Une partie du résultat de la commande `route` ci-dessus est assez évidente. La colonne **Flag** fournit quelques informations sur la route. Le fanion **U** indique que la route est active. Un **!** signifie que la route est rejetée, c'est-à-dire qu'une route avec un **!** ne sera pas utilisée. Le fanion **n** signifie que la route n'a pas été mise en cache. Le noyau maintient un cache des routes pour une recherche plus rapide, séparément de toutes les routes connues. Le fanion **G** indique une passerelle. La colonne **Metric** ou **Met** n'est pas utilisée par le noyau. Elle fait référence à la distance administrative de la cible. Cette distance administrative est utilisée par les protocoles de routage pour déterminer les routes dynamiques. La colonne **Ref** correspond au nombre de références, ou au nombre d'utilisations d'une route. Comme **Metric**, elle n'est pas utilisée par le noyau Linux. La colonne **Use** montre le nombre de consultations pour une route.

Dans l'affichage de `netstat -r`, **MSS** indique la taille maximale des segments pour les connexions TCP sur cette route. La colonne **Window** indique la taille de la fenêtre TCP par défaut. La colonne **irrtt** indique le temps d'aller-retour des paquets sur cette route.

Le résultat de `ip route` et de `ip -6 route` se présente comme suit :

1. Destination.
2. Adresse optionnelle suivie de l'interface.
3. Protocole de routage utilisé pour ajouter la route.
4. La portée de l'itinéraire. En l'absence d'indication, il s'agit d'une portée globale ou d'une passerelle.
5. La métrique de l'itinéraire. Elle est utilisée par les protocoles de routage dynamique pour déterminer le coût de l'itinéraire. Elle n'est pas utilisée par la plupart des systèmes.

6. S'il s'agit d'un itinéraire IPv6, la préférence d'itinéraire RFC4191.

Quelques exemples devraient permettre d'y voir plus clair :

Exemple IPv4

```
default via 10.0.2.2 dev enp0s3 proto dhcp metric 100
```

1. La destination est la route par défaut.
2. L'adresse de la passerelle est `10.0.2.2` accessible via l'interface `enp0s3`.
3. Elle a été ajoutée à la table de routage par DHCP.
4. Le champ d'application a été omis, il est donc global.
5. La route a un coût de `100`.
6. Pas de préférence de route IPv6.

Exemple IPv6

```
fc0::/64 dev enp0s8 proto kernel metric 256 pref medium
```

1. La destination est `fc0::/64`.
2. Elle est accessible via l'interface `enp0s8`.
3. Elle a été ajoutée automatiquement par le noyau.
4. Le champ d'application a été omis, il est donc global.
5. La route a un coût de `256`.
6. Sa préférence IPv6 est `medium`.

Gérer les routes

Les routes peuvent être gérées en utilisant `route` ou `ip route`. Voici un exemple d'ajout et de suppression d'une route à l'aide de la commande `route`. Avec `route`, vous devez utiliser l'option `-6` pour l'IPv6 :

```
# ping6 -c 2 2001:db8:1::20
connect: Network is unreachable
# route -6 add 2001:db8:1::/64 gw 2001:db8::3
# ping6 -c 2 2001:db8:1::20
PING 2001:db8:1::20(2001:db8:1::20) 56 data bytes
64 bytes from 2001:db8:1::20: icmp_seq=1 ttl=64 time=0.451 ms
```

```
64 bytes from 2001:db8:1::20: icmp_seq=2 ttl=64 time=0.438 ms
# route -6 del 2001:db8:1::/64 gw 2001:db8::3
# ping6 -c 2 2001:db8:1::20
connect: Network is unreachable
```

Voici le même exemple avec la commande `ip route` :

```
# ping6 -c 2 2001:db8:1:20
connect: Network is unreachable
# ip route add 2001:db8:1::/64 via 2001:db8::3
# ping6 -c 2 2001:db8:1:20
PING 2001:db8:1::20(2001:db8:1::20) 56 data bytes
64 bytes from 2001:db8:1::20: icmp_seq=2 ttl=64 time=0.529 ms
64 bytes from 2001:db8:1::20: icmp_seq=2 ttl=64 time=0.438 ms
# ip route del 2001:db8:1::/64 via 2001:db8::3
# ping6 -c 2 2001:db8:1::20
connect: Network is unreachable
```

Exercices guidés

1. Quelles commandes peuvent être utilisées pour afficher la liste des cartes réseau présentes dans le système ?

2. Comment peut-on désactiver temporairement une interface ? Comment la réactiver ?

3. Lequel des éléments ci-dessous vous semble un masque de sous-réseau raisonnable pour IPv4 ?

0.0.0.255	
255.0.255.0	
255.252.0.0	
/24	

4. Quelles commandes pouvez-vous utiliser pour vérifier votre route par défaut ?

5. Comment ajouter une deuxième adresse IP à une interface ?

Exercices d'approfondissement

1. Quelle sous-commande de `ip` peut être utilisée pour configurer le balisage `vlan` ?
2. Comment configurer une route par défaut ?
3. Comment obtenir des informations détaillées sur la commande `ip neighbour` ? Que se passe-t-il si vous l'exécutez seule ?
4. Comment sauvegarder votre table de routage ? Comment la restaurer ?
5. Quelle sous-commande `ip` peut être utilisée pour configurer les options de *spanning tree* ?

Résumé

En règle générale, le réseau est configuré par les scripts de démarrage d'un système ou par un assistant comme NetworkManager. La plupart des distributions disposent d'outils qui éditent les fichiers de configuration des scripts de démarrage pour vous. Consultez la documentation de votre distribution pour en savoir plus.

Savoir configurer manuellement le réseau vous permet de résoudre les problèmes plus efficacement. Cette compétence peut être utile dans les environnements minimalistes, par exemple pour restaurer une sauvegarde ou pour migrer vers un nouveau matériel.

Les outils présentés dans cette section présentent bien plus de fonctionnalités que celles abordées dans cette leçon. Prenez le temps de parcourir la page de manuel de chacun d'entre eux pour vous familiariser avec les options disponibles. Les commandes `ss` et `ip` constituent l'approche moderne des choses, tandis que les autres outils couverts sont considérés comme obsolètes, même s'ils sont encore utilisés au quotidien.

La pratique est le meilleur moyen de se familiariser avec les outils abordés. Sur un ordinateur doté d'une quantité modeste de mémoire vive, il est possible de mettre en place un laboratoire en réseau avec des machines virtuelles sur lesquelles vous pourrez vous entraîner. Trois machines virtuelles suffisent pour se familiariser avec les outils décrits.

Voici les commandes utilisées dans cette leçon :

ifconfig

Commande historique utilisée pour configurer les interfaces réseau et vérifier leur état.

ip

Commande moderne et polyvalente utilisée pour configurer les interfaces réseau et vérifier leur état.

netstat

Commande historique utilisée pour visualiser les connexions réseau actuelles et les informations sur les routes.

route

Commande historique utilisée pour visualiser ou modifier la table de routage d'un système.

Réponses aux exercices guidés

1. Quelles commandes permettent d'afficher la liste des interfaces réseau ?

N'importe laquelle des commandes ci-dessous :

```
ip link, ifconfig -a ou ls /sys/class/net
```

2. Comment peut-on désactiver temporairement une interface ? Comment la réactiver ?

Vous pouvez utiliser `ifconfig` ou `ip link` :

En utilisant `ifconfig`:

```
$ ifconfig wlan1 down
$ ifconfig wlan1 up
```

En utilisant `ip link`:

```
$ ip link set wlan1 down
$ ip link set wlan1 up
```

3. Lequel des éléments ci-dessous vous semble un masque de sous-réseau raisonnable pour IPv4 ?

- 255.252.0.0
- /24

Les autres masques cités ne sont pas valides dans la mesure où ils ne séparent pas proprement l'adresse en deux parties, avec la première partie qui définit le réseau et la seconde l'hôte. Les bits à gauche d'un masque seront toujours 1 et les bits à droite seront toujours 0.

4. Quelles commandes pouvez-vous utiliser pour vérifier votre route par défaut ?

Vous pouvez utiliser `route`, `netstat -r` ou `ip route` :

```
$ route
Kernel IP routing table
Destination    Gateway      Genmask      Flags Metric Ref    Use Iface
default        server       0.0.0.0      UG    600    0      0 wlan1
192.168.1.0    0.0.0.0     255.255.255.0 U     600    0      0 wlan1
```

```
$ netstat -r
Kernel IP routing table
Destination      Gateway          Genmask          Flags   MSS Window  irtt Iface
default          server          0.0.0.0          UG      0 0        0 wlan1
192.168.1.0      0.0.0.0         255.255.255.0    U       0 0        0 wlan1
$ ip route
default via 192.168.1.20 dev wlan1 proto static metric 600
192.168.1.0/24 dev wlan1 proto kernel scope link src 192.168.1.24 metric 600
```

5. Comment ajouter une deuxième adresse IP à une interface ?

Vous pouvez utiliser `ip address` ou `ifconfig`. Gardez à l'esprit que `ifconfig` est un outil obsolète :

```
$ ip addr add 172.16.15.16/16 dev enp0s9 label enp0s9:sub1
```

La partie de la commande `label enp0s9:sub1` ajoute un alias à `enp0s9`. Si vous n'utilisez pas l'ancienne commande `ifconfig`, vous pouvez l'omettre. Dans le cas contraire, la commande fonctionnera toujours, mais l'adresse que vous venez d'ajouter n'apparaîtra pas dans le résultat de `ifconfig`.

Vous pouvez également utiliser `ifconfig` :

```
$ ifconfig enp0s9:sub1 172.16.15.16/16
```

Réponses aux exercices d'approfondissement

1. Quelle sous-commande de `ip` peut être utilisée pour configurer le balisage vlan ?

La sous-commande `ip link` possède une option `vlan` qui peut être utilisée. Voici un exemple de balisage d'une sous-interface avec le vlan 20.

```
# ip link add link enp0s9 name enp0s9.20 type vlan id 20
```

2. Comment configurer une route par défaut ?

En utilisant `route` ou `ip route` :

```
# route add default gw 192.168.1.1
# ip route add default via 192.168.1.1
```

3. Comment obtenir des informations détaillées sur la commande `ip neighbour` ? Que se passe-t-il si vous l'exécutez seule ?

En lisant la page de manuel :

```
$ man ip-neighbour
```

Elle affiche votre cache ARP :

```
$ ip neighbour
10.0.2.2 dev enp0s3 lladdr 52:54:00:12:35:02 REACHABLE
```

4. Comment sauvegarder votre table de routage ? Comment la restaurer ?

L'exemple ci-dessous illustre la sauvegarde et la restauration d'une table de routage :

```
# ip route save > /root/routes/route_backup
# ip route restore < /root/routes/route_backup
```

5. Quelle sous-commande `ip` peut être utilisée pour configurer les options de *spanning tree* ?

Comme pour la gestion des paramètres vlan, la sous-commande `ip link` permet de configurer le *spanning tree* en utilisant le type `bridge`. L'exemple montre l'ajout d'une interface virtuelle

avec une priorité STP de 50 :

```
# ip link add link enp0s9 name enp0s9.50 type bridge priority 50
```



109.3 Leçon 2

Certification :	LPIC-1
Version :	5.0
Thème :	109 Principes de base des réseaux
Objectif :	109.3 Dépannage réseau de base
Leçon :	2 sur 2

Introduction

Les systèmes d'exploitation Linux disposent d'une grande panoplie d'outils pour résoudre les problèmes liés au réseau. Cette leçon va couvrir certains des outils les plus courants. À ce stade, vous devriez avoir une bonne connaissance du modèle OSI et d'autres modèles de réseau, de l'adressage IPv4 et IPv6, ainsi que des principes de base du routage et de la commutation.

Le meilleur moyen de tester une connexion réseau consiste à simplement utiliser votre application. Si ça ne marche pas, vous avez toute une série d'outils à disposition pour diagnostiquer le problème.

Tester les connexions avec ping

Les commandes `ping` et `ping6` peuvent être utilisées pour envoyer une requête écho ICMP (*ICMP echo*) à une adresse IPv4 ou IPv6, respectivement. Une requête écho ICMP envoie une petite quantité de données à l'adresse de destination. Si celle-ci est joignable, elle renvoie à l'expéditeur un message de réponse écho ICMP (*ICMP echo reply*) avec les mêmes données que celles qui lui ont été envoyées :

```
$ ping -c 3 192.168.50.2
PING 192.168.50.2 (192.168.50.2) 56(84) bytes of data.
64 bytes from 192.168.50.2: icmp_seq=1 ttl=64 time=0.525 ms
64 bytes from 192.168.50.2: icmp_seq=2 ttl=64 time=0.419 ms
64 bytes from 192.168.50.2: icmp_seq=3 ttl=64 time=0.449 ms

--- 192.168.50.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2006ms
rtt min/avg/max/mdev = 0.419/0.464/0.525/0.047 ms
```

```
$ ping6 -c 3 2001:db8::10
PING 2001:db8::10(2001:db8::10) 56 data bytes
64 bytes from 2001:db8::10: icmp_seq=1 ttl=64 time=0.425 ms
64 bytes from 2001:db8::10: icmp_seq=2 ttl=64 time=0.480 ms
64 bytes from 2001:db8::10: icmp_seq=3 ttl=64 time=0.725 ms

--- 2001:db8::10 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2000ms
rtt min/avg/max/mdev = 0.425/0.543/0.725/0.131 ms
```

L'option `-c` est utilisée pour spécifier le nombre de paquets à envoyer. Sans cette option, `ping` et `ping6` vont continuer à envoyer des paquets jusqu'à ce que vous les arrêtez, typiquement avec la combinaison de touches `Ctrl` + `C`.

Si vous ne pouvez pas envoyer de ping à un hôte, cela ne veut pas dire pour autant que vous ne pouvez pas vous y connecter. Bon nombre d'entreprises disposent de pare-feu ou de règles de contrôle d'accès aux routeurs qui bloquent tout ce qui n'est pas strictement nécessaire au fonctionnement de leurs systèmes. Cela inclut les requêtes les réponses écho ICMP. Ces paquets peuvent contenir des données arbitraires et un hacker ingénieux pourrait les utiliser pour exfiltrer des données.

Retracer les itinéraires

Les outils `traceroute` et `traceroute6` peuvent être utilisés pour afficher le parcours d'un paquet jusqu'à sa destination. Pour ce faire, ils envoient plusieurs paquets à la destination, en incrémentant le champ *temps de vie* (*Time-To-Live* - TTL) de l'en-tête IP pour chaque paquet subséquent. Chacun des routeurs sur cet itinéraire répondra par un message ICMP de dépassement du TTL :

```
$ traceroute 192.168.1.20
```



```

traceroute to 192.168.1.20 (192.168.1.20), 30 hops max, 60 byte packets
 1  10.0.2.2 (10.0.2.2)  0.396 ms  0.171 ms  0.132 ms
 2  192.168.1.20 (192.168.1.20)  2.665 ms  2.573 ms  2.573 ms
$ traceroute 192.168.50.2
traceroute to 192.168.50.2 (192.168.50.2), 30 hops max, 60 byte packets
 1  192.168.50.2 (192.168.50.2)  0.433 ms  0.273 ms  0.171 ms
$ traceroute6 2001:db8::11
traceroute to 2001:db8::11 (2001:db8::11), 30 hops max, 80 byte packets
 1  2001:db8::11 (2001:db8::11)  0.716 ms  0.550 ms  0.641 ms
$ traceroute 2001:db8::11
traceroute to 2001:db8::11 (2001:db8::11), 30 hops max, 80 byte packets
 1  2001:db8::10 (2001:db8::11)  0.617 ms  0.461 ms  0.387 ms
$ traceroute net2.example.net
traceroute to net2.example.net (192.168.50.2), 30 hops max, 60 byte packets
 1  net2.example.net (192.168.50.2)  0.533 ms  0.529 ms  0.504 ms
$ traceroute6 net2.example.net
traceroute to net2.example.net (2001:db8::11), 30 hops max, 80 byte packets
 1  net2.example.net (2001:db8::11)  0.738 ms  0.607 ms  0.304 ms

```

Par défaut, `traceroute` envoie 3 paquets UDP avec des données hétéroclites au port 33434, en l'incrémentant à chaque fois qu'il envoie un paquet. Chaque ligne du résultat de la commande est une interface de routeur traversée par le paquet. Le temps indiqué à chaque ligne du résultat correspond à la durée de l'aller-retour pour chaque paquet. L'adresse IP est l'adresse de l'interface du routeur en question. Si `traceroute` le peut, il utilise le nom DNS de l'interface du routeur. Parfois, vous verrez `*` à la place de la durée. Cela signifie que `traceroute` n'a jamais reçu le message de TTL dépassé pour ce paquet. Lorsque vous commencez à voir ce genre de message, cela signifie souvent que la dernière réponse correspond au dernier saut sur l'itinéraire.

Si vous avez un accès root, l'option `-I` va configurer `traceroute` pour qu'il utilise des requêtes écho ICMP au lieu de paquets UDP. C'est souvent plus efficace qu'UDP car l'hôte de destination est plus susceptible de répondre à une requête écho ICMP qu'à un paquet UDP :

```

# traceroute -I learning.lpi.org
traceroute to learning.lpi.org (208.94.166.201), 30 hops max, 60 byte packets
 1  047-132-144-001.res.spectrum.com (47.132.144.1)  9.764 ms  9.702 ms  9.693 ms
 2  096-034-094-106.biz.spectrum.com (96.34.94.106)  8.389 ms  8.481 ms  8.480 ms
 3  dtr01hlgnc-gbe-4-15.hlg.nc.charter.com (96.34.64.172)  8.763 ms  8.775 ms  8.770 ms
 4  acr01mgtnnc-vln-492.mgtn.nc.charter.com (96.34.67.202)  27.080 ms  27.154 ms  27.151 ms
 5  bbr01gnvlsc-bue-3.gnvl.sc.charter.com (96.34.2.112)  31.339 ms  31.398 ms  31.395 ms
 6  bbr01aldlmi-tge-0-0-0-13.aldl.mi.charter.com (96.34.0.161)  39.092 ms  38.794 ms  38.821 ms
ms
 7  prr01ashbva-bue-3.ashb.va.charter.com (96.34.3.51)  34.208 ms  36.474 ms  36.544 ms

```

```

8  bx2-ashburn.bell.ca (206.126.236.203)  53.973 ms  35.975 ms  38.250 ms
9  tcore4-ashburnbk_0-12-0-0.net.bell.ca (64.230.125.190)  66.315 ms  65.319 ms  65.345 ms
10 tcore4-toronto47_2-8-0-3.net.bell.ca (64.230.51.22)  67.427 ms  67.502 ms  67.498 ms
11 agg1-toronto47_xe-7-0-0_core.net.bell.ca (64.230.161.114)  61.270 ms  61.299 ms  61.291
ms
12 dis4-clarkson16_5-0.net.bell.ca (64.230.131.98)  61.101 ms  61.177 ms  61.168 ms
13 207.35.12.142 (207.35.12.142)  70.009 ms  70.069 ms  59.893 ms
14 unassigned-117.001.centrilogic.com (66.135.117.1)  61.778 ms  61.950 ms  63.041 ms
15 unassigned-116.122.akn.ca (66.135.116.122)  62.702 ms  62.759 ms  62.755 ms
16 208.94.166.201 (208.94.166.201)  62.936 ms  62.932 ms  62.921 ms

```

Certaines entreprises bloquent les requêtes écho ICMP et les réponses. Pour contourner ce problème, vous pouvez utiliser TCP. En utilisant un port TCP ouvert connu, vous pouvez garantir que l'hôte de destination va répondre. Pour utiliser TCP, invoquez l'option `-T` avec `-p` pour spécifier le port. Comme pour les requêtes écho ICMP, vous devez être `root` pour le faire :

```

# traceroute -m 60 -T -p 80 learning.lpi.org
traceroute to learning.lpi.org (208.94.166.201), 60 hops max, 60 byte packets
1  * * *
2  096-034-094-106.biz.spectrum.com (96.34.94.106)  12.178 ms  12.229 ms  12.175 ms
3  dtr01hlrgnc-gbe-4-15.hlrg.nc.charter.com (96.34.64.172)  12.134 ms  12.093 ms  12.062 ms
4  acr01mgtnnc-vln-492.mgtn.nc.charter.com (96.34.67.202)  31.146 ms  31.192 ms  31.828 ms
5  bbr01gnvlsc-bue-3.gnvl.sc.charter.com (96.34.2.112)  39.057 ms  46.706 ms  39.745 ms
6  bbr01aldlmi-tge-0-0-0-13.aldl.mi.charter.com (96.34.0.161)  50.590 ms  58.852 ms  58.841
ms
7  prr01ashbva-bue-3.ashb.va.charter.com (96.34.3.51)  34.556 ms  37.892 ms  38.274 ms
8  bx2-ashburn.bell.ca (206.126.236.203)  38.249 ms  36.991 ms  36.270 ms
9  tcore4-ashburnbk_0-12-0-0.net.bell.ca (64.230.125.190)  66.779 ms  63.218 ms tcore3-
ashburnbk_100ge0-12-0-0.net.bell.ca (64.230.125.188)  60.441 ms
10 tcore4-toronto47_2-8-0-3.net.bell.ca (64.230.51.22)  63.932 ms  63.733 ms  68.847 ms
11 agg2-toronto47_xe-7-0-0_core.net.bell.ca (64.230.161.118)  60.144 ms  60.443 ms agg1-
toronto47_xe-7-0-0_core.net.bell.ca (64.230.161.114)  60.851 ms
12 dis4-clarkson16_5-0.net.bell.ca (64.230.131.98)  67.246 ms dis4-clarkson16_7-
0.net.bell.ca (64.230.131.102)  68.404 ms dis4-clarkson16_5-0.net.bell.ca (64.230.131.98)
67.403 ms
13 207.35.12.142 (207.35.12.142)  66.138 ms  60.608 ms  64.656 ms
14 unassigned-117.001.centrilogic.com (66.135.117.1)  70.690 ms  62.190 ms  61.787 ms
15 unassigned-116.122.akn.ca (66.135.116.122)  62.692 ms  69.470 ms  68.815 ms
16 208.94.166.201 (208.94.166.201)  61.433 ms  65.421 ms  65.247 ms
17 208.94.166.201 (208.94.166.201)  64.023 ms  62.181 ms  61.899 ms

```

Comme `ping`, `traceroute` a ses limites. Il se peut que les pare-feu et les routeurs bloquent les

paquets envoyé par ou renvoyé à `traceroute`. Si vous avez un accès `root`, certaines options pourront vous aider à obtenir des résultats précis.

Trouver les MTU avec `tracepath`

La commande `tracepath` ressemble à `traceroute`. Elle traque les tailles des *Maximum Transmission Unit* (MTU) le long du chemin. Le MTU est soit un paramètre configuré sur une interface réseau, soit une limitation matérielle de la plus grande unité de données du protocole qu'elle peut transmettre ou recevoir. Le programme `tracepath` fonctionne de la même manière que `traceroute` dans la mesure où il incrémente le TTL à chaque paquet. La différence, c'est qu'il envoie un très gros datagramme UDP. Ce datagramme sera forcément plus grand que le périphérique avec le plus petit MTU le long de la route. Lorsque le paquet atteint ce périphérique, il répond généralement par un paquet de destination inaccessible. Le paquet ICMP destination inaccessible contient un champ pour le MTU du lien sur lequel il enverrait le paquet s'il en était capable. `tracepath` envoie alors tous les paquets suivants avec cette taille :

```
$ tracepath 192.168.1.20
1?: [LOCALHOST]                pmtu 1500
1:  10.0.2.2                    0.321ms
1:  10.0.2.2                    0.110ms
2:  192.168.1.20                2.714ms reached
Resume: pmtu 1500 hops 2 back 64
```

Contrairement à `traceroute`, vous devez utiliser explicitement `tracepath6` pour l'IPv6 :

```
$ tracepath 2001:db8::11
tracepath: 2001:db8::11: Address family for hostname not supported
$ tracepath6 2001:db8::11
1?: [LOCALHOST]                0.027ms pmtu 1500
1:  net2.example.net            0.917ms reached
1:  net2.example.net            0.527ms reached
Resume: pmtu 1500 hops 1 back 1
```

Le résultat ressemble à celui de `traceroute`. L'avantage de `tracepath` est qu'il affiche le plus petit MTU sur l'ensemble du lien à la dernière ligne. Cela peut être pratique pour diagnostiquer les connexions incompatibles avec les fragments.

Comme pour les outils de diagnostic précédents, il se peut que certains équipements bloquent vos paquets.

Créer des connexions arbitraires

Le programme `nc`, connu sous le nom de netcat, peut envoyer ou recevoir des données arbitraires sur une connexion réseau TCP ou UDP. Les exemples ci-dessous devraient permettre de clarifier son fonctionnement.

Voici un exemple de mise en place d'un client d'écoute (*listener*) sur le port 1234 :

```
$ nc -l 1234
LPI Example
```

Le résultat `LPI Example` apparaît à la suite de l'exemple ci-dessous qui configure un expéditeur netcat pour envoyer des paquets à `net2.example.net` sur le port 1234. L'option `-l` est utilisée pour spécifier que vous souhaitez que `nc` reçoive des données au lieu d'en envoyer :

```
$ nc net2.example.net 1234
LPI Example
```

Utilisez `Ctrl` + `C` sur l'un ou l'autre système pour interrompre la connexion.

Netcat fonctionne avec les adresses IPv4 et IPv6. Il est compatible avec les protocoles TCP et UDP. On peut même l'utiliser pour mettre en place un shell distant rudimentaire.

WARNING

Notez que toutes les installations de `nc` ne supportent pas l'option `-e`. Consultez les pages de manuel de votre installation pour en savoir plus sur les considérations de sécurité de cette option ainsi que sur les autres méthodes qui permettent d'exécuter des commandes sur un système distant.

```
$ hostname
net2
$ nc -u -e /bin/bash -l 1234
```

L'option `-u` est pour UDP. L'option `-e` demande à netcat d'envoyer tout ce qu'il reçoit vers l'entrée standard de l'exécutable en argument. En l'occurrence, `/bin/bash`.

```
$ hostname
net1
$ nc -u net2.example.net 1234
hostname
net2
```

```
pwd
/home/emma
```

Vous avez remarqué que le résultat de la commande `hostname` correspond à celui de l'hôte en écoute et que la commande `pwd` affiche un répertoire ?

Afficher les connexions en cours et les clients d'écoute

Les outils `netstat` et `ss` peuvent être utilisés pour afficher le statut de vos clients d'écoute et de vos connexions. Comme `ifconfig`, `netstat` est un outil obsolète. `netstat` et `ss` ont des résultats et des options similaires. Voici quelques options disponibles pour les deux programmes :

-a

Affiche toutes les sockets.

-l

Affiche les sockets en écoute.

-p

Affiche le processus associé à la connexion.

-n

Empêche la résolution de noms pour les ports et les adresses.

-t

Affiche les connexions TCP.

-u

Affiche les connexions UDP.

Les exemples ci-dessous montrent le résultat d'un ensemble d'options couramment utilisées pour les deux programmes :

```
# netstat -tulnp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program
name
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      892/sshd
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN      1141/master
tcp6       0      0 :::22                  :::*                    LISTEN      892/sshd
tcp6       0      0 :::1:25                 :::*                    LISTEN      1141/master
```

```

udp      0      0 0.0.0.0:68          0.0.0.0:*          692/dhclient
# ss -tulnp
# ss -tulnp
Netid  State      Recv-Q  Send-Q      Local Address:Port      Peer
Address:Port
udp    UNCONN     0        0              :68                *
users: (("dhclient",pid=693,fd=6))
tcp    LISTEN     0       128              :22                *
users: (("sshd",pid=892,fd=3))
tcp    LISTEN     0       100      127.0.0.1:25        :
users: (("master",pid=1099,fd=13))
tcp    LISTEN     0       128      [::]:22             [::]:*
users: (("sshd",pid=892,fd=4))
tcp    LISTEN     0       100      [::1]:25            [::]:*
users: (("master",pid=1099,fd=14))

```

La colonne **Recv-Q** correspond au nombre de paquets qu'une socket a reçus mais qu'elle n'a pas transmis à son programme. La colonne **Send-Q** indique le nombre de paquets qu'une socket a envoyés et qui n'ont pas été confirmés par le destinataire. Les autres colonnes sont facilement compréhensibles.

Exercices guidés

1. Quelle(s) commande(s) utiliseriez-vous pour envoyer un écho ICMP à `learning.lpi.org` ?

2. Comment pouvez-vous déterminer la route vers `8.8.8.8` ?

3. Quelle commande vous montrerait si des processus écoutent sur le port TCP 80 ?

4. Comment trouver quel processus est à l'écoute sur un port ?

5. Comment pouvez-vous déterminer le MTU maximum d'un chemin du réseau ?

Exercices d'approfondissement

1. Comment utiliser netcat pour envoyer une requête HTTP à un serveur web ?

2. Quelles sont les raisons pour lesquelles le ping vers un hôte peut échouer ?

3. Nommez un outil que vous pourriez utiliser pour voir les paquets réseau qui atteignent ou quittent un hôte Linux.

4. Comment forcer traceroute à utiliser une interface différente ?

5. Est-ce que traceroute est capable d'afficher les MTUs ?

Résumé

La configuration du réseau est généralement effectuée par les scripts de démarrage du système ou par un assistant comme NetworkManager. La plupart des distributions disposent d'outils qui se chargent d'éditer les fichiers de configuration des scripts de démarrage pour vous. Consultez la documentation de votre distribution pour en savoir plus.

La capacité à configurer manuellement le réseau vous permet de déboguer plus efficacement. Elle est utile dans les environnements minimalistes, par exemple pour restaurer des sauvegardes ou pour migrer vers un nouveau matériel.

Les outils décrits dans cette section ont plus de fonctionnalités que celles abordées au fil de cette leçon. Il serait intéressant de parcourir la page de manuel de chacun d'entre eux pour vous familiariser avec les options disponibles. Les commandes `ss` et `ip` incarnent la façon moderne de faire les choses, tandis que les autres, bien que toujours employés, sont considérés comme obsolètes.

La meilleure façon de se familiariser avec les outils abordés est la pratique. Avec un ordinateur doté d'une quantité modeste de RAM, vous pouvez mettre en place un laboratoire de réseau virtuel avec des machines virtuelles pour vous entraîner. Trois machines virtuelles suffisent pour vous familiariser avec les outils en question.

Voici les commandes utilisées dans cette leçon :

ping et ping6

Utilisés pour transmettre des paquets ICMP à un hôte distant afin de tester la disponibilité d'une connexion réseau.

traceroute et traceroute6

Utilisés pour tracer un chemin à travers un réseau afin d'en déterminer la connectivité.

tracpath et tracpath6

Utilisés pour tracer un chemin à travers un réseau et pour déterminer la taille des MTU le long d'un itinéraire.

nc

Utilisé pour établir des connexions arbitraires sur un réseau afin de tester la connectivité, ainsi que pour interroger un réseau sur les services et les périphériques disponibles.

netstat

Commande obsolète utilisée pour déterminer les connexions réseau ouvertes et les statistiques

d'un système.

SS

Commande moderne utilisée pour déterminer les connexions réseau ouvertes et les statistiques d'un système.

Réponses aux exercices guidés

1. Quelle(s) commande(s) utiliseriez-vous pour envoyer un écho ICMP à `learning.lpi.org` ?

Vous utiliseriez `ping` ou `ping6` :

```
$ ping learning.lpi.org
```

ou

```
$ ping6 learning.lpi.org
```

2. Comment pouvez-vous déterminer la route vers `8.8.8.8` ?

En utilisant les commandes `tracpath` ou `traceroute`.

```
$ tracpath 8.8.8.8
```

ou

```
$ traceroute 8.8.8.8
```

3. Quelle commande vous montrerait si des processus écoutent sur le port TCP 80 ?

Avec `ss` :

```
$ ss -ln | grep ":80"
```

Avec `netstat` :

```
$ netstat -ln | grep ":80"
```

Même s'il ne s'agit pas d'une exigence de l'examen, vous pouvez également utiliser `lsof` :

```
# lsof -Pi:80
```

4. Comment trouver quel processus est à l'écoute sur un port ?

Encore une fois, il y a plusieurs façons de faire cela. Vous pouvez utiliser `lsof` de la même manière que la réponse ci-dessus, en remplaçant le numéro de port. Vous pouvez aussi utiliser `netstat` ou `ss` avec l'option `-p`. Rappelez-vous que `netstat` est considéré comme un outil obsolète.

```
# netstat -lnp | grep ":22"
```

Les mêmes options qui fonctionnent avec `netstat` fonctionnent également avec `ss` :

```
# ss -lnp | grep ":22"
```

5. Comment pouvez-vous déterminer le MTU maximum d'un chemin du réseau ?

En utilisant la commande `tracert` :

```
$ tracert somehost.example.com
```

Réponses aux exercices d'approfondissement

1. Comment utiliser netcat pour envoyer une requête HTTP à un serveur web ?

En saisissant la ligne de requête HTTP, les en-têtes éventuels et une ligne vide dans le terminal :

```
$ nc learning.lpi.org 80
GET /index.html HTTP/1.1
HOST: learning.lpi.org

HTTP/1.1 302 Found
Location: https://learning.lpi.org:443/index.html
Date: Wed, 27 May 2020 22:54:46 GMT
Content-Length: 5
Content-Type: text/plain; charset=utf-8

Found
```

2. Quelles sont les raisons pour lesquelles le ping vers un hôte peut échouer ?

Il y a un certain nombre de raisons possibles. En voici quelques-unes :

- L'hôte distant est hors service.
- Une liste d'accès du routeur bloque votre ping.
- Le pare-feu de l'hôte distant bloque votre ping.
- Vous utilisez peut-être un nom d'hôte ou une adresse incorrecte.
- Votre résolution de noms renvoie une adresse incorrecte.
- La configuration réseau de votre machine est incorrecte.
- Le pare-feu de votre machine bloque le ping.
- La configuration réseau de l'hôte distant est incorrecte.
- Les interfaces de votre machine sont déconnectées.
- L'interface de la machine distante est déconnectée.
- Un composant réseau comme un switch, un câble ou un routeur entre votre machine et celle de l'hôte distant ne fonctionne plus.

3. Nommez un outil que vous pourriez utiliser pour voir les paquets réseau qui atteignent ou quittent un hôte Linux.

Les deux outils `tcpdump` et `wireshark` peuvent être utilisés.

4. Comment forcer `traceroute` à utiliser une interface différente ?

En utilisant l'option `-i` :

```
$ traceroute -i eth2 learning.lpi.org
traceroute -i eth2 learning.lpi.org
traceroute to learning.lpi.org (208.94.166.201), 30 hops max, 60 byte packets
...
```

5. Est-ce que `traceroute` est capable d'afficher les MTUs ?

Oui, avec l'option `--mtu` :

```
# traceroute -I --mtu learning.lpi.org
traceroute to learning.lpi.org (208.94.166.201), 30 hops max, 65000 byte packets
 1  047-132-144-001.res.spectrum.com (47.132.144.1)  9.974 ms F=1500  10.476 ms  4.743 ms
 2  096-034-094-106.biz.spectrum.com (96.34.94.106)  8.697 ms  9.963 ms  10.321 ms
...
```



109.4 Configuration de la résolution de noms

Référence aux objectifs de LPI

LPIC-1 version 5.0, Exam 102, Objective 109.4

Valeur

2

Domaines de connaissance les plus importants

- Requêtes sur serveurs DNS distants.
- Configuration de la résolution de nom locale et utilisation de serveurs DNS distants.
- Modification de l'ordre de la résolution des noms.
- Résolution d'erreurs reliées avec la résolution de nom.
- Connaissance de base de `systemd-resolved`

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

- `/etc/hosts`
- `/etc/resolv.conf`
- `/etc/nsswitch.conf`
- `host`
- `dig`
- `getent`



109.4 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	109 Principes de base des réseaux
Objectif :	109.4 Configurer les DNS côté client
Leçon :	1 sur 1

Introduction

Cette leçon aborde la configuration DNS côté client et l'utilisation de quelques outils de résolution de noms en ligne de commande.

On ne peut pas mémoriser et conserver les adresses IP, les UID, les GID et d'autres données numériques pour tout. Les services de résolution de noms permettent de traduire des noms simples à retenir en nombres et vice versa. Cette leçon se concentre sur la résolution des noms d'hôtes, mais un processus similaire s'applique aux noms d'utilisateurs, aux noms de groupes, aux numéros de ports et bien d'autres choses encore.

Le processus de résolution des noms

Les programmes qui transforment les noms en valeurs numériques utilisent presque toujours des fonctions fournies par la bibliothèque C standard, la GNU glibc sur les systèmes Linux. La première chose que font ces fonctions consiste à lire le fichier `/etc/nsswitch.conf` pour y trouver des instructions sur la manière de résoudre ce type de nom. Cette leçon se concentre sur la résolution des noms d'hôtes, mais le même processus s'applique à d'autres types de résolution de noms. Une fois que le processus a lu `/etc/nsswitch.conf`, il recherche le nom de la manière

spécifiée. Puisque `/etc/nsswitch.conf` supporte les plugins, ce qui vient ensuite peut être tout et n'importe quoi. Une fois que la fonction a fini de rechercher le nom ou la valeur, elle renvoie le résultat au processus qui l'a invoquée.

Les classes DNS

Le DNS comporte trois classes d'enregistrements : IN, HS et CH. Dans cette leçon, toutes les requêtes DNS seront de type IN. La classe IN correspond aux adresses Internet qui utilisent la pile TCP/IP. CH est l'abréviation de ChaosNet, une technologie réseau éphémère et qui n'est plus utilisée. La classe HS correspond à Hesiod. Hesiod est un moyen de stocker des éléments comme les mots de passe et les entrées de groupe dans le DNS. Hesiod dépasse le cadre de cette leçon.

Comprendre `/etc/nsswitch.conf`

Le meilleur moyen d'en savoir plus sur ce fichier est de consulter la page de manuel. Elle est disponible sur la plupart des systèmes Linux. On peut y accéder avec la commande `man nsswitch.conf`. Elle peut également être trouvée à l'adresse suivante : <https://man7.org/linux/man-pages/man5/nsswitch.conf.5.html>.

Voici un exemple simple de `/etc/nsswitch.conf` tiré de la page de manuel :

```
passwd:          compat
group:           compat
shadow:          compat

hosts:           dns [!UNAVAIL=return] files
networks:        nis [NOTFOUND=return] files
ethers:          nis [NOTFOUND=return] files
protocols:       nis [NOTFOUND=return] files
rpc:             nis [NOTFOUND=return] files
services:        nis [NOTFOUND=return] files
# This is a comment. It is ignored by the resolution functions.
```

Le fichier est organisé en colonnes. La colonne la plus à gauche correspond au type de base de données de noms. Les autres colonnes sont les méthodes que les fonctions de résolution doivent utiliser pour rechercher un nom. Les méthodes sont suivies par les fonctions de gauche à droite. Les colonnes avec `[]` sont utilisées pour fournir une logique conditionnelle limitée à la colonne immédiatement à gauche.

Admettons qu'un processus essaie de résoudre le nom d'hôte `learning.lpi.org`. Il fera un appel à la bibliothèque C appropriée (probablement `gethostbyname`). Cette fonction va donc lire

`/etc/nsswitch.conf`. Puisque le processus recherche un nom d'hôte, il trouvera la ligne commençant par `hosts`. Il essaiera alors d'utiliser le DNS pour résoudre le nom. La colonne suivante, `!UNAVAIL=return` signifie que si le service n'est *pas* indisponible, alors il ne faut pas essayer la source suivante, c'est-à-dire que si le DNS est disponible, il faut arrêter d'essayer de résoudre le nom d'hôte même si les serveurs de noms n'y parviennent pas. Si le DNS n'est pas disponible, il faut passer à la source suivante. Dans ce cas, la source suivante est `files`.

Une colonne au format `[result=action]` signifie que lorsqu'une recherche dans le résolveur de la colonne de gauche est `result`, alors `action` est exécutée. Si `result` est précédé d'un `!`, cela signifie que si le résultat n'est pas `result`, alors on effectue `action`. Pour une description des résultats et des actions possibles, reportez-vous à la page de manuel.

Supposons maintenant qu'un processus essaie de résoudre un numéro de port en fonction d'un nom de service. Il va lire la ligne `services`. La première source répertoriée est `NIS`. `NIS` est l'acronyme de *Network Information Service* (également connu sous le nom de *Yellow Pages* ou Pages Jaunes). Il s'agit d'un ancien service qui permettait une gestion centralisée de choses comme les utilisateurs. On ne l'utilise plus que rarement en raison de sa sécurité insuffisante. La colonne suivante `[NOTFOUND=return]` signifie que si la recherche a réussi mais que le service n'a pas été trouvé, il faut arrêter la recherche. Si la condition ci-dessus ne s'applique pas, il faut utiliser les fichiers locaux.

Tout ce qui se trouve à droite de `#` est un commentaire et sera ignoré.

Le fichier `/etc/resolv.conf`

Le fichier `/etc/resolv.conf` est utilisé pour configurer la résolution d'hôtes via le DNS. Certaines distributions comportent des scripts de démarrage, des démons et autres outils qui peuvent écrire dans ce fichier. Gardez cela à l'esprit lorsque vous le modifiez à la main. Vérifiez votre distribution et la documentation de ses outils de configuration réseau si c'est le cas. Certains outils comme Network Manager ajoutent un commentaire au fichier pour vous mettre en garde que les modifications manuelles seront écrasées.

Tout comme `/etc/nsswitch.conf`, il y a une page de manuel associée à ce fichier. Vous pouvez y accéder avec la commande `man resolv.conf` ou à l'adresse <https://man7.org/linux/man-pages/man5/resolv.conf.5.html>.

Le format de ce fichier est assez simple. Dans la colonne de gauche, vous avez l'option `name`. Les autres colonnes sur la même ligne correspondent à la valeur de l'option.

L'option la plus courante est `nameserver`. Elle est utilisée pour spécifier l'adresse IPv4 ou IPv6 d'un serveur DNS. Au moment de la rédaction de ces lignes, vous pouvez spécifier jusqu'à trois serveurs de noms. Si votre `/etc/resolv.conf` n'a pas d'option `nameserver`, votre système

utilisera le serveur DNS de la machine locale par défaut.

Voici un exemple simple de fichier qui correspond aux configurations les plus courantes :

```
search lpi.org
nameserver 10.0.0.53
nameserver fd00:ffff::2:53
```

L'option `search` est utilisée pour permettre des recherches abrégées. Dans l'exemple, un seul domaine de recherche `lpi.org` est configuré. Cela signifie que toute tentative de résolution d'un nom d'hôte sans la partie domaine aura `.lpi.org` ajouté avant la recherche. Par exemple, si vous essayez de rechercher un hôte nommé `learning`, le résolveur va rechercher `learning.lpi.org`. Vous pouvez configurer jusqu'à six domaines de recherche.

Une autre option courante est `domain`. Elle est utilisée pour définir votre nom de domaine local. Si cette option est absente, elle correspond par défaut à tout ce qui se trouve après le premier `.` dans le nom d'hôte de la machine. Si le nom d'hôte ne contient pas de `.`, on considère que la machine fait partie du domaine racine. Tout comme `search`, `domain` peut être utilisé pour des recherches de noms abrégés.

Gardez à l'esprit que `domain` et `search` s'excluent mutuellement. Si les deux sont présents, la dernière instance dans le fichier sera utilisée.

Vous pouvez définir plusieurs options qui affectent le comportement du résolveur. Pour les paramétrer, utilisez le mot-clé `options` suivi du nom de l'option à définir et, le cas échéant, d'un `:` suivi de la valeur. Voici un exemple de configuration de l'option `timeout` qui correspond à la durée en secondes pendant laquelle le résolveur attendra un serveur de noms avant d'abandonner :

```
option timeout:3
```

Il existe d'autres options pour `resolv.conf`, mais celles-ci sont les plus courantes.

Le fichier `/etc/hosts`

Le fichier `/etc/hosts` est utilisé pour résoudre les noms en adresses IP et vice versa. Les protocoles IPv4 et IPv6 sont tous deux pris en charge. La colonne de gauche est l'adresse IP, les autres sont les noms associés à cette adresse. L'utilisation la plus courante de `/etc/hosts` est pour les hôtes et les adresses où le DNS n'est pas possible, comme les adresses de boucle locale. Dans l'exemple ci-dessous, les adresses IP des composants critiques de l'infrastructure sont

définies.

Voici un exemple concret d'un fichier `/etc/hosts` :

```
127.0.0.1      localhost
127.0.1.1      proxy
::1           localhost ip6-localhost ip6-loopback
ff02::1       ip6-allnodes
ff02::2       ip6-allrouters

10.0.0.1       gateway.lpi.org gateway gw
fd00:ffff::1  gateway.lpi.org gateway gw

10.0.1.53      dns1.lpi.org
fd00:ffff::1:53 dns1.lpi.org
10.0.2.53      dns2.lpi.org
fd00:ffff::2:53 dns2.lpi.org
```

systemd-resolved

Systemd fournit un service appelé `systemd-resolved`. Il fournit mDNS, DNS et LLMNR. Lorsqu'il fonctionne, il écoute les requêtes DNS sur `127.0.0.53`. Il ne fournit *pas* de serveur DNS à proprement parler. Toutes les requêtes DNS qu'il reçoit sont traitées en interrogeant les serveurs configurés dans `/etc/systemd/resolv.conf` ou `/etc/resolv.conf`. Si vous souhaitez l'utiliser, ajoutez `resolve` pour `hosts` dans `/etc/nsswitch.conf`. Gardez à l'esprit que le paquet logiciel qui contient la bibliothèque `systemd-resolved` peut ne pas être installé par défaut.

Les outils de résolution de noms

Les utilisateurs de Linux disposent de plusieurs outils pour la résolution de noms. Cette leçon en présente trois. Le premier, `getent`, est utile pour voir comment les requêtes concrètes seront résolues. Une deuxième commande, `host`, est utile pour les requêtes DNS simples. Enfin, un troisième programme appelé `dig` est utile pour les opérations DNS complexes qui peuvent aider à diagnostiquer les problèmes des serveurs DNS.

La commande `getent`

La commande `getent` est utilisée pour afficher les entrées des bases de données des services de noms. Elle peut récupérer des enregistrements depuis n'importe quelle source configurable par `/etc/nsswitch.conf`.

Pour utiliser `getent`, indiquez en argument le type de nom que vous souhaitez résoudre et, le cas échéant, une entrée spécifique à rechercher. Si vous ne spécifiez que le type de nom, `getent` va essayer d'afficher toutes les entrées de ce type de données :

```
$ getent hosts
127.0.0.1      localhost
127.0.1.1      proxy
10.0.1.53      dns1.lpi.org
10.0.2.53      dns2.lpi.org
127.0.0.1      localhost ip6-localhost ip6-loopback
$ getent hosts dns1.lpi.org
fd00:ffff::1:53 dns1.lpi.org
```

À partir de la version 2.2.5 de la glibc, vous pouvez forcer `getent` à utiliser une source de données spécifique avec l'option `-s`. L'exemple ci-dessous le montre :

```
$ getent -s files hosts learning.lpi.org
::1          learning.lpi.org
$ getent -s dns hosts learning.lpi.org
208.94.166.198 learning.lpi.org
```

La commande `host`

`host` est un outil simple pour la recherche d'entrées DNS. Sans option, si `host` reçoit un nom, il retourne les enregistrements A, AAAA et MX. Si on lui fournit une adresse IPv4 ou IPv6, il retourne l'enregistrement PTR s'il existe :

```
$ host wikipedia.org
wikipedia.org has address 208.80.154.224
wikipedia.org has IPv6 address 2620:0:861:ed1a::1
wikipedia.org mail is handled by 10 mx1001.wikimedia.org.
wikipedia.org mail is handled by 50 mx2001.wikimedia.org.
$ host 208.80.154.224
224.154.80.208.in-addr.arpa domain name pointer text-lb.eqiad.wikimedia.org.
```

Si vous recherchez un type d'enregistrement spécifique, il suffit d'utiliser `host -t` :

```
$ host -t NS lpi.org
lpi.org name server dns1.easydns.com.
lpi.org name server dns3.easydns.ca.
```

```
lpi.org name server dns2.easydns.net.
$ host -t SOA lpi.org
lpi.org has SOA record dns1.easydns.com. zone.easydns.com. 1593109612 3600 600 1209600 300
```

`host` peut également être utilisé pour interroger un serveur de noms donné si vous ne voulez pas utiliser ceux du fichier `/etc/resolv.conf`. Ajoutez simplement l'adresse IP ou le nom d'hôte du serveur que vous souhaitez utiliser comme dernier argument :

```
$ host -t MX lpi.org dns1.easydns.com
Using domain server:
Name: dns1.easydns.com
Address: 64.68.192.10#53
Aliases:

lpi.org mail is handled by 10 aspmx4.googlemail.com.
lpi.org mail is handled by 10 aspmx2.googlemail.com.
lpi.org mail is handled by 5 alt1.aspmx.l.google.com.
lpi.org mail is handled by 0 aspmx.l.google.com.
lpi.org mail is handled by 10 aspmx5.googlemail.com.
lpi.org mail is handled by 10 aspmx3.googlemail.com.
lpi.org mail is handled by 5 alt2.aspmx.l.google.com.
```

La commande `dig`

Un autre outil pour interroger les serveurs DNS est `dig`. Cette commande est beaucoup plus bavarde que `host`. Par défaut, `dig` interroge les enregistrements A. Il est probablement trop bavard pour rechercher simplement une adresse IP ou un nom d'hôte. `dig` fonctionnera pour des recherches simples, mais il est plus adapté pour diagnostiquer la configuration d'un serveur DNS :

```
$ dig learning.lpi.org

; <<>> DiG 9.11.5-P4-5.1+deb10u1-Debian <<>> learning.lpi.org
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 63004
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 3, ADDITIONAL: 5

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: ca7a415be1cec45592b082665ef87f3483b81ddd61063c30 (good)
;; QUESTION SECTION:
```

```

;learning.lpi.org.      IN  A

;; ANSWER SECTION:
learning.lpi.org.      600 IN  A    208.94.166.198

;; AUTHORITY SECTION:
lpi.org.               86400 IN  NS   dns2.easydns.net.
lpi.org.               86400 IN  NS   dns1.easydns.com.
lpi.org.               86400 IN  NS   dns3.easydns.ca.

;; ADDITIONAL SECTION:
dns1.easydns.com.      172682 IN  A    64.68.192.10
dns2.easydns.net.      170226 IN  A    198.41.222.254
dns1.easydns.com.      172682 IN  AAAA   2400:cb00:2049:1::a29f:1835
dns2.easydns.net.      170226 IN  AAAA   2400:cb00:2049:1::c629:defe

;; Query time: 135 msec
;; SERVER: 192.168.1.20#53(192.168.1.20)
;; WHEN: Sun Jun 28 07:29:56 EDT 2020
;; MSG SIZE rcvd: 266

```

Comme vous le voyez, `dig` fournit beaucoup d'informations. Le résultat est divisé en plusieurs sections. La première section affiche des informations sur la version de `dig` installée et la requête envoyée, ainsi que toutes les options utilisées pour la commande. Elle affiche également des informations sur la requête et la réponse.

La prochaine section donne des informations sur les extensions EDNS utilisées et sur la requête. Dans l'exemple, l'extension *cookie* est utilisée. `dig` recherche un enregistrement A pour `learning.lpi.org`.

La section suivante présente le résultat de la requête. Le nombre dans la deuxième colonne correspond au TTL de la ressource en secondes.

Le reste du résultat fournit des informations sur les serveurs de noms du domaine, y compris les enregistrements NS du serveur ainsi que les enregistrements A et AAAA des serveurs dans l'enregistrement NS du domaine.

Comme pour `host`, vous pouvez spécifier un type d'enregistrement avec l'option `-t` :

```

$ dig -t SOA lpi.org

; <<>> DiG 9.11.5-P4-5.1+deb10u1-Debian <<>> -t SOA lpi.org
;; global options: +cmd

```

```
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 16695
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 3, ADDITIONAL: 6

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 185c67140a63baf46c4493215ef8906f7bfbe15bdca3b01a (good)
;; QUESTION SECTION:
;lpi.org.          IN  SOA

;; ANSWER SECTION:
lpi.org.          600 IN  SOA dns1.easydns.com. zone.easydns.com. 1593109612 3600 600 1209600
300

;; AUTHORITY SECTION:
lpi.org.          81989 IN  NS  dns1.easydns.com.
lpi.org.          81989 IN  NS  dns2.easydns.net.
lpi.org.          81989 IN  NS  dns3.easydns.ca.

;; ADDITIONAL SECTION:
dns1.easydns.com. 168271 IN  A   64.68.192.10
dns2.easydns.net. 165815 IN  A   198.41.222.254
dns3.easydns.ca.  107 IN  A   64.68.196.10
dns1.easydns.com. 168271 IN  AAAA 2400:cb00:2049:1::a29f:1835
dns2.easydns.net. 165815 IN  AAAA 2400:cb00:2049:1::c629:defe

;; Query time: 94 msec
;; SERVER: 192.168.1.20#53(192.168.1.20)
;; WHEN: Sun Jun 28 08:43:27 EDT 2020
;; MSG SIZE rcvd: 298
```

Dig dispose de nombreuses options qui permettent d'affiner à la fois le résultat et la requête envoyée au serveur. Ces options commencent par `+`. L'une d'entre elles est l'option `short`, qui supprime toutes les données à l'exception du résultat :

```
$ dig +short lpi.org
65.39.134.165
$ dig +short -t SOA lpi.org
dns1.easydns.com. zone.easydns.com. 1593109612 3600 600 1209600 300
```

Voici un exemple pour désactiver l'extension EDNS *cookie* :


```
$ dig +nocookie -t MX lpi.org
```

```
; <<>> DiG 9.11.5-P4-5.1+deb10u1-Debian <<>> +nocookie -t MX lpi.org
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 47774
;; flags: qr rd ra; QUERY: 1, ANSWER: 7, AUTHORITY: 3, ADDITIONAL: 5

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;lpi.org.                IN  MX

;; ANSWER SECTION:
lpi.org.                468 IN  MX  0 aspmx.l.google.com.
lpi.org.                468 IN  MX  10 aspmx4.googlemail.com.
lpi.org.                468 IN  MX  10 aspmx5.googlemail.com.
lpi.org.                468 IN  MX  10 aspmx2.googlemail.com.
lpi.org.                468 IN  MX  10 aspmx3.googlemail.com.
lpi.org.                468 IN  MX  5 alt2.aspmx.l.google.com.
lpi.org.                468 IN  MX  5 alt1.aspmx.l.google.com.

;; AUTHORITY SECTION:
lpi.org.                77130 IN  NS  dns2.easydns.net.
lpi.org.                77130 IN  NS  dns3.easydns.ca.
lpi.org.                77130 IN  NS  dns1.easydns.com.

;; ADDITIONAL SECTION:
dns1.easydns.com.       76140 IN  A    64.68.192.10
dns2.easydns.net.       73684 IN  A    198.41.222.254
dns1.easydns.com.       76140 IN  AAAA  2400:cb00:2049:1::a29f:1835
dns2.easydns.net.       73684 IN  AAAA  2400:cb00:2049:1::c629:de

;; Query time: 2 msec
;; SERVER: 192.168.1.20#53(192.168.1.20)
;; WHEN: Mon Jun 29 10:18:58 EDT 2020
;; MSG SIZE rcvd: 389
```

Exercices guidés

1. Que fait la commande ci-dessous ?

```
getent group openldap
```

2. Quelle est la principale différence entre `getent` et les autres outils présentés, `host` et `dig` ?

3. Quelle option de `dig` et `host` permet de spécifier le type d'enregistrement que vous souhaitez récupérer ?

4. Laquelle des entrées suivantes est correcte pour le fichier `/etc/hosts` ?

<code>::1 localhost</code>	
<code>localhost 127.0.0.1</code>	

5. Quelle option de `getent` permet de spécifier la source de données à utiliser pour effectuer une recherche ?

Exercices d'approfondissement

1. Si vous modifiez le fichier `/etc/resolv.conf` ci-dessous avec un éditeur de texte, que va-t-il se passer ?

```
# Generated by NetworkManager
nameserver 192.168.1.20
```

Les modifications seront écrasées par NetworkManager.	
NetworkManager va mettre à jour sa configuration avec vos modifications.	
Vos modifications n'auront aucun effet sur le système.	
NetworkManager sera désactivé.	

2. Que signifie la ligne suivante dans `/etc/nsswitch.conf` :

```
hosts: files [SUCCESS=continue] dns
```

3. Compte tenu du fichier `/etc/resolv.conf` ci-dessous, pourquoi le système ne résout-il pas les noms via le DNS ?

```
search lpi.org
#nameserver fd00:ffff::1:53
#nameserver 10.0.1.53
```

4. Que fait la commande `dig +noall +answer +question lpi.org` ?

5. Comment pouvez-vous remplacer les paramètres par défaut de `dig` sans les spécifier dans la ligne de commande ?

Résumé

La commande `getent` est un excellent outil pour afficher les résultats des requêtes du résolveur. Pour les requêtes DNS simples, `host` est facile à utiliser et affiche des résultats clairs. Si vous avez besoin d'informations détaillées ou si vous devez peaufiner une requête DNS, `dig` est sans doute le meilleur choix.

Grâce à la possibilité d'ajouter des plugins de bibliothèques partagées et de configurer le comportement du résolveur, Linux offre une excellente prise en charge de la résolution de noms et de valeurs numériques de différents types. Le programme `getent` peut être utilisé pour résoudre les noms à l'aide des bibliothèques du résolveur. `host` et `dig` peuvent être utilisés pour interroger les serveurs DNS.

Le fichier `/etc/nsswitch.conf` sert à configurer le comportement du résolveur. Vous pouvez modifier les sources de données et ajouter une logique conditionnelle simple pour les types de noms avec plusieurs sources.

Le DNS est configuré en modifiant le fichier `/etc/resolv.conf`. La plupart des distributions disposent d'outils qui gèrent ce fichier pour vous. Assurez-vous de consulter la documentation de votre système si les modifications manuelles ne sont pas persistantes.

Le fichier `/etc/hosts` est utilisé pour résoudre les noms d'hôtes en adresses IP et vice versa. Il est généralement utilisé pour définir des noms qui ne sont pas disponibles via le DNS, comme `localhost`.

Vous pouvez ajouter des commentaires dans les fichiers de configuration abordés dans cette leçon. Tout texte situé à droite du signe `#` sera ignoré par le système.

Réponses aux exercices guidés

1. Que fait la commande ci-dessous ?

```
getent group openldap
```

Elle lit le fichier `/etc/group`, recherche le groupe `openldap` parmi les sources répertoriées et affiche les informations à son sujet si elle le trouve.

2. Quelle est la principale différence entre `getent` et les autres outils présentés, `host` et `dig` ?

`getent` recherche les noms à l'aide des bibliothèques de résolution, tandis que les autres outils interrogent simplement le DNS. `getent` peut être utilisé pour diagnostiquer les problèmes liés à votre fichier `/etc/nsswitch.conf` et à la configuration des bibliothèques de résolution de noms que votre système est censé utiliser. `host` et `dig` sont utilisés pour rechercher des enregistrements DNS.

3. Quelle option de `dig` et `host` permet de spécifier le type d'enregistrement que vous souhaitez récupérer ?

Les deux programmes utilisent `-t` pour spécifier le type d'enregistrement que vous souhaitez rechercher.

4. Laquelle des entrées suivantes est correcte pour le fichier `/etc/hosts` ?

<code>::1 localhost</code>	X
<code>localhost 127.0.0.1</code>	

`::1 localhost` est la ligne correcte. La colonne de gauche indique toujours une adresse IPv4 ou IPv6.

5. Quelle option de `getent` permet de spécifier la source de données à utiliser pour effectuer une recherche ?

L'option `-s` permet de spécifier la source de données. Par exemple :

```
$ getent -s files hosts learning.lpi.org
192.168.10.25    learning.lpi.org
$ getent -s dns hosts learning.lpi.org
208.94.166.198  learning.lpi.org
```

Réponses aux exercices d'approfondissement

1. Si vous modifiez le fichier `/etc/resolv.conf` ci-dessous avec un éditeur de texte, que va-t-il se passer ?

```
# Generated by NetworkManager
nameserver 192.168.1.20
```

Les modifications seront écrasées par NetworkManager.	X
NetworkManager va mettre à jour sa configuration avec vos modifications.	
Vos modifications n'auront aucun effet sur le système.	
NetworkManager sera désactivé.	

2. Que signifie la ligne suivante dans `/etc/nsswitch.conf` :

```
hosts: files [SUCCESS=continue] dns
```

Les recherches de noms d'hôtes vont d'abord vérifier votre fichier `/etc/hosts`, puis le DNS. Si une entrée est trouvée dans le fichier et dans le DNS, c'est l'entrée du DNS qui sera utilisée.

3. Compte tenu du fichier `/etc/resolv.conf` ci-dessous, pourquoi le système ne résout-il pas les noms via le DNS ?

```
search lpi.org
#nameserver fd00::ffff::1:53
#nameserver 10.0.1.53
```

Les deux serveurs DNS sont commentés et aucun serveur DNS n'est en cours d'exécution sur l'hôte local.

4. Que fait la commande `dig +noall +answer +question lpi.org` ?

Elle recherche l'enregistrement A pour `lpi.org` et affiche uniquement la requête et la réponse.

5. Comment pouvez-vous remplacer les paramètres par défaut de `dig` sans les spécifier dans la

ligne de commande ?

Vous créez un fichier `.digrc` dans votre répertoire personnel.



Thème 110 : Sécurité



110.1 Tâches d'administration de sécurité

Référence aux objectifs de LPI

LPIC-1 version 5.0, Exam 102, Objective 110.1

Valeur

3

Domaines de connaissance les plus importants

- Audit du système pour retrouver les fichiers ayant les permissions suid/guid positionnées.
- Définition ou modification des mots de passe utilisateur ainsi que des informations d'expiration du compte.
- Utilisation de nmap et netstat pour connaître les ports ouverts sur une machine
- Définition des limites utilisateur pour les connexions, les processus et l'utilisation de la mémoire.
- Détermination des connexions utilisateur passées ou actuelles.
- Configuration et utilisation élémentaire de sudo.

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

- `find`
- `passwd`
- `fuser`
- `lsof`
- `nmap`
- `chage`
- `netstat`

- `sudo`
- `/etc/sudoers`
- `su`
- `usermod`
- `ulimit`
- `who, w, last`



110.1 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	110 Sécurité
Objectif :	110.1 Tâches administratives pour la sécurité
Leçon :	1 sur 1

Introduction

La sécurité est indispensable dans l'administration système. Tout administrateur Linux qui se respecte se doit de garder un œil sur un certain nombre d'éléments tels que les permissions spéciales sur les fichiers, l'expiration des mots de passe des utilisateurs, les ports et les sockets ouverts, l'utilisation des ressources système, la gestion des utilisateurs connectés et l'élévation des privilèges via `su` et `sudo`. Dans cette leçon, nous allons passer en revue chacun de ces sujets.

Rechercher les fichiers avec les permissions SUID et SGID

En dehors des droits d'accès traditionnels en *lecture*, en *écriture* et en *exécution*, les fichiers d'un système Linux peuvent également disposer de droits d'accès spéciaux tels que les bits *SUID* et *SGID*.

Le bit SUID permet d'exécuter le fichier avec les privilèges du propriétaire du fichier. Il est représenté numériquement par `4000` et symboliquement par `s` ou `S` sur le bit d'exécution du propriétaire. Un exemple classique de fichier exécutable avec la permission SUID est `passwd` :

```
carol@debian:~$ ls -l /usr/bin/passwd
```

```
-rwsr-xr-x 1 root root 63736 jul 27 2018 /usr/bin/passwd
```

Le **s** minuscule dans **rws** indique la présence du SUID sur le fichier, ainsi que les droits d'exécution. Un **S** majuscule (**rwS**) signifierait en revanche que la permission d'exécution sous-jacente n'est pas accordée.

NOTE

Vous découvrirez la commande **passwd** un peu plus loin. Cet outil est principalement utilisé par **root** pour définir/modifier les mots de passe des utilisateurs (par exemple : **passwd carol**). Cela dit, les utilisateurs normaux peuvent également s'en servir pour changer leur mot de passe. C'est la raison pour laquelle la commande est fournie avec le bit SUID.

D'autre part, le bit SGID peut être défini à la fois sur les fichiers et les répertoires. Avec les fichiers, son comportement est similaire à celui du SUID, mais les droits sont ceux du groupe propriétaire. Lorsqu'il est défini sur un répertoire, il permet à tous les fichiers créés dans celui-ci d'hériter de la propriété du groupe du répertoire. Comme le SUID, le SGID est représenté symboliquement par **s** ou **S** sur le bit d'exécution du groupe. Numériquement, il est représenté par **2000**. Vous pouvez définir le SGID sur un répertoire à l'aide de **chmod**. Vous devrez ajouter **2** (SGID) aux permissions traditionnelles (755 dans notre cas) :

```
carol@debian:~$ ls -ld shared_directory
drwxr-xr-x 2 carol carol 4096 may 30 23:55 shared_directory
carol@debian:~$ sudo chmod 2755 shared_directory/
carol@debian:~$ ls -ld shared_directory
drwxr-sr-x 2 carol carol 4096 may 30 23:55 shared_directory
```

Pour rechercher des fichiers dont le bit SUID et, ou SGID est activé, vous pouvez utiliser la commande **find** avec l'option **-perm**. Les valeurs numériques et symboliques sont acceptées. Elles peuvent être saisies seules ou précédées d'un tiret (-) ou d'une barre oblique (/). Voici leur signification :

-perm valeur_numérique ou -perm valeur_symbolique

Rechercher les fichiers ayant la permission spéciale *exclusivement*.

-perm -valeur_numérique ou -perm -valeur_symbolique

Rechercher les fichiers ayant la permission spéciale et d'autres permissions.

-perm /valeur-numérique ou -perm /valeur-symbolique

Rechercher les fichiers ayant l'une ou l'autre des permissions spéciales (et d'autres permissions).

Par exemple, pour trouver les fichiers avec *uniquement* le bit SUID activé dans le répertoire courant, vous utiliserez la commande suivante :

```
carol@debian:~$ find . -perm 4000
carol@debian:~$ touch file
carol@debian:~$ chmod 4000 file
carol@debian:~$ find . -perm 4000
./file
```

Comme il n'y avait aucun fichier disposant exclusivement du SUID, notez que nous en avons créé un pour afficher un résultat. Vous pouvez d'ailleurs exécuter la même commande en notation symbolique :

```
carol@debian:~$ find . -perm u+s
./file
```

Pour trouver les fichiers avec un SUID (indépendamment de toute autre permission) dans le répertoire `/usr/bin/`, vous pouvez utiliser l'une des commandes suivantes :

```
carol@debian:~$ sudo find /usr/bin -perm -4000
/usr/bin/umount
/usr/bin/newgrp
/usr/bin/gpasswd
/usr/bin/chfn
/usr/bin/mount
/usr/bin/passwd
/usr/bin/chsh
/usr/bin/sudo
/usr/bin/su
carol@debian:~$ sudo find /usr/bin -perm -u+s
/usr/bin/umount
/usr/bin/newgrp
/usr/bin/gpasswd
/usr/bin/chfn
/usr/bin/mount
/usr/bin/passwd
/usr/bin/chsh
/usr/bin/sudo
/usr/bin/su
```

Si vous recherchez des fichiers dans le même répertoire avec le bit SGID défini, vous pouvez

exécuter `find /usr/bin/ -perm -2000` ou `find /usr/bin/ -perm -g+s`.

Enfin, pour trouver les fichiers auxquels l'une des deux autorisations spéciales a été attribuée, additionnez 4 et 2 et utilisez `/` :

```
carol@debian:~$ sudo find /usr/bin -perm /6000
/usr/bin/dotlock.mailutils
/usr/bin/umount
/usr/bin/newgrp
/usr/bin/wall
/usr/bin/ssh-agent
/usr/bin/chage
/usr/bin/dotlockfile
/usr/bin/gpasswd
/usr/bin/chfn
/usr/bin/mount
/usr/bin/passwd
/usr/bin/chsh
/usr/bin/expiry
/usr/bin/sudo
/usr/bin/bsd-write
/usr/bin/crontab
/usr/bin/su
```

Gestion et expiration des mots de passe

Comme nous venons de le voir, vous pouvez utiliser la commande `passwd` pour modifier votre mot de passe en tant qu'utilisateur normal. Par ailleurs, le paramètre `-S` ou `--status` permet d'obtenir des informations sur l'état de votre compte :

```
carol@debian:~$ passwd -S
carol P 12/07/2019 0 99999 7 -1
```

Voici une description détaillée des sept champs qui s'affichent dans le résultat :

carol

Nom d'utilisateur.

P

Indique que l'utilisateur dispose d'un mot de passe valide (P) ; les autres valeurs possibles sont L pour un mot de passe verrouillé (*locked*) et NP pour un mot de passe inexistant (*no password*).

12/07/2019

Date du dernier changement de mot de passe.

0

Durée minimale en jours (nombre minimal de jours entre deux changements de mot de passe). Une valeur de 0 signifie que le mot de passe peut être changé à tout moment.

99999

Durée maximale en jours (nombre maximal de jours pendant lesquels le mot de passe est valide). Une valeur de 99999 désactive l'expiration du mot de passe.

7

Période d'avertissement en jours (nombre de jours avant l'expiration du mot de passe pendant lesquels l'utilisateur sera averti).

-1

Période d'inactivité du mot de passe en jours (nombre de jours d'inactivité après l'expiration du mot de passe avant le verrouillage du compte). Une valeur de -1 supprimera l'inactivité d'un compte.

En dehors de la consultation de l'état d'un compte, vous utiliserez la commande `passwd` en tant que root pour effectuer certaines opérations de maintenance de base sur les comptes. Vous pouvez verrouiller et déverrouiller des comptes, forcer un utilisateur à changer son mot de passe lors de sa prochaine connexion et supprimer le mot de passe d'un utilisateur à l'aide des options `-l`, `-u`, `-e` et `-d` respectivement.

Pour tester ces options, le moment est venu de présenter la commande `su`. Grâce à `su`, vous pouvez changer d'utilisateur pendant une session de connexion. Par exemple, utilisons `passwd` en tant que root pour verrouiller le mot de passe de `carol`. Ensuite, nous allons basculer vers `carol` et vérifier l'état de notre compte pour nous assurer que le mot de passe a bien été verrouillé (L) et ne peut pas être modifié. Enfin, en revenant à l'utilisateur root, nous allons déverrouiller le mot de passe de `carol` :

```
root@debian:~# passwd -l carol
passwd: password expiry information changed.
root@debian:~# su - carol
carol@debian:~$ passwd -S
carol L 05/31/2020 0 99999 7 -1
carol@debian:~$ passwd
Changing password for carol.
Current password:
```

```
passwd: Authentication token manipulation error
passwd: password unchanged
carol@debian:~$ exit
logout
root@debian:~# passwd -u carol
passwd: password expiry information changed.
```

Une autre solution consiste à verrouiller et déverrouiller le mot de passe d'un utilisateur à l'aide de la commande `usermod` :

Verrouiller le mot de passe de carol

```
usermod -L carol ou usermod --lock carol.
```

Déverrouiller le mot de passe de carol

```
usermod -U carol ou usermod --unlock carol.
```

NOTE

Les options `-f` ou `--inactive` permettent d'utiliser `usermod` pour définir le nombre de jours avant la désactivation d'un compte dont le mot de passe a expiré (par exemple : `usermod -f 3 carol`).

En dehors de `passwd` et `usermod`, la commande la plus immédiate pour gérer les mots de passe et la durée de validité des comptes est `chage` ("change age"). En tant que `root`, vous pouvez passer à `chage` l'option `-l` (ou `--list`) suivie d'un nom d'utilisateur pour afficher le mot de passe actuel et les informations d'expiration du compte de cet utilisateur ; en tant qu'utilisateur normal, vous pouvez afficher vos propres informations :

```
carol@debian:~$ chage -l carol
Last password change           : Aug 06, 2019
Password expires                : never
Password inactive               : never
Account expires                 : never
Minimum number of days between password change : 0
Maximum number of days between password change : 99999
Number of days of warning before password expires : 7
```

Exécutée sans options avec un seul nom d'utilisateur en argument, la commande `chage` va fonctionner de manière interactive :

```
root@debian:~# chage carol
Changing the aging information for carol
Enter the new value, or press ENTER for the default
```



```

Minimum Password Age [0]:
Maximum Password Age [99999]:
Last Password Change (YYYY-MM-DD) [2020-06-01]:
Password Expiration Warning [7]:
Password Inactive [-1]:
Account Expiration Date (YYYY-MM-DD) [-1]:

```

Voici les options qui permettent de modifier les différents paramètres de chage :

-m jours nom d'utilisateur ou --mindays jours nom d'utilisateur

Définit le nombre minimum de jours entre deux changements de mot de passe (par exemple : chage -m 5 carol). Une valeur de 0 permet à l'utilisateur de changer son mot de passe à tout moment.

-M jours nom d'utilisateur ou --maxdays jours nom d'utilisateur

Définit le nombre maximal de jours pendant lesquels le mot de passe sera valide (par exemple : chage -M 30 carol). Pour désactiver l'expiration du mot de passe, il est d'usage de donner à cette option la valeur 99999.

-d jours nom d'utilisateur ou --lastday jours nom d'utilisateur

Définit le nombre de jours écoulés depuis la dernière modification du mot de passe (par exemple : chage -d 10 carol). Une valeur de 0 obligera l'utilisateur à modifier son mot de passe lors de la prochaine connexion.

-W jours nom d'utilisateur ou --warndays jours nom d'utilisateur

Définit le nombre de jours pendant lesquels l'utilisateur sera averti de l'expiration de son mot de passe.

-I jours nom d'utilisateur ou --inactive jours nom d'utilisateur

Définit le nombre de jours d'inactivité après l'expiration du mot de passe (par exemple : chage -I 10 carol)—équivalent à `usermod -f` ou `usermod --inactive`. Une fois ce nombre de jours écoulé, le compte sera verrouillé. Avec une valeur de 0, le compte ne sera pas verrouillé en revanche.

-E date nom d'utilisateur ou --expiredate date nom d'utilisateur

Définit la date (ou le nombre de jours depuis l'époque, c'est-à-dire le 1er janvier 1970) à laquelle le compte sera verrouillé. Elle est généralement exprimée au format AAAA-MM-JJ (par exemple : chage -E 2050-12-13 carol).

NOTE Pour en savoir plus sur les commandes `passwd`, `usermod` et `chage`, ainsi que sur

leurs options, consultez leurs pages de manuel respectives.

Dépistage des ports ouverts

Pour garder un œil sur les ports ouverts, les systèmes Linux disposent généralement de quatre outils efficaces : `lsof`, `fuser`, `netstat` et `nmap`. Nous allons les présenter dans cette section.

`lsof` signifie “list open files” (afficher les fichiers ouverts), ce qui n’est pas anodin si l’on considère que, sous Linux, tout est fichier. En fait, si vous tapez `lsof` dans le terminal, vous obtenez une longue liste de fichiers classiques, de fichiers de périphériques, de sockets, etc. En revanche, dans le cadre de cette leçon, nous allons nous concentrer principalement sur les ports. Pour afficher la liste de tous les fichiers réseau “Internet”, invoquez `lsof` avec l’option `-i` :

```
root@debian:~# lsof -i
COMMAND PID    USER  FD  TYPE DEVICE SIZE/OFF NODE NAME
dhclient 357    root   7u  IPv4 13493      0t0  UDP *:bootpc
sshd     389    root   3u  IPv4 13689      0t0  TCP *:ssh (LISTEN)
sshd     389    root   4u  IPv6 13700      0t0  TCP *:ssh (LISTEN)
apache2  399    root   3u  IPv6 13826      0t0  TCP *:http (LISTEN)
apache2  401 www-data 3u  IPv6 13826      0t0  TCP *:http (LISTEN)
apache2  402 www-data 3u  IPv6 13826      0t0  TCP *:http (LISTEN)
sshd     557    root   3u  IPv4 14701      0t0  TCP 192.168.1.7:ssh->192.168.1.4:60510
(ESTABLISHED)
sshd     569    carol   3u  IPv4 14701      0t0  TCP 192.168.1.7:ssh->192.168.1.4:60510
(ESTABLISHED)
```

En dehors du service `bootpc` — utilisé par DHCP — le résultat affiche deux services à l’écoute de connexions — `ssh` et le serveur web Apache (`http`) — ainsi que deux connexions SSH établies. Vous pouvez spécifier un hôte particulier à l’aide de la notation `@adresse_ip` pour vérifier ses connexions :

```
root@debian:~# lsof -i@192.168.1.7
COMMAND PID  USER  FD  TYPE DEVICE SIZE/OFF NODE NAME
sshd     557  root   3u  IPv4 14701      0t0  TCP 192.168.1.7:ssh->192.168.1.4:60510
(ESTABLISHED)
sshd     569 carol   3u  IPv4 14701      0t0  TCP 192.168.1.7:ssh->192.168.1.4:60510
(ESTABLISHED)
```

NOTE

Pour afficher uniquement les fichiers réseau IPv4 et IPv6, utilisez les options `-i4` et `-i6` respectivement.

De même, vous pouvez filtrer par port en passant l'argument `:port` à l'option `-i` (ou `-i@ip-address`) :

```
root@debian:~# lsof -i :22
COMMAND PID  USER  FD   TYPE DEVICE SIZE/OFF NODE NAME
sshd     389   root   3u    IPv4  13689      0t0  TCP *:ssh (LISTEN)
sshd     389   root   4u    IPv6  13700      0t0  TCP *:ssh (LISTEN)
sshd     557   root   3u    IPv4  14701      0t0  TCP 192.168.1.7:ssh->192.168.1.4:60510
(ESTABLISHED)
sshd     569   carol   3u    IPv4  14701      0t0  TCP 192.168.1.7:ssh->192.168.1.4:60510
(ESTABLISHED)
```

Les ports multiples sont séparés par des virgules (et les plages sont spécifiées à l'aide d'un tiret) :

```
root@debian:~# lsof -i@192.168.1.7:22,80
COMMAND PID  USER  FD   TYPE DEVICE SIZE/OFF NODE NAME
sshd     705   root   3u    IPv4  13960      0t0  TCP 192.168.1.7:ssh->192.168.1.4:44766
(ESTABLISHED)
sshd     718   carol   3u    IPv4  13960      0t0  TCP 192.168.1.7:ssh->192.168.1.4:44766
(ESTABLISHED)
```

NOTE

Le nombre d'options disponibles pour `lsof` est assez impressionnant. Pour en savoir plus, consultez sa page de manuel.

La prochaine commande réseau sur la liste est `fuser`. Son objectif principal est de trouver "l'utilisateur d'un fichier", ce qui revient à savoir quels processus accèdent à quels fichiers ; elle fournit également d'autres informations comme le type d'accès. Par exemple, pour vérifier le répertoire de travail actuel, il suffit d'exécuter `fuser`. Pour en savoir plus, il vaut mieux utiliser l'option bavarde (`-v` ou `--verbose`) :

```
root@debian:~# fuser .
/root:                    580c
root@debian:~# fuser -v .
USER      PID ACCESS COMMAND
/root:    root      580 ..c.. bash
```

Analysons le résultat :

Fichier

Le fichier sur lequel nous obtenons des informations (`/root`).

Colonne USER

Le propriétaire du fichier (root).

Colonne PID

L'identifiant du processus (580).

Colonne ACCESS

Type d'accès (. . c . .). L'un des suivants :

c

Répertoire actuel.

e

Exécutable en cours d'exécution.

f

Fichier ouvert (omis dans le mode d'affichage par défaut).

F

Fichier ouvert en écriture (omis dans le mode d'affichage par défaut).

r

Répertoire racine

m

Fichier mmap ou bibliothèque partagée.

.

Espace réservé (omis dans le mode d'affichage par défaut).

Colonne COMMAND

La commande associée au fichier (bash).

L'option `-n` (ou `--namespace`) vous permet d'obtenir des informations sur les ports/sockets réseau. Vous devez également fournir le protocole réseau et le numéro de port. Ainsi, pour obtenir des informations sur le serveur web Apache, vous devez exécuter la commande suivante :

```
root@debian:~# fuser -vn tcp 80
USER      PID ACCESS COMMAND
80/tcp:   root    402 F.... apache2
          www-data 404 F.... apache2
```

```
www-data    405 F....  apache2
```

NOTE

`fuser` peut également être utilisé pour tuer les processus qui accèdent au fichier à l'aide des options `-k` ou `--kill` (par exemple `fuser -k 80/tcp`). Consultez la page du manuel pour en savoir plus.

Passons maintenant à `netstat`, un outil réseau très polyvalent qui sert principalement à afficher les “statistiques réseau”.

Exécutée sans options, la commande `netstat` affiche à la fois les connexions Internet actives et les sockets Unix. En raison de la taille de l’affichage, il vaut mieux combiner la commande avec `less` :

```
carol@debian:~$ netstat | less
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 192.168.1.7:ssh        192.168.1.4:55444      ESTABLISHED
Active UNIX domain sockets (w/o servers)
Proto RefCnt Flags       Type        State         I-Node  Path
unix    2      [ ]          DGRAM                    10509   /run/systemd/journal/syslog
unix    3      [ ]          DGRAM                    10123   /run/systemd/notify
(...)
```

Pour afficher uniquement les ports et sockets “en écoute”, utilisez les options `-l` ou `--listening`. Les options `-t/--tcp` et `-u/--udp` peuvent être ajoutées pour filtrer respectivement par protocole TCP et UDP (elles peuvent également être combinées dans la même commande). De même, `-e/--extend` va afficher des informations supplémentaires :

```
carol@debian:~$ netstat -lu
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
udp        0      0 0.0.0.0:bootpc         0.0.0.0:*
carol@debian:~$ netstat -lt
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:ssh            0.0.0.0:*               LISTEN
tcp        0      0 localhost:smtp         0.0.0.0:*               LISTEN
tcp6       0      0 [::]:http             [::]:*                  LISTEN
tcp6       0      0 [::]:ssh               [::]:*                  LISTEN
tcp6       0      0 localhost:smtp         [::]:*                  LISTEN
carol@debian:~$ netstat -lute
Active Internet connections (only servers)
```

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State	User
Inode						
tcp	0	0	0.0.0.0:ssh	0.0.0.0:*	LISTEN	root
13729						
tcp	0	0	localhost:smtp	0.0.0.0:*	LISTEN	root
14372						
tcp6	0	0	:::http	:::*	LISTEN	root
14159						
tcp6	0	0	:::ssh	:::*	LISTEN	root
13740						
tcp6	0	0	localhost:smtp	:::*	LISTEN	root
14374						
udp	0	0	0.0.0.0:bootpc	0.0.0.0:*		root
13604						

Si vous omettez l'option `-l`, seules les connexions établies seront affichées :

```
carol@debian:~$ netstat -ute
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       User
Inode
tcp      0      0 192.168.1.7:ssh        192.168.1.4:39144      ESTABLISHED root
15103
```

Si vous vous intéressez uniquement aux informations numériques concernant les ports et les hôtes, vous pouvez utiliser l'option `-n` ou `--numeric` pour n'afficher que les numéros de port et les adresses IP. Notez comment `ssh` devient `22` lorsque vous ajoutez `-n` à la commande ci-dessus :

```
carol@debian:~$ netstat -uten
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       User
Inode
tcp      0      0 192.168.1.7:22        192.168.1.4:39144      ESTABLISHED 0
15103
```

Comme vous pouvez le constater, certaines combinaisons d'options permettent de créer des commandes `netstat` très utiles et efficaces. Consultez les pages de manuel pour en savoir plus et trouver les combinaisons qui correspondent le mieux à vos besoins.

Pour finir, nous allons présenter `nmap`, ou “_network mapper_”. Un autre outil très puissant, ce scanner de ports s'exécute en spécifiant une adresse IP ou un nom d'hôte :

```

root@debian:~# nmap localhost
Starting Nmap 7.70 ( https://nmap.org ) at 2020-06-04 19:29 CEST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.0000040s latency).
Other addresses for localhost (not scanned): ::1
Not shown: 998 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http

Nmap done: 1 IP address (1 host up) scanned in 1.58 seconds

```

Au-delà d'un seul hôte, `nmap` vous permet de scanner :

Plusieurs hôtes

En les séparant par des espaces (par exemple : `nmap localhost 192.168.1.7`).

Plages d'hôtes

En utilisant un tiret (par exemple : `nmap 192.168.1.3-20`).

Sous-réseaux

En utilisant un métacaractère ou la notation CIDR (par exemple : `nmap 192.168.1.*` ou `nmap 192.168.1.0/24`). Vous pouvez exclure des hôtes particuliers (par exemple : `nmap 192.168.1.0/24 --exclude 192.168.1.7`).

Pour scanner un port en particulier, utilisez l'option `-p` suivie du numéro de port ou du nom du service (`nmap -p 22` et `nmap -p ssh` donneront le même résultat) :

```

root@debian:~# nmap -p 22 localhost
Starting Nmap 7.70 ( https://nmap.org ) at 2020-06-04 19:54 CEST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000024s latency).
Other addresses for localhost (not scanned): ::1

PORT      STATE SERVICE
22/tcp    open  ssh

Nmap done: 1 IP address (1 host up) scanned in 0.22 seconds

```

Vous pouvez également scanner plusieurs ports ou plages de ports en utilisant des virgules et des tirets :

```
root@debian:~# nmap -p ssh,80 localhost
Starting Nmap 7.70 ( https://nmap.org ) at 2020-06-04 19:58 CEST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000051s latency).
Other addresses for localhost (not scanned): ::1

PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http

Nmap done: 1 IP address (1 host up) scanned in 0.22 seconds
```

```
root@debian:~# nmap -p 22-80 localhost
Starting Nmap 7.70 ( https://nmap.org ) at 2020-06-04 19:58 CEST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000011s latency).
Other addresses for localhost (not scanned): ::1
Not shown: 57 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http

Nmap done: 1 IP address (1 host up) scanned in 1.47 seconds
```

Voici deux autres options importantes et pratiques de `nmap` :

-F

Effectuer une analyse rapide des 100 ports les plus courants.

-v

Obtenir un affichage détaillé (`-vv` affichera un résultat encore plus détaillé).

NOTE

`nmap` peut exécuter des commandes assez complexes en utilisant différents types de scan. Ce sujet dépasse toutefois le cadre de cette leçon.

Limiter les connexions des utilisateurs, les processus et l'utilisation de la mémoire

Les ressources d'un système Linux ne sont pas illimitées. En tant qu'administrateur système, vous devez donc veiller à trouver un bon équilibre entre les limites d'utilisation des ressources par les

utilisateurs et le bon fonctionnement du système d'exploitation. La commande `ulimit` peut vous aider à cet égard.

`ulimit` gère les limites *souples* (*soft*) et *strictes* (*hard*), spécifiées respectivement par les options `-S` et `-H`. Exécuté sans option ni argument, `ulimit` affiche les blocs de fichiers souples de l'utilisateur actuel :

```
carol@debian:~$ ulimit
unlimited
```

Avec l'option `-a`, `ulimit` affichera toutes les limites souples actuelles (comme avec `-Sa`) ; pour afficher toutes les limites strictes en vigueur, utilisez `-Ha` :

```
carol@debian:~$ ulimit -a
core file size          (blocks, -c) 0
data seg size           (kbytes, -d) unlimited
scheduling priority     (-e) 0
(...)
carol@debian:~$ ulimit -Ha
core file size          (blocks, -c) unlimited
data seg size           (kbytes, -d) unlimited
scheduling priority     (-e) 0
(...)
```

Les ressources shell disponibles sont spécifiées par des options comme :

-b

Taille maximale du tampon de socket.

-f

Taille maximale des fichiers écrits par le shell et ses processus enfants.

-l

Taille maximale pouvant être verrouillée en mémoire.

-m

Taille maximale du jeu résident (RSS ou *resident set size*)—partie actuelle de la mémoire occupée par un processus dans la mémoire principale (RAM).

-v

Quantité maximale de mémoire virtuelle.

-u

Nombre maximal de processus disponibles pour un seul utilisateur

Ainsi, pour afficher les limites, vous utiliserez `ulimit` suivi de `-S` (*soft*) ou `-H` (*hard*) et de la ressource en question ; si ni `-S` ni `-H` ne sont fournis, les limites *soft* seront affichées :

```
carol@debian:~$ ulimit -u
10000
carol@debian:~$ ulimit -Su
10000
carol@debian:~$ ulimit -Hu
15672
```

De même, pour définir de nouvelles limites sur une ressource donnée, vous devez spécifier soit `-S` soit `-H` suivi de l'option de ressource correspondante et de la nouvelle valeur. Cette valeur peut être un nombre ou les mots clés `soft` (limite souple actuelle), `hard` (limite stricte actuelle) ou `unlimited` (aucune limite). Si ni `-S` ni `-H` ne sont spécifiés, les deux limites seront définies. Par exemple, consultons d'abord la valeur de la taille maximale actuelle pour les fichiers écrits par le shell et ses enfants :

```
root@debian:~# ulimit -Sf
unlimited
root@debian:~# ulimit -Hf
unlimited
```

Maintenant, modifions la valeur de `unlimited` (illimité) à `500` blocs sans spécifier ni `-S` ni `-H`. Notez comment les limites souples et strictes sont modifiées :

```
root@debian:~# ulimit -f 500
root@debian:~# ulimit -Sf
500
root@debian:~# ulimit -Hf
500
```

Pour finir, nous allons réduire uniquement la limite souple à 200 blocs :

```
root@debian:~# ulimit -Sf 200
root@debian:~# ulimit -Sf
200
root@debian:~# ulimit -Hf
500
```

Les limites strictes ne peuvent être augmentées que par l'utilisateur root. En revanche, les utilisateurs normaux peuvent réduire les limites strictes et augmenter les limites souples jusqu'à la valeur des limites strictes. Pour que les nouvelles valeurs limites soient conservées après le redémarrage, vous devez les écrire dans le fichier `/etc/security/limits.conf`. C'est également le fichier utilisé par l'administrateur pour appliquer des restrictions à certains utilisateurs.

NOTE

Attention, il n'existe pas de page de manuel `ulimit` à proprement parler. Il s'agit d'une commande interne de Bash, vous devez donc vous reporter à la page de manuel de Bash (`man bash`) pour en savoir plus.

Gérer les utilisateurs connectés

En tant qu'administrateur système, vous devez également assurer le suivi des utilisateurs connectés. Trois outils peuvent vous aider dans cette tâche : `last`, `who` et `w`.

`last` affiche la liste des derniers utilisateurs connectés, avec les informations les plus récentes en haut :

```
root@debian:~# last
carol    pts/0      192.168.1.4    Sat Jun 6 14:25  still logged in
reboot   system boot 4.19.0-9-amd64 Sat Jun 6 14:24  still running
mimi     pts/0      192.168.1.4    Sat Jun 6 12:07 - 14:24  (02:16)
reboot   system boot 4.19.0-9-amd64 Sat Jun 6 12:07 - 14:24  (02:17)
(...)
wtmp begins Sun May 31 14:14:58 2020
```

En considérant la liste tronquée, on obtient des informations sur les deux derniers utilisateurs du système. Les deux premières lignes nous renseignent sur l'utilisatrice `carol` ; les deux lignes suivantes, sur l'utilisatrice `mimi`. Voici les informations fournies :

1. L'utilisatrice `carol` sur le terminal `pts/0` de l'hôte `192.168.1.4` a démarré sa session le samedi 6 juin (Sat Jun 6) à 14 h 25. Elle est toujours connectée (`still logged in`). Le système, qui utilise le noyau `4.19.0-9-amd64`, a été démarré (`reboot system boot`) le

samedi 6 juin à 14 h 24. Il est toujours en cours d'exécution (`still running`).

2. L'utilisatrice `mimi` sur le terminal `pts/0` de l'hôte `192.168.1.4` a démarré sa session le samedi 6 juin à 12 h 07 et s'est déconnectée (`logged out`) à 14 h 24. La session a duré au total 2 heures et 16 minutes (`02:16`). Le système, qui utilise le noyau `4.19.0-9-amd64`, a été démarré (`reboot system boot`) le samedi 6 juin à 12h07 et a été éteint à 14h24. Il a fonctionné pendant 2 heures et 17 minutes (`02:17`).

NOTE

La ligne `wtmp begins Sun May 31 14:14:58 2020` fait référence à `/var/log/wtmp`, le fichier journal spécifique à partir duquel `last` récupère ses informations.

Vous pouvez fournir un nom d'utilisateur en argument à `last` pour afficher uniquement les entrées pour cet utilisateur :

```
root@debian:~# last carol
carol pts/0 192.168.1.4 Sat Jun 6 14:25 still logged in
carol pts/0 192.168.1.4 Sat Jun 6 12:07 - 14:24 (02:16)
carol pts/0 192.168.1.4 Fri Jun 5 00:48 - 01:28 (00:39)
(...)
```

Dans la deuxième colonne (`terminal`), `pts` signifie *Pseudo Terminal Slave* (esclave de pseudo-terminal) par opposition à un véritable terminal *TeLeTYpewriter* (téléscripateur) ou `tty` ; `0` fait référence au premier (le décompte commence à zéro).

NOTE

Pour afficher les tentatives de connexion incorrectes, utilisez `lastb` au lieu de `last`.

Les commandes `who` et `w` se concentrent sur les utilisateurs actuellement connectés et sont assez similaires. La première affiche les utilisateurs connectés, tandis que la seconde affiche également des informations sur ce qu'ils sont en train de faire.

Exécutée sans option, la commande `who` affiche quatre colonnes correspondant à l'utilisateur connecté, au terminal, à la date et à l'heure ainsi qu'au nom d'hôte :

```
root@debian:~# who
carol pts/0 2020-06-06 17:16 (192.168.1.4)
mimi pts/1 2020-06-06 17:28 (192.168.1.4)
```

`who` accepte une série d'options parmi lesquelles nous pouvons retenir celles-ci :

-b, --boot

Afficher l'heure du dernier démarrage du système.

-r, --runlevel

Afficher le niveau d'exécution actuel.

-H, --heading

Afficher les en-têtes de colonnes.

Par rapport à `who`, `w` fournit des informations un peu plus détaillées :

```
root@debian:~# w
17:56:12 up 40 min,  2 users,  load average: 0.04, 0.12, 0.09
USER      TTY      FROM            LOGIN@   IDLE   JCPU   PCPU WHAT
carol     pts/0    192.168.1.4     17:16    1.00s  0.15s  0.05s sshd: carol [priv]
mimi      pts/1    192.168.1.4     17:28    15:08  0.05s  0.05s -bash
```

La ligne du haut affiche des informations sur l'heure actuelle (17:56:12), la durée de fonctionnement du système (up 40 min), le nombre d'utilisateurs actuellement connectés (2 users) et la charge moyenne (load average: 0.04, 0.12, 0.09). Ces valeurs correspondent à la moyenne du nombre de tâches dans la file d'attente au cours des 1, 5 et 15 dernières minutes, respectivement.

Vous voyez alors huit colonnes ; examinons-les en détail :

USER

Nom d'utilisateur.

TTY

Nom du terminal auquel l'utilisateur est connecté.

FROM

Hôte distant à partir duquel l'utilisateur s'est connecté.

LOGIN@

Heure de connexion.

IDLE

Temps d'inactivité.

JCPU

Temps utilisé par tous les processus attachés au `tty` (y compris les tâches en arrière-plan actuellement en cours d'exécution).

PCPU

Temps utilisé par le processus en cours (celui qui apparaît sous `WHAT`).

WHAT

Ligne de commande du processus en cours.

Tout comme pour `who`, vous pouvez fournir des noms d'utilisateurs en argument à `w` :

```

root@debian:~# w mimi
18:23:15 up 1:07, 2 users, load average: 0.00, 0.02, 0.05
USER      TTY      FROM          LOGIN@   IDLE   JCPU   PCPU WHAT
mimi      pts/1    192.168.1.4    17:28    9:23   0.06s  0.06s -bash

```

Configuration et utilisation de base de `sudo`

Comme nous l'avons déjà mentionné un peu plus haut, la commande `su` vous permet de passer à n'importe quel autre utilisateur du système à condition de fournir le mot de passe de l'utilisateur cible. Dans le cas de l'utilisateur `root`, le fait que son mot de passe soit distribué ou connu de (nombreux) utilisateurs met le système en péril et constitue une très mauvaise pratique en matière de sécurité. L'utilisation de base de `su` est `su - nom-d'utilisateur-cible`. Cependant, lorsque vous passez à `root`, le nom d'utilisateur cible est facultatif :

```

carol@debian:~$ su - root
Password:
root@debian:~# exit
logout
carol@debian:~$ su -
Password:
root@debian:~#

```

L'utilisation du tiret (-) garantit que l'environnement de l'utilisateur cible est chargé. Sans cela, l'environnement de l'ancien utilisateur sera conservé :

```

carol@debian:~$ su
Password:

```

```
root@debian:/home/carol#
```

D'autre part, il existe la commande `sudo`. Elle vous permet d'exécuter une commande en tant qu'utilisateur root ou n'importe quel autre utilisateur d'ailleurs. Du point de vue de la sécurité, la commande `sudo` représente une bien meilleure option que `su`, étant donné qu'elle présente deux avantages significatifs :

1. Pour exécuter une commande en tant que root, vous n'avez pas besoin du mot de passe de l'utilisateur root, mais uniquement de celui de l'utilisateur qui lance la commande, conformément à une politique de sécurité. La politique de sécurité par défaut est `sudoers`, comme spécifié dans `/etc/sudoers` et `/etc/sudoers.d/*`.
2. `sudo` vous permet d'exécuter des commandes ponctuelles avec les privilèges élevés au lieu de lancer tout un nouveau sous-shell pour root comme le fait `su`.

La syntaxe de base de `sudo` est `sudo -u nom-utilisateur-cible commande`. Par contre, pour exécuter une commande en tant qu'utilisateur root, l'option `-u nom-utilisateur-cible` n'est pas nécessaire :

```
carol@debian:~$ sudo -u mimi whoami
mimi
carol@debian:~$ sudo whoami
root
```

NOTE

`sudoers` utilise un horodatage par utilisateur (et par terminal) pour la mise en cache des informations d'identification, ce qui vous permet d'utiliser `sudo` sans mot de passe pendant une période par défaut de quinze minutes. Cette valeur par défaut peut être modifiée en ajoutant l'option `timestamp_timeout` comme paramètre `Defaults` dans `/etc/sudoers` (par exemple : `Defaults timestamp_timeout=1` définira le délai d'expiration du cache des informations d'identification à une minute).

Le fichier `/etc/sudoers`

Le fichier de configuration principal de `sudo` est `/etc/sudoers` (il y a aussi le répertoire `/etc/sudoers.d`). C'est là que sont définis les privilèges `sudo` des utilisateurs. Autrement dit, c'est là que vous spécifiez qui peut exécuter quelles commandes en tant que quels utilisateurs sur quelles machines, ainsi que d'autres paramètres. Voici la syntaxe utilisée :

```
carol@debian:~$ sudo less /etc/sudoers
(...)
```

```
# User privilege specification
root    ALL=(ALL:ALL) ALL

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL

(...)
```

La spécification des privilèges pour l'utilisateur `root` est `ALL=(ALL:ALL) ALL`. Cela veut dire que l'utilisateur `root` (`root`) peut se connecter depuis tous les hôtes (`ALL`), en tant que tous les utilisateurs et tous les groupes (`(ALL:ALL)`), et exécuter toutes les commandes (`ALL`). C'est la même chose pour les membres du groupe `sudo` — notez que les noms de groupe sont précédés d'un signe pourcentage (%).

Ainsi, pour que l'utilisatrice `carol` puisse vérifier l'état du service `apache2` depuis n'importe quel hôte en tant que n'importe quel utilisateur ou groupe, vous ajouterez la ligne suivante dans le fichier `sudoers` :

```
carol    ALL=(ALL:ALL) /usr/bin/systemctl status apache2
```

Vous voulez peut-être éviter à `carol` d'avoir à fournir son mot de passe pour exécuter la commande `systemctl status apache2`. Pour ce faire, il suffit de modifier la ligne comme ceci :

```
carol    ALL=(ALL:ALL) NOPASSWD: /usr/bin/systemctl status apache2
```

Admettons que vous souhaitiez désormais limiter vos hôtes à `192.168.1.7` et permettre à `carol` d'exécuter `systemctl status apache2` en tant qu'utilisatrice `mimi`. Vous modifieriez alors la ligne comme ceci :

```
carol    192.168.1.7=(mimi) /usr/bin/systemctl status apache2
```

Vous pouvez désormais vérifier l'état du serveur web Apache en tant qu'utilisatrice `mimi` :

```
carol@debian:~$ sudo -u mimi systemctl status apache2
• apache2.service - The Apache HTTP Server
  Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
  Active: active (running) since Tue 2020-06-09 13:12:19 CEST; 29min ago
(...)
```

Si `carol` est promue administratrice système et que vous souhaitez lui accorder tous les

privilèges, la solution la plus simple consiste à l'ajouter au groupe spécial `sudo` à l'aide de la commande `usermod` et de l'option `-G` (songez également à utiliser l'option `-a` qui garantit que l'utilisateur ne sera supprimé d'aucun autre groupe auquel il appartient) :

```
root@debian:~# sudo usermod -aG sudo carol
```

NOTE

Dans la famille des distributions Red Hat, le groupe `wheel` est l'équivalent du groupe administratif spécial `sudo` des systèmes Debian.

Au lieu de modifier directement le fichier `/etc/sudoers`, il vaut mieux utiliser la commande `visudo` en tant que `root` (par exemple : `visudo`). Elle ouvrira le fichier `/etc/sudoers` à l'aide de votre éditeur de texte prédéfini. Pour modifier l'éditeur de texte par défaut, vous pouvez ajouter l'option `editor` comme paramètre `Defaults` dans le fichier `/etc/sudoers`. Par exemple, pour changer l'éditeur et utiliser `nano`, vous ajouterez la ligne suivante :

```
Defaults    editor=/usr/bin/nano
```

NOTE

Alternativement, vous pouvez spécifier un éditeur de texte via la variable d'environnement `EDITOR` lorsque vous utilisez `visudo` (par exemple : `EDITOR=/usr/bin/nano visudo`)

En dehors des utilisateurs et des groupes, vous pouvez également utiliser des alias dans `/etc/sudoers`. Il existe trois principales catégories d'alias que vous pouvez définir : les *alias d'hôte* (`Host_Alias`), les *alias d'utilisateur* (`User_Alias`) et les *alias de commande* (`Cmnd_Alias`). Voici un exemple :

```
# Host alias specification

Host_Alias SERVERS = 192.168.1.7, server1, server2

# User alias specification

User_Alias REGULAR_USERS = john, mary, alex

User_Alias PRIVILEGED_USERS = mimi

User_Alias ADMINS = carol, %sudo, PRIVILEGED_USERS, !REGULAR_USERS

# Cmnd alias specification

Cmnd_Alias SERVICES = /usr/bin/systemctl *
```

```
# User privilege specification
root    ALL=(ALL:ALL) ALL
ADMINS  SERVERS=SERVICES

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL
```

Prenons l'exemple du fichier `sudoers` ci-dessous pour expliquer plus en détail les trois types d'alias :

Alias d'hôte

Ils incluent une liste séparée par des virgules de noms d'hôte et d'adresses IP ainsi que de réseaux et de groupes de réseaux (précédés du signe `+`). Les masques de réseau peuvent également être spécifiés. L'alias d'hôte `SERVERS` comprend une adresse IP et deux noms d'hôtes :

```
Host_Alias SERVERS = 192.168.1.7, server1, server2
```

Alias d'utilisateurs

Ils incluent une liste séparée par des virgules d'utilisateurs spécifiés sous forme de noms d'utilisateurs, de groupes (précédés de `%`) et de groupes réseau (précédés de `+`). Vous pouvez exclure certains utilisateurs à l'aide du signe `!`. L'alias utilisateur `ADMINS`, par exemple, inclut l'utilisatrice `carol`, les membres du groupe `sudo` et les membres de l'alias utilisateur `PRIVILEGE_USERS` qui n'appartiennent pas à l'alias utilisateur `REGULAR_USERS` :

```
User_Alias ADMINS = carol, %sudo, PRIVILEGED_USERS, !REGULAR_USERS
```

Alias de commande

Ils incluent une liste de commandes et de répertoires séparés par des virgules. Si un répertoire est spécifié, tous les fichiers qu'il contient seront inclus, mais les sous-répertoires seront ignorés. L'alias de commande `SERVICES` comprend une seule commande avec toutes ses sous-commandes, comme indiqué par l'astérisque (`*`) :

```
Cmnd_Alias SERVICES = /usr/bin/systemctl *
```

Conformément aux spécifications d'alias, la ligne `ADMINS SERVERS=SERVICES` dans la section `User privilege specification` se traduit comme suit : tous les utilisateurs appartenant à

ADMINS peuvent utiliser `sudo` pour exécuter n'importe quelle commande dans SERVICES sur n'importe quel serveur dans SERVERS.

NOTE

Il existe un quatrième type d'alias que vous pouvez inclure dans `/etc/sudoers` : les alias *runas* (`Runas_Alias`). Ils ressemblent beaucoup aux alias d'utilisateur, mais vous permettent de spécifier les utilisateurs par leur *ID utilisateur* (UID). Cette fonctionnalité peut s'avérer pratique dans certains cas de figure.

Exercices guidés

1. Complétez le tableau suivant relatif aux autorisations spéciales :

Autorisation spéciale	Représentation numérique	Représentation symbolique	Rechercher les fichiers dotés uniquement de ces permissions
SUID			
SGID			

2. En règle générale, ça ne sert pas à grand-chose d'afficher uniquement les fichiers avec le bit SUID ou SGID activé. Effectuez les tâches suivantes pour montrer que vos recherches peuvent être plus productives :

- Retrouvez tous les fichiers avec la permission SUID (et d'autres permissions) définies dans `/usr/bin` :

- Retrouvez tous les fichiers avec la permission SGID (et d'autres permissions) définies dans `/usr/bin` :

- Retrouvez tous les fichiers avec un bit SUID ou SGID activé dans `/usr/bin` :

3. La commande `chage` vous permet de modifier les informations relatives à l'expiration du mot de passe d'un utilisateur. En tant que root, complétez le tableau suivant en fournissant les commandes appropriées pour l'utilisatrice `mary` :

Signification	Commandes <code>chage</code>
Faire en sorte que le mot de passe soit valide pendant 365 jours.	
Obliger l'utilisatrice à changer son mot de passe lors de sa prochaine connexion.	
Définir le nombre minimum de jours entre les changements de mot de passe à 1.	
Désactiver l'expiration du mot de passe.	

Signification	Commandes chage
Permettre à l'utilisatrice de changer son mot de passe à tout moment.	
Définir la période d'avertissement sur 7 jours et la date d'expiration du compte au le 20 août 2050.	
Afficher les informations actuelles relatives à l'expiration du mot de passe de l'utilisatrice.	

4. Complétez le tableau ci-dessous avec l'outil réseau approprié :

Action	Commande(s)
Afficher les fichiers réseau pour l'hôte 192.168.1.55 sur le port 22 à l'aide de <code>lsof</code> .	
Afficher les processus qui accèdent au port par défaut du serveur web Apache sur votre machine à l'aide de <code>fuser</code> .	
Répertorier tous les sockets <code>udp</code> en écoute sur votre machine à l'aide de la commande <code>netstat</code> .	
Scanner les ports 80 à 443 sur l'hôte 192.168.1.55 à l'aide de <code>nmap</code> .	

5. Effectuez les tâches suivantes relatives à la taille du jeu résident (RSS) et à `ulimit` en tant qu'utilisateur normal :

- Afficher les limites *soft* sur le *RSS maximal* :

- Afficher les limites *hard* sur le *RSS maximal* :

- Définir les limites *soft* du *RSS maximal* à 5000 kilo-octets :

- Définir les limites *hard* du *RSS maximal* à 10000 kilo-octets :

- Pour finir, essayez d'augmenter la limite *hard* du *RSS maximal* jusqu'à 15 000 kilo-octets. Pouvez-vous le faire ? Pourquoi ?

6. Considérez le résultat suivant de la commande `last` et répondez aux questions :

```
carol    pts/0          192.168.1.4      Sun May 31 14:16 - 14:22  (00:06)
```

- Est-ce que `carol` s'est connectée depuis un hôte distant ? Pourquoi ?

- Combien de temps a duré la session de `carol` ?

- Est-ce que `carol` était connectée via un terminal texte classique ? Pourquoi ?

7. Considérez l'extrait suivant du fichier `/etc/sudoers` et répondez à la question ci-dessous.

```
# Host alias specification

Host_Alias SERVERS = 192.168.1.7, server1, server2

# User alias specification

User_Alias REGULAR_USERS = john, mary, alex

User_Alias PRIVILEGED_USERS = mimi

User_Alias ADMINS = carol, %sudo, PRIVILEGED_USERS, !REGULAR_USERS

# Cmnd alias specification

Cmnd_Alias WEB_SERVER_STATUS = /usr/bin/systemctl status apache2

# User privilege specification
root    ALL=(ALL:ALL) ALL
ADMINS  SERVERS=WEB_SERVER_STATUS

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL
```

Est-ce que alex peut vérifier l'état du serveur Web Apache sur n'importe quel hôte ? Pourquoi ?

Exercices d'approfondissement

1. En dehors des permissions SUID et SGID, il existe une troisième permission spéciale : le *sticky bit*. À l'heure actuelle, il est principalement utilisé sur des répertoires comme `/tmp` pour empêcher les utilisateurs normaux de supprimer ou de déplacer des fichiers qui ne leur appartiennent pas. Effectuez les tâches suivantes :

- Activez le *sticky bit* sur `~/temporal` :

- Trouvez les répertoires avec le *sticky bit* (et n'importe quelle autre permission) défini dans votre répertoire personnel :

- Désactivez le *sticky bit* sur `~/temporal` :

2. Lorsque le mot de passe d'un utilisateur est verrouillé via `passwd -l utilisateur` ou `usermod -L utilisateur`, comment pouvez-vous le savoir en consultant `/etc/shadow` ?

3. Quelle est la commande `usermod` équivalente à `chage -E date utilisateur` ou `chage --expiredate date utilisateur` ?

4. Indiquez deux commandes `nmap` différentes pour analyser les 65535 ports sur `localhost` :

Résumé

Dans cette leçon, vous avez appris à effectuer un certain nombre de tâches administratives liées à la sécurité. Voici les sujets qui ont été abordés :

- Rechercher les fichiers dotés des autorisations spéciales `SUID` et `SGID`.
- Définir et modifier les mots de passe des utilisateurs et gérer les informations relatives à l'expiration des mots de passe.
- Utiliser une panoplie d'outils réseau pour découvrir les ports ouverts sur les hôtes/réseaux.
- Définir des limites pour les ressources du système.
- Vérifier les utilisateurs qui se sont connectés au système ou qui sont actuellement connectés.
- Utilisation et configuration de base de `sudo` (via le fichier `/etc/sudoers`).

Voici les commandes et les fichiers que nous avons abordés dans cette leçon :

find

Rechercher des fichiers dans une arborescence de répertoires.

passwd

Modifier le mot de passe d'un utilisateur.

chmod

Modifier les permissions d'un fichier.

chage

Modifier les informations d'expiration du mot de passe d'un utilisateur.

lsof

Afficher la liste des fichiers ouverts.

fuser

Identifier les processus qui utilisent des fichiers ou des sockets.

netstat

Afficher les connexions réseau.

nmap

Outil d'exploration réseau et scanner de ports.

ulimit

Afficher et définir les limites des utilisateurs.

/etc/security/limits.conf

Fichier de configuration qui permet d'appliquer des restrictions aux utilisateurs.

last

Afficher la liste des derniers utilisateurs connectés.

lastb

Afficher la liste des tentatives de connexion échouées.

/var/log/wtmp

Base de données des connexions des utilisateurs.

who

Afficher les utilisateurs connectés.

w

Afficher les utilisateurs connectés et ce qu'ils sont en train de faire.

su

Changer d'utilisateur ou devenir super-utilisateur.

sudo

Exécuter une commande avec les privilèges d'un autre utilisateur (y compris le super-utilisateur).

/etc/sudoers

Fichier de configuration par défaut pour la politique de sécurité **sudo**.

Réponses aux exercices guidés

1. Complétez le tableau suivant relatif aux autorisations spéciales :

Autorisation spéciale	Représentation numérique	Représentation symbolique	Rechercher les fichiers dotés uniquement de ces permissions
SUID	4000	s, S	<code>find -perm 4000,</code> <code>find -perm u+s</code>
SGID	2000	s, S	<code>find -perm 2000,</code> <code>find -perm g+s</code>

2. En règle générale, ça ne sert pas à grand-chose d'afficher uniquement les fichiers avec le bit SUID ou SGID activé. Effectuez les tâches suivantes pour montrer que vos recherches peuvent être plus productives :

- Retrouvez tous les fichiers avec la permission SUID (et d'autres permissions) définies dans `/usr/bin` :

```
find /usr/bin -perm -4000 ou find /usr/bin -perm -u+s
```

- Retrouvez tous les fichiers avec la permission SGID (et d'autres permissions) définies dans `/usr/bin` :

```
find /usr/bin -perm -2000 ou find /usr/bin -perm -g+s
```

- Retrouvez tous les fichiers avec un bit SUID ou SGID activé dans `/usr/bin` :

```
find /usr/bin -perm /6000
```

3. La commande `chage` vous permet de modifier les informations relatives à l'expiration du mot de passe d'un utilisateur. En tant que root, complétez le tableau suivant en fournissant les commandes appropriées pour l'utilisatrice `mary` :

Signification	Commandes <code>chage</code>
Faire en sorte que le mot de passe soit valide pendant 365 jours.	<code>chage -M 365 mary</code> , <code>chage --maxdays 365 mary</code>
Obliger l'utilisatrice à changer son mot de passe lors de sa prochaine connexion.	<code>chage -d 0 mary</code> , <code>chage --lastday 0 mary</code>

Signification	Commandes chage
Définir le nombre minimum de jours entre les changements de mot de passe à 1.	<code>chage -m 1 mary</code> , <code>chage --mindays 1 mary</code>
Désactiver l'expiration du mot de passe.	<code>chage -M 99999 mary</code> , <code>chage --maxdays 99999 mary</code>
Permettre à l'utilisatrice de changer son mot de passe à tout moment.	<code>chage -m 0 mary</code> , <code>chage --mindays 0 mary</code>
Définir la période d'avertissement sur 7 jours et la date d'expiration du compte au le 20 août 2050.	<code>chage -W 7 -E 2050-08-20 mary</code> , <code>chage --warndays 7 --expiredate 2050-08-20 mary</code>
Afficher les informations actuelles relatives à l'expiration du mot de passe de l'utilisatrice.	<code>chage -l mary</code> , <code>chage --list mary</code>

4. Complétez le tableau ci-dessous avec l'outil réseau approprié :

Action	Commande(s)
Afficher les fichiers réseau pour l'hôte 192.168.1.55 sur le port 22 à l'aide de <code>lsof</code> .	<code>lsof -i@192.168.1.55:22</code>
Afficher les processus qui accèdent au port par défaut du serveur web Apache sur votre machine à l'aide de <code>fuser</code> .	<code>fuser -vn tcp 80</code> , <code>fuser --verbose --namespace tcp 80</code>
Répertorier tous les sockets <code>udp</code> en écoute sur votre machine à l'aide de la commande <code>netstat</code> .	<code>netstat -lu</code> , <code>netstat --listening --udp</code>
Scanner les ports 80 à 443 sur l'hôte 192.168.1.55 à l'aide de <code>nmap</code> .	<code>nmap -p 80-443 192.168.1.55</code>

5. Effectuez les tâches suivantes relatives à la taille du jeu résident (RSS) et à `ulimit` en tant qu'utilisateur normal :

- Afficher les limites *soft* sur le *RSS maximal* :

```
ulimit -m, ulimit -Sm
```

- Afficher les limites *hard* sur le *RSS maximal* :

```
ulimit -Hm
```

- Définir les limites *soft* du *RSS maximal* à 5000 kilo-octets :

```
ulimit -Sm 5000
```

- Définir les limites *hard* du *RSS maximal* à 10000 kilo-octets :

```
ulimit -Hm 10000
```

- Pour finir, essayez d'augmenter la limite *hard* du *RSS maximal* jusqu'à 15 000 kilo-octets. Pouvez-vous le faire ? Pourquoi ?

Non. Une fois définies, les limites *hard* ne peuvent pas être augmentées par les utilisateurs normaux.

6. Considérez le résultat suivant de la commande `last` et répondez aux questions :

```
carol    pts/0          192.168.1.4      Sun May 31 14:16 - 14:22  (00:06)
```

- Est-ce que `carol` s'est connectée depuis un hôte distant ? Pourquoi ?

Oui, l'adresse IP de l'hôte distant apparaît dans la troisième colonne.

- Combien de temps a duré la session de `carol` ?

Six minutes (comme indiqué dans la dernière colonne).

- Est-ce que `carol` était connectée via un terminal texte classique ? Pourquoi ?

Non, `pts/0` dans la deuxième colonne indique que la connexion a été établie via un émulateur de terminal graphique (également appelé *Pseudo Terminal Slave*).

7. Considérez l'extrait suivant du fichier `/etc/sudoers` et répondez à la question ci-dessous.

```
# Host alias specification

Host_Alias SERVERS = 192.168.1.7, server1, server2

# User alias specification

User_Alias REGULAR_USERS = john, mary, alex

User_Alias PRIVILEGED_USERS = mimi
```

```
User_Alias ADMINS = carol, %sudo, PRIVILEGED_USERS, !REGULAR_USERS

# Cmnd alias specification

Cmnd_Alias WEB_SERVER_STATUS = /usr/bin/systemctl status apache2

# User privilege specification
root    ALL=(ALL:ALL) ALL
ADMINS  SERVERS=WEB_SERVER_STATUS

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL
```

Est-ce que alex peut vérifier l'état du serveur Web Apache sur n'importe quel hôte ? Pourquoi ?

Non, étant donné qu'il est membre du groupe REGULAR_USERS et que ce groupe d'utilisateurs est exclu du groupe ADMINS ; les seuls utilisateurs (à l'exception de carol, des membres du groupe sudo et de root) qui peuvent exécuter systemctl status apache2 sur les SERVERS.

Réponses aux exercices d'approfondissement

1. En dehors des permissions SUID et SGID, il existe une troisième permission spéciale : le *sticky bit*. À l'heure actuelle, il est principalement utilisé sur des répertoires comme `/tmp` pour empêcher les utilisateurs normaux de supprimer ou de déplacer des fichiers qui ne leur appartiennent pas. Effectuez les tâches suivantes :

- Activez le *sticky bit* sur `~/temporal` :

```
chmod +t temporal, chmod 1755 temporal
```

- Trouvez les répertoires avec le *sticky bit* (et n'importe quelle autre permission) défini dans votre répertoire personnel :

```
find ~ -perm -1000, find ~ -perm /1000
```

- Désactivez le *sticky bit* sur `~/temporal` :

```
chmod -t temporal, chmod 0755 temporal
```

2. Lorsque le mot de passe d'un utilisateur est verrouillé via `passwd -l utilisateur` ou `usermod -L utilisateur`, comment pouvez-vous le savoir en consultant `/etc/shadow` ?

Un point d'exclamation s'affiche dans le deuxième champ juste après le nom d'utilisateur concerné (par exemple : `mary: !$6$g0g9xJgv...`).

3. Quelle est la commande `usermod` équivalente à `chage -E date utilisateur` ou `chage --expiredate date utilisateur` ?

```
usermod -e date utilisateur, usermod --expiredate date utilisateur
```

4. Indiquez deux commandes `nmap` différentes pour analyser les 65 535 ports sur `localhost` :

```
nmap -p 1-65535 localhost et nmap -p- localhost
```



110.2 Configuration de la sécurité du système

Référence aux objectifs de LPI

LPIC-1 version 5.0, Exam 102, Objective 110.2

Valeur

3

Domaines de connaissance les plus importants

- Compréhension des mots de passe shadow et de leur fonctionnement.
- Arrêt des services inutiles.
- Compréhension du rôle des TCP wrappers.

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

- `/etc/nologin`
- `/etc/passwd`
- `/etc/shadow`
- `/etc/xinetd.d/`
- `/etc/xinetd.conf`
- `systemd.socket`
- `/etc/inittab`
- `/etc/init.d/`
- `/etc/hosts.allow`
- `/etc/hosts.deny`



110.2 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	110 Sécurité
Objectif :	110.2 Configurer la sécurité de l'hôte
Leçon :	1 sur 1

Introduction

Ce chapitre présente quatre méthodes de base pour améliorer la sécurité de l'hôte :

1. Quelques commandes de base et paramètres de configuration pour améliorer la sécurité de l'authentification avec les mots de passe cachés (*shadow passwords*).
2. Comment utiliser les super-démons pour surveiller les connexions réseau entrantes.
3. Vérifier les services réseau à la recherche de démons inutiles.
4. Les *TCP wrappers* comme une sorte de pare-feu simple.

Améliorer la sécurité de l'authentification avec les mots de passe cachés

Les composants de base des données relatives au compte d'un utilisateur sont stockés dans le fichier `/etc/passwd`. Ce fichier contient sept champs : identifiant de connexion, substitut de mot de passe, ID de l'utilisateur, ID du groupe, commentaire (également appelé GECOS), emplacement du répertoire personnel et l'interpréteur de commandes (*shell*) par défaut. Pour mémoriser

facilement l'ordre de ces champs, pensez à la procédure de connexion d'un utilisateur : vous entrez d'abord un nom d'utilisateur et un mot de passe (représenté ici par un x). Le système les associe à un ID utilisateur (UID) et à un ID groupe (GID). Une fois le champ commentaire ou GECOS renseigné, le répertoire personnel de l'utilisateur est attribué, puis l'interpréteur de commandes par défaut est spécifié.

Bien que dans les systèmes modernes, le mot de passe n'est plus stocké dans le fichier `/etc/passwd`. Au lieu de cela, le champ du mot de passe contient uniquement un x minuscule. Le fichier `/etc/passwd` doit être lisible par tous les utilisateurs. Il n'est donc pas recommandé d'y stocker les mots de passe. Le x indique que le mot de passe chiffré (haché) est en réalité stocké dans le fichier `/etc/shadow`. Ce fichier ne doit pas être lisible par tous les utilisateurs.

Les mots de passe sont configurés à l'aide des commandes `passwd` et `chage`. Ces deux commandes modifient l'entrée correspondant à l'utilisatrice `emma` dans le fichier `/etc/shadow`. En tant que super-utilisateur, vous pouvez définir le mot de passe de l'utilisatrice `emma` à l'aide de la commande suivante :

```
$ sudo passwd emma
New password:
Retype new password:
passwd: password updated successfully
```

Vous serez invité à confirmer deux fois le nouveau mot de passe.

Pour afficher la date d'expiration du mot de passe et les autres paramètres relatifs à l'expiration du mot de passe pour l'utilisatrice `emma`, utilisez :

```
$ sudo chage -l emma
Last password change                : Apr 27, 2020
Password expires                    : never
Password inactive                   : never
Account expires                    : never
Minimum number of days between password change : 0
Maximum number of days between password change : 99999
Number of days of warning before password expires : 7
```

Pour empêcher l'utilisatrice `emma` de se connecter au système, le super-utilisateur peut définir une date d'expiration du mot de passe antérieure à la date actuelle. Par exemple, si la date actuelle est le 27 mars 2020, vous pouvez faire expirer le mot de passe de l'utilisatrice en utilisant une date antérieure :

```
$ sudo chage -E 2020-03-26 emma
```

Autrement, le super-utilisateur pourra utiliser :

```
$ sudo passwd -l emma
```

pour verrouiller temporairement le compte à l'aide de l'option `-l` de la commande `passwd`. Pour tester ces changements, essayez de vous connecter avec le compte `emma` :

```
$ sudo login emma
Password:
Your account has expired; please contact your system administrator

Authentication failure
```

Pour empêcher temporairement tous les utilisateurs à l'exception de `root` de se connecter au système, le super-utilisateur peut créer un fichier nommé `/etc/nologin`. Ce fichier peut contenir un message destiné aux utilisateurs leur expliquant pourquoi ils ne peuvent pas se connecter (des notifications de maintenance du système par exemple). Pour plus de détails, consultez `man 5 nologin`. Notez qu'il existe également une commande `nologin` qui peut être utilisée pour empêcher la connexion lorsqu'elle est définie comme interpréteur de commandes par défaut pour un utilisateur. Par exemple :

```
$ sudo usermod -s /sbin/nologin emma
```

Consultez `man 8 nologin` pour en savoir plus.

Utiliser un super-démon pour surveiller les connexions réseau entrantes

Les services réseau comme les serveurs web, les serveurs de messagerie et les serveurs d'impression fonctionnent généralement comme des services autonomes qui écoutent sur un port réseau dédié. Tous ces services autonomes fonctionnent en parallèle. Sur les systèmes classiques basés sur Sys-V-init, chacun de ces services peut être contrôlé par la commande `service`. Sur les systèmes modernes basés sur `systemd`, on utilise `systemctl` pour gérer le service.

Dans le bon vieux temps, les ressources informatiques disponibles étaient beaucoup plus limitées. Il n'était pas envisageable d'exécuter simultanément plusieurs services en mode autonome. Au

lieu de cela, on utilisait un super-démon pour surveiller les connexions réseau entrantes et démarrer le service approprié à la demande. Cette méthode pour établir une connexion réseau prenait un peu plus de temps. Les super-démons les plus connus sont `inetd` et `xinetd`. Sur les systèmes actuels basés sur `systemd`, l'unité `systemd.socket` peut être utilisée de manière similaire. Dans cette section, nous utiliserons `xinetd` pour intercepter les connexions vers le démon `sshd` et démarrer ce démon à la demande pour montrer comment le super-démon était utilisé.

Avant de configurer le service `xinetd`, quelques réglages préalables sont nécessaires. Peu importe que vous utilisiez un système basé sur Debian ou Red Hat. Ces explications ont été testées avec Debian/GNU Linux 9.9, mais elles devraient fonctionner sur n'importe quel système Linux moderne qui utilise `systemd` sans nécessiter de changements significatifs. Assurez-vous tout d'abord que les paquets `openssh-server` et `xinetd` sont installés. Vérifiez ensuite que le service SSH fonctionne :

```
$ systemctl status sshd
• ssh.service - OpenBSD Secure Shell server
  Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
  Active: active (running) since Mon 2020-04-27 09:33:48 EDT; 3h 11min ago
    Docs: man:sshd(8)
          man:sshd_config(5)
  Process: 430 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
 Main PID: 460 (sshd)
    Tasks: 1 (limit: 1119)
  Memory: 5.3M
  CGroup: /system.slice/ssh.service
          └─460 /usr/sbin/sshd -D
```

Vérifiez également que le service SSH écoute sur son port réseau standard 22 :

```
# lsof -i :22
COMMAND PID USER  FD  TYPE  DEVICE SIZE/OFF NODE NAME
sshd    1194 root   3u   IPv4 16053268      0t0  TCP *:ssh (LISTEN)
sshd    1194 root   4u   IPv6 16053270      0t0  TCP *:ssh (LISTEN)
```

Enfin, arrêtez le service SSH :

```
$ sudo systemctl stop sshd.service
```

Si vous souhaitez rendre cette modification permanente pour la conserver après le redémarrage,

utilisez la commande `systemctl disable sshd.service`.

Vous pouvez maintenant créer le fichier de configuration `xinetd` `/etc/xinetd.d/ssh` avec quelques paramètres de base :

```
service ssh
{
    disable      = no
    socket_type  = stream
    protocol     = tcp
    wait         = no
    user         = root
    server       = /usr/sbin/sshd
    server_args  = -i
    flags        = IPv4
    interface    = 192.168.178.1
}
```

Redémarrez le service `xinetd` avec :

```
$ sudo systemctl restart xinetd.service
```

Vérifiez quel service écoute actuellement les connexions SSH entrantes.

```
$ sudo lsof -i :22
COMMAND  PID USER  FD  TYPE  DEVICE  SIZE/OFF  NODE  NAME
xinetd   24098 root   5u   IPv4  7345141      0t0   TCP 192.168.178.1:ssh (LISTEN)
```

Nous pouvons voir que le service `xinetd` a pris le contrôle de l'accès au port 22.

Voici quelques détails supplémentaires à propos de la configuration de `xinetd`. Le fichier de configuration principal est `/etc/xinetd.conf` :

```
# Simple configuration file for xinetd
#
# Some defaults, and include /etc/xinetd.d/

defaults
{
    # Please note that you need a log_type line to be able to use log_on_success
```

```
# and log_on_failure. The default is the following :
# log_type = SYSLOG daemon info

}

includedir /etc/xinetd.d
```

En dehors des paramètres par défaut, une seule directive permet de définir un répertoire d'inclusion. Dans ce répertoire, vous pouvez configurer un fichier de configuration unique pour chaque service que vous souhaitez gérer avec `xinetd`. Nous l'avons fait ci-dessus pour le service SSH en nommant le fichier `/etc/xinetd.d/ssh`. Les noms des fichiers de configuration peuvent être choisis arbitrairement, à l'exception des noms de fichiers contenant un point (.) ou se terminant par un tilde (~). Cela dit, il est courant de nommer le fichier d'après le service que vous souhaitez configurer.

Certains fichiers de configuration dans le répertoire `/etc/xinetd.d/` sont déjà fournis par la distribution :

```
$ ls -l /etc/xinetd.d
total 52
-rw-r--r-- 1 root root 640 Feb  5  2018 chargen
-rw-r--r-- 1 root root 313 Feb  5  2018 chargen-udp
-rw-r--r-- 1 root root 502 Apr 11 10:18 daytime
-rw-r--r-- 1 root root 313 Feb  5  2018 daytime-udp
-rw-r--r-- 1 root root 391 Feb  5  2018 discard
-rw-r--r-- 1 root root 312 Feb  5  2018 discard-udp
-rw-r--r-- 1 root root 422 Feb  5  2018 echo
-rw-r--r-- 1 root root 304 Feb  5  2018 echo-udp
-rw-r--r-- 1 root root 312 Feb  5  2018 servers
-rw-r--r-- 1 root root 314 Feb  5  2018 services
-rw-r--r-- 1 root root 569 Feb  5  2018 time
-rw-r--r-- 1 root root 313 Feb  5  2018 time-udp
```

Ces fichiers peuvent être utilisés comme modèles dans les rares cas où vous devez utiliser certains services hérités tels que `daytime`, une implémentation très ancienne d'un serveur de temps. Tous ces fichiers modèles contiennent la directive `disable = yes`.

Voici quelques détails supplémentaires à propos des directives utilisées dans le fichier d'exemple `/etc/xinetd.d/ssh` pour SSH ci-dessus.

```
service ssh
```

```
{
    disable      = no
    socket_type  = stream
    protocol     = tcp
    wait        = no
    user         = root
    server       = /usr/sbin/sshd
    server_args  = -i
    flags       = IPv4
    interface    = 192.168.178.1
}
```

service

Indique le service que xinetd doit contrôler. Vous pouvez utiliser soit un numéro de port comme 22, soit le nom associé au numéro de port dans `/etc/services`, par exemple `ssh`.

```
{
```

Les paramètres détaillés commencent par une accolade ouvrante.

disable

Pour activer ces paramètres, mettez la valeur `no`. Si vous voulez désactiver temporairement les paramètres, vous pouvez mettre `yes`.

socket_type

Vous pouvez choisir `stream` pour les sockets TCP ou `dgram` pour les sockets UDP et plus encore.

protocol

Choisissez TCP ou UDP.

wait

Pour les connexions TCP, cette valeur est généralement réglée sur `no`.

user

Le service démarré ici sera la propriété de cet utilisateur.

server

Chemin complet vers le service qui est censé être lancé par xinetd.

server_args

Vous pouvez ajouter des options pour le service ici. S'ils sont démarrés par un super-serveur, la plupart des services requièrent une option particulière. Pour SSH, ce serait l'option `-i`.

flags

Vous pouvez choisir IPv4, IPv6 et d'autres.

interface

L'interface réseau que `xinetd` doit contrôler. Note : vous pouvez également choisir la directive `bind`, qui n'est qu'un synonyme de `interface`.

}

Terminez par une accolade fermante.

Les successeurs des services démarrés par le super-serveur `xinetd` sont les unités de socket `systemd`. La configuration d'une unité de socket `systemd` est extrêmement simple et rapide, étant donné qu'il existe déjà une unité de socket `systemd` prédéfinie pour SSH. Assurez-vous que les services `xinetd` et SSH ne sont pas en cours d'exécution.

Il ne vous reste plus qu'à démarrer l'unité de socket SSH :

```
$ sudo systemctl start ssh.socket
```

Pour vérifier quel service écoute à présent sur le port 22, nous utilisons encore une fois `lsof`. Notez ici que nous avons utilisé l'option `-P` pour afficher le numéro de port au lieu du nom du service dans le résultat :

```
$ sudo lsof -i :22 -P
COMMAND PID USER  FD   TYPE    DEVICE  SIZE/OFF  NODE NAME
systemd   1 root   57u  IPv6 14730112      0t0  TCP *:22 (LISTEN)
```

Pour terminer cette session, essayez de vous connecter à votre serveur à l'aide d'un client SSH de votre choix.

TIP

Si `systemctl start ssh.socket` ne fonctionne pas avec votre distribution, essayez `systemctl start sshd.socket`.

Vérifier les services pour détecter les démons inutiles

Pour des raisons de sécurité et pour gérer les ressources du système, il est important d'avoir une vue d'ensemble des services en cours d'exécution. Les services inutiles doivent être désactivés. Par exemple, si vous n'avez pas besoin de servir des pages web, il n'est pas nécessaire de faire tourner un serveur web comme Apache ou nginx.

Sur les systèmes basés sur Sys-V-init, vous pouvez vérifier l'état de tous les services comme ceci :

```
$ sudo service --status-all
```

Vérifiez si chacun des services répertoriés dans le résultat de la commande est nécessaire et (pour les systèmes basés sur Debian) désactivez tous les services inutiles avec :

```
$ sudo update-rc.d SERVICE-NAME remove
```

Sur les systèmes basés sur Red Hat, vous utiliseriez plutôt :

```
$ sudo chkconfig SERVICE-NAME off
```

Sur les systèmes modernes basés sur systemd, nous pouvons utiliser la commande suivante pour afficher la liste de tous les services en cours d'exécution :

```
$ systemctl list-units --state active --type service
```

Vous pouvez ensuite désactiver chaque unité de service inutile à l'aide de l'option :

```
$ sudo systemctl disable UNIT --now
```

Cette commande va arrêter le service et le supprimer de la liste des services pour éviter qu'il ne se lance au prochain démarrage du système.

Par ailleurs, pour avoir un aperçu des services réseau en écoute, vous pouvez utiliser `netstat` sur les systèmes plus anciens (à condition d'avoir installé le paquet `net-tools`) :

```
$ netstat -ltu
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp      0      0 0.0.0.0:ssh              0.0.0.0:*               LISTEN
tcp      0      0 localhost:mysql          0.0.0.0:*               LISTEN
tcp6     0      0 [::]:http               [::]:*                  LISTEN
tcp6     0      0 [::]:ssh                 [::]:*                  LISTEN
udp      0      0 0.0.0.0:bootpc           0.0.0.0:*
```

Sur les systèmes modernes, vous pouvez également utiliser la commande équivalente `ss` (pour

“_socket services_”):

```
$ ss -ltn
Netid      State      Recv-Q     Send-Q     Local Address:Port      Peer
Address:Port
udp        UNCONN     0           0           0.0.0.0:bootpc
0.0.0.0:*
tcp        LISTEN     0           128         0.0.0.0:ssh
0.0.0.0:*
tcp        LISTEN     0           80          127.0.0.1:mysql
0.0.0.0:*
tcp        LISTEN     0           128         *:http
*:*
tcp        LISTEN     0           128         [::]:ssh
[::]:*
```

Les TCP Wrappers comme une sorte de pare-feu simple

À l’époque où les pare-feu sous Linux n’existaient pas, on utilisait les TCP wrappers pour sécuriser les connexions réseau vers un hôte. De nos jours, la plupart des programmes ne respectent plus les TCP wrappers. Dans les distributions récentes basées sur Red Hat (Fedora 29 par exemple), la prise en charge des TCP wrappers a été complètement supprimée. Cependant, il est utile d’avoir quelques connaissances de base sur cette technologie afin de pouvoir gérer les systèmes Linux historiques qui utilisent encore les TCP wrappers.

Nous allons encore une fois utiliser le service SSH comme exemple de base. Le service sur notre hôte exemple devrait être accessible depuis le réseau local uniquement. Tout d’abord, nous vérifions si le démon SSH utilise la bibliothèque `libwrap` qui fournit le support des TCP wrappers :

```
$ ldd /usr/sbin/sshd | grep "libwrap"
libwrap.so.0 => /lib/x86_64-linux-gnu/libwrap.so.0 (0x00007f91dbec0000)
```

Maintenant nous ajoutons la ligne suivante dans le fichier `/etc/hosts.deny` :

```
sshd: ALL
```

Enfin, nous ajoutons une exception pour les connexions SSH depuis le réseau local dans le fichier `/etc/hosts.allow` :

```
sshd: LOCAL
```

Les changements prennent effet immédiatement, sans qu'il soit nécessaire de redémarrer un service quelconque. Vous pouvez en avoir le cœur net avec le client `ssh`.

Exercices guidés

1. Comment le compte verrouillé `emma` peut-il être déverrouillé ?

2. Jusqu'à présent, le compte `emma` avait une date d'expiration. Comment peut-on fixer la date d'expiration à `never` (c'est-à-dire *jamais*) ?

3. Imaginez que le service d'impression CUPS, qui gère les travaux d'impression, ne soit pas nécessaire sur votre serveur. Comment désactiver ce service une fois pour toutes ? Comment vérifier que le port approprié n'est plus actif ?

4. Vous avez installé le serveur web `nginx`. Comment pouvez-vous vérifier si `nginx` prend en charge les TCP wrappers ?

Exercices d'approfondissement

1. Vérifiez si l'existence du fichier `/etc/nologin` empêche la connexion de l'utilisateur `root`.
2. Est-ce que l'existence du fichier `/etc/nologin` empêche les connexions sans mot de passe avec les clés SSH ?
3. Que se passe-t-il lors de la connexion lorsque le fichier `/etc/nologin` contient la seule ligne de texte `login currently is not possible` ?
4. Est-ce qu'une utilisatrice normale `emma` peut obtenir des informations sur l'utilisateur `root` contenues dans le fichier `/etc/passwd`, par exemple avec la commande `grep root /etc/passwd` ?
5. Est-ce qu'une utilisatrice normale `emma` peut récupérer des informations sur son propre mot de passe haché enregistré dans le fichier `/etc/shadow`, par exemple avec la commande `grep -i emma /etc/shadow` ?
6. Quelles sont les étapes à suivre pour activer et vérifier que le service obsolète `daytime` est pris en charge par `xinetd` ? Notez qu'il s'agit là d'un exercice purement théorique. Ne faites pas ça dans un environnement de production.

Résumé

Voici ce que nous avons appris dans cette leçon :

1. Dans quel fichier sont stockés les mots de passe ainsi que certains paramètres de sécurité relatifs aux mots de passe, par exemple le délai d'expiration.
2. La fonction du super-démon `xinetd` et comment le mettre en marche et lancer le service `sshd` à la demande.
3. Vérifier quels sont les services réseau en cours d'exécution et comment désactiver les services inutiles.
4. Utiliser les TCP wrappers comme une sorte de pare-feu simple.

Commandes utilisées dans les travaux pratiques et les exercices :

chage

Modifier la durée de vie du mot de passe d'un utilisateur.

chkconfig

Une commande classique initialement utilisée sur les systèmes Red Hat pour déterminer si un service doit être lancé au démarrage ou non.

netstat

Un outil classique (désormais intégré dans le paquet `net-tools`) qui affiche les démons qui accèdent aux ports réseau d'un système et leur utilisation.

nologin

Une commande qui peut être utilisée à la place de l'interpréteur de commandes d'un utilisateur pour l'empêcher de se connecter.

passwd

Permet de définir ou modifier le mot de passe d'un utilisateur.

service

Ancienne manière de contrôler l'état d'un démon, comme l'arrêt ou le démarrage d'un service.

ss

L'équivalent moderne de `netstat`, qui affiche également des informations sur les différents sockets utilisés sur le système.

systemctl

La commande de contrôle du système utilisée pour contrôler divers aspects des services et des sockets sur un ordinateur avec systemd.

update-rc.d

Une commande classique similaire à `chkconfig` qui active ou désactive le lancement d'un service au démarrage sur les distributions basées sur Debian.

xinetd

Un super-démon qui peut contrôler l'accès à un service réseau à la demande, laissant ainsi un service inactif jusqu'à ce qu'il soit effectivement sollicité pour effectuer une tâche.

Réponses aux exercices guidés

1. Comment le compte verrouillé `emma` peut-il être déverrouillé ?

Le super-utilisateur peut exécuter `passwd -u emma` pour déverrouiller le compte.

2. Jusqu'à présent, le compte `emma` avait une date d'expiration. Comment peut-on fixer la date d'expiration à `never` (c'est-à-dire *jamais*) ?

Le super-utilisateur peut utiliser `chage -E -1 emma` pour mettre la date d'expiration à `never`. Ce réglage peut être vérifié avec `chage -l emma`.

3. Imaginez que le service d'impression CUPS, qui gère les travaux d'impression, ne soit pas nécessaire sur votre serveur. Comment désactiver ce service une fois pour toutes ? Comment vérifier que le port approprié n'est plus actif ?

En tant que super-utilisateur, invoquez :

```
systemctl disable cups.service --now
```

Vérifiez :

```
netstat -l | grep ":ipp " ou `ss -l | grep ":ipp "
```

4. Vous avez installé le serveur web `nginx`. Comment pouvez-vous vérifier si `nginx` prend en charge les TCP wrappers ?

La commande

```
ldd /usr/sbin/nginx | grep "libwrap"
```

va afficher un résultat si `nginx` prend en charge les TCP wrappers.

Réponses aux exercices d'approfondissement

1. Vérifiez si l'existence du fichier `/etc/nologin` empêche la connexion de l'utilisateur `root`.

L'utilisateur `root` est toujours capable de se connecter.

2. Est-ce que l'existence du fichier `/etc/nologin` empêche les connexions sans mot de passe avec les clés SSH ?

Oui, les connexions sans mot de passe sont également bloquées.

3. Que se passe-t-il lors de la connexion lorsque le fichier `/etc/nologin` contient la seule ligne de texte `login currently is not possible` ?

Le message `login currently is not possible` sera affiché, et vous ne pourrez pas vous connecter.

4. Est-ce qu'une utilisatrice normale `emma` peut obtenir des informations sur l'utilisateur `root` contenues dans le fichier `/etc/passwd`, par exemple avec la commande `grep root /etc/passwd` ?

Oui, étant donné que tous les utilisateurs ont le droit de lire ce fichier.

5. Est-ce qu'une utilisatrice normale `emma` peut récupérer des informations sur son propre mot de passe haché enregistré dans le fichier `/etc/shadow`, par exemple avec la commande `grep -i emma /etc/shadow` ?

Non, parce que les utilisateurs normaux n'ont pas le droit de lire ce fichier.

6. Quelles sont les étapes à suivre pour activer et vérifier que le service obsolète `daytime` est pris en charge par `xinetd` ? Notez qu'il s'agit là d'un exercice purement théorique. Ne faites pas ça dans un environnement de production.

Dans un premier temps, modifiez le fichier `/etc/xinetd.d/daytime` et mettez la directive `disable = no`. Ensuite, redémarrez le service `xinetd` avec `systemctl restart xinetd.service` (ou `service xinetd restart` sur les systèmes avec Sys-V-Init). Maintenant vous pouvez vérifier si ça marche avec `nc localhost daytime`. Au lieu de `nc`, vous pouvez également utiliser `netcat`.



110.3 Sécurisation des données avec le chiffrement

Référence aux objectifs de LPI

LPIC-1 version 5.0, Exam 102, Objective 110.3

Valeur

4

Domaines de connaissance les plus importants

- Configuration élémentaire et utilisation des clients OpenSSH2.
- Compréhension du rôle des clés du serveur hôte OpenSSH.
- Utilisation et configuration de base de GnuPG, y compris la révocation des clés.
- Utilisation de GPG pour chiffrer, déchiffrer, signer et vérifier des fichiers.
- Compréhension des tunnels SSH (y compris les tunnels X11).

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

- `ssh`
- `ssh-keygen`
- `ssh-agent`
- `ssh-add`
- `~/.ssh/id_rsa` et `id_rsa.pub`
- `~/.ssh/id_dsa` et `id_dsa.pub`
- `~/.ssh/id_ecdsa` et `id_ecdsa.pub`
- `~/.ssh/id_ed25519` et `id_ed25519.pub`
- `/etc/ssh/ssh_host_rsa_key` et `ssh_host_rsa_key.pub`

- `/etc/ssh/ssh_host_dsa_key` et `ssh_host_dsa_key.pub`
- `/etc/ssh/ssh_host_ecdsa_key` et `ssh_host_ecdsa_key.pub`
- `/etc/ssh/ssh_host_ed25519_key` et `ssh_host_ed25519_key.pub`
- `~/.ssh/authorized_keys`
- `ssh_known_hosts`
- `gpg`
- `gpg-agent`
- `~/.gnupg/`



110.3 Leçon 1

Certification :	LPIC-1
Version :	5.0
Thème :	110 Sécurité
Objectif :	110.3 Sécuriser les données avec le chiffrement
Leçon :	1 sur 2

Introduction

La sécurisation des données par le chiffrement est d'une importance primordiale dans de nombreux aspects de l'administration des systèmes modernes, et plus encore lorsqu'il s'agit d'accéder à distance à des systèmes. Contrairement aux solutions non sécurisées comme *telnet*, *rlogin* ou *FTP*, le protocole *SSH (Secure Shell)* a été conçu dans un esprit de sécurité. Il utilise la cryptographie à clé publique pour authentifier les hôtes et les utilisateurs et pour chiffrer tous les échanges d'informations qui s'ensuivent. Par ailleurs, SSH peut être utilisé pour établir des *tunnels de ports*, ce qui permet, entre autres, à un protocole non chiffré de transmettre des données via une connexion SSH chiffrée. La version actuelle et recommandée du protocole SSH est la 2.0. *OpenSSH* est une implémentation libre et open source du protocole SSH.

Cette leçon aborde la configuration de base du client *OpenSSH* ainsi que le rôle des clés d'hôte du serveur *OpenSSH*. Le concept des tunnels de ports SSH sera également abordé. Nous utiliserons deux machines avec la configuration suivante :

Rôle de la machine	OS	Adresse IP	Nom d'hôte	Utilisateur
Client	Debian GNU/Linux 10 (buster)	192.168.1.55	debian	carol
Serveur	openSUSE Leap 15.1	192.168.1.77	halof	ina

Configuration et utilisation de base du client OpenSSH

Le serveur et le client OpenSSH sont livrés dans des paquets distincts, mais on peut généralement installer un méta-paquet qui va fournir les deux en même temps. Pour établir une session à distance avec le serveur SSH, vous utilisez la commande `ssh`, en spécifiant l'utilisateur que vous voulez connecter sur la machine distante et l'adresse IP ou le nom d'hôte de la machine distante. La première fois que vous vous connectez à un hôte distant, vous allez recevoir un message comme celui-ci :

```
carol@debian:~$ ssh ina@192.168.1.77
The authenticity of host '192.168.1.77 (192.168.1.77)' can't be established.
ECDSA key fingerprint is SHA256:5JF7anupYipByCQm2BPvDHRVFJJixeslmppi2NwATYI.
Are you sure you want to continue connecting (yes/no)?
```

Une fois que vous avez tapé `yes` et appuyé sur Entrée, vous devrez saisir le mot de passe de l'utilisateur distant. S'il est tapé correctement, un message d'avertissement s'affiche, et vous voilà connecté à l'hôte distant :

```
Warning: Permanently added '192.168.1.77' (ECDSA) to the list of known hosts.
Password:
Last login: Sat Jun 20 10:52:45 2020 from 192.168.1.4
Have a lot of fun...
ina@halof:~>
```

Les messages sont assez explicites : comme c'était la première fois que vous établissez une connexion avec le serveur distant 192.168.1.77, son authenticité n'a pas pu être vérifiée dans une base de données. Le serveur distant a donc fourni une *empreinte ECDSA* de sa clé publique (en utilisant la fonction de hachage SHA256). Une fois la connexion acceptée, la clé publique du serveur distant est ajoutée à la base de données des hôtes connus (*known hosts*), ce qui permet d'authentifier le serveur pour les connexions futures. Cette liste des clés publiques des hôtes

connus est conservée dans le fichier `known_hosts` qui se trouve dans `~/.ssh` :

```
ina@halof:~> exit
logout
Connection to 192.168.1.77 closed.
carol@debian:~$ ls .ssh/
known_hosts
```

Tant `.ssh` que `known_hosts` ont été créés après l'établissement de la première connexion à distance. `~/.ssh` est le répertoire par défaut pour la configuration utilisateur et les informations d'authentification.

NOTE

Vous pouvez également utiliser `ssh` pour exécuter une seule commande sur l'hôte distant et revenir ensuite à votre terminal local (par exemple : lancer `ssh ina@halof ls`).

Si vous utilisez le même utilisateur sur la machine locale et distante, il n'est pas nécessaire de spécifier le nom d'utilisateur lors de l'établissement de la connexion SSH. Par exemple, si vous êtes connectée en tant qu'utilisatrice `carol` sur `debian` et que vous voulez vous connecter à `halof` également en tant qu'utilisatrice `carol`, il vous suffit de taper `ssh 192.168.1.77` ou `ssh halof` (si le nom peut être résolu) :

```
carol@debian:~$ ssh halof
Password:
Last login: Wed Jul  1 23:45:02 2020 from 192.168.1.55
Have a lot of fun...
carol@halof:~>
```

Admettons maintenant que vous établissez une nouvelle connexion à distance avec un hôte qui a la même adresse IP que `halof` (ce qui est courant si vous utilisez le DHCP dans votre réseau local). Vous serez averti de la possibilité d'une attaque par interception (attaque MITM ou *Man In The Middle Attack*) :

```
carol@debian:~$ ssh john@192.168.1.77
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@  WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!  @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that a host key has just been changed.
The fingerprint for the ECDSA key sent by the remote host is
```

```

SHA256:KH4q3vP6C7e0SEjyG8WlZ9fVlf+jmWJ5139RBxBh3TY.
Please contact your system administrator.
Add correct host key in /home/carol/.ssh/known_hosts to get rid of this message.
Offending ECDSA key in /home/carol/.ssh/known_hosts:1
  remove with:
    ssh-keygen -f "/home/carol/.ssh/known_hosts" -R "192.168.1.77"
ECDSA host key for 192.168.1.77 has changed and you have requested strict checking.
Host key verification failed.

```

Puisque vous n'avez pas affaire à une attaque par interception, vous pouvez sereinement ajouter l'empreinte de la clé publique du nouvel hôte à `.ssh/known_hosts`. Comme l'indique le message, vous pouvez d'abord utiliser la commande `ssh-keygen -f /home/carol/.ssh/known_hosts -R 192.168.1.77` pour supprimer la clé incriminée (vous pouvez aussi utiliser `ssh-keygen -R 192.168.1.77` pour supprimer toutes les clés appartenant à 192.168.1.77 de `~/.ssh/known_hosts`). Ensuite, vous allez pouvoir établir une connexion avec le nouvel hôte.

Authentification par clé SSH

Vous pouvez configurer votre client SSH de manière à ce qu'il utilise des clés publiques au lieu du mot de passe à l'ouverture de la session. C'est la méthode préférée pour se connecter à un serveur distant via SSH, étant donné qu'elle est beaucoup plus sûre. La première chose à faire est de créer une paire de clés sur la machine client. Pour ce faire, vous utiliserez `ssh-keygen` avec l'option `-t` en spécifiant le type de chiffrement que vous souhaitez, dans notre cas l'algorithme de signature numérique à courbe elliptique (*Elliptic Curve Digital Signature Algorithm*). Ensuite, on vous demandera le chemin pour enregistrer la paire de clés (`~/.ssh/` est pratique et c'est l'emplacement par défaut) et une phrase de passe. Cette dernière est facultative, mais il est fortement recommandé d'en utiliser une.

```

carol@debian:~/.ssh$ ssh-keygen -t ecdsa
Generating public/private ecdsa key pair.
Enter file in which to save the key (/home/carol/.ssh/id_ecdsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/carol/.ssh/id_ecdsa.
Your public key has been saved in /home/carol/.ssh/id_ecdsa.pub.
The key fingerprint is:
SHA256:tlamD0SaTquPZYdNepwj8XN4xvqmHCbe8g5FKKUfMo8 carol@debian
The key's randomart image is:
+---[ECDSA 256]---+
|      .           |
|     o .          |
|    = o o         |

```

```
|      B *      |
|    E B S o    |
|    o & O      |
|    @ ^ =      |
|    *.@ @.     |
|    o.o+B+o    |
+-----[SHA256]-----+
```

NOTE

Lorsque vous créez la paire de clés, vous pouvez passer l'option `-b` à `ssh-keygen` pour spécifier la taille de la clé en bits (par exemple : `ssh-keygen -t ecdsa -b 521`).

La commande précédente a généré deux nouveaux fichiers dans votre répertoire `~/.ssh` :

```
carol@debian:~/.ssh$ ls
id_ecdsa id_ecdsa.pub known_hosts
```

id_ecdsa

C'est votre clé privée.

id_ecdsa.pub

C'est votre clé publique.

NOTE

Dans la cryptographie asymétrique (ou cryptographie à clé publique), les clés publique et privée sont mathématiquement liées l'une à l'autre de telle sorte que ce qui est chiffré par l'une ne peut être déchiffré que par l'autre.

La prochaine chose à faire est d'ajouter votre clé publique au fichier `~/.ssh/authorized_keys` de l'utilisateur que vous voulez connecter sur l'hôte distant (si le répertoire `~/.ssh` n'existe pas encore, vous devrez d'abord le créer). Vous pouvez copier votre clé publique sur le serveur distant de plusieurs façons : en utilisant une clé USB, via la commande `scp` — qui va transférer le fichier en utilisant SSH — ou en utilisant `cat` et `ssh` sur le contenu de votre clé publique comme ceci :

```
carol@debian:~/.ssh$ cat id_ecdsa.pub | ssh ina@192.168.1.77 'cat >> .ssh/authorized_keys'
Password:
```

Une fois que votre clé publique a été ajoutée au fichier `authorized_keys` sur l'hôte distant, vous pouvez être confronté à deux scénarios lorsque vous essayez d'établir une nouvelle connexion :

- Si vous n'avez pas fourni de phrase de passe lors de la création de la paire de clés, vous serez

connecté automatiquement. Même si cette méthode est pratique, elle peut s'avérer peu sécurisée selon le contexte :

```
carol@debian:~$ ssh ina@192.168.1.77
Last login: Thu Jun 25 20:31:03 2020 from 192.168.1.55
Have a lot of fun...
ina@halof:~>
```

- Si vous avez fourni une phrase de passe lors de la création de la paire de clés, vous devrez la saisir à chaque connexion, un peu comme un mot de passe. En dehors de la clé publique, cette méthode ajoute une couche de sécurité supplémentaire sous forme de phrase de passe et peut donc être considérée comme plus sûre. Cependant, en termes de confort, c'est exactement la même chose que de devoir taper un mot de passe à chaque connexion. Si vous n'utilisez pas de phrase de passe et que quelqu'un parvient à obtenir votre fichier de clé SSH privée, il aura accès à tous les serveurs sur lesquels votre clé publique est installée.

```
carol@debian:~/.ssh$ ssh ina@192.168.1.77
Enter passphrase for key '/home/carol/.ssh/id_ecdsa':
Last login: Thu Jun 25 20:39:30 2020 from 192.168.1.55
Have a lot of fun...
ina@halof:~>
```

Il existe une méthode qui combine la sécurité et le confort : l'utilisation de l'*agent d'authentification SSH* (`ssh-agent`). L'agent d'authentification crée son propre interpréteur de commandes (*shell*) et garde en mémoire vos clés privées — pour l'authentification par clé publique — jusqu'à la fin de la session. Voyons comment cela fonctionne un peu plus en détail :

1. Utilisez `ssh-agent` pour lancer un nouveau interpréteur de commandes Bash :

```
carol@debian:~/.ssh$ ssh-agent /bin/bash
carol@debian:~/.ssh$
```

2. Utilisez la commande `ssh-add` pour ajouter votre clé privée à une zone sécurisée de la mémoire. Si vous avez fourni une phrase de passe lors de la génération de la paire de clés — ce qui est recommandé pour plus de sécurité — elle vous sera demandée :

```
carol@debian:~/.ssh$ ssh-add
Enter passphrase for /home/carol/.ssh/id_ecdsa:
Identity added: /home/carol/.ssh/id_ecdsa (carol@debian)
```

Une fois votre identité ajoutée, vous pouvez vous connecter à n'importe quel serveur distant sur lequel votre clé publique est présente sans avoir à retaper la phrase de passe. Sur les ordinateurs de bureau modernes, il est courant d'exécuter cette commande au démarrage de l'ordinateur, étant donné qu'elle restera en mémoire jusqu'à ce que l'ordinateur soit éteint (ou que la clé soit déchargée manuellement).

Pour terminer cette section, voyons les quatre types d'algorithmes de clé publique qui peuvent être spécifiés avec `ssh-keygen` :

RSA

Nommé d'après ses créateurs Ron Rivest, Adi Shamir et Leonard Adleman, il a été publié en 1977. Il est considéré comme sûr et encore largement utilisé aujourd'hui. La taille minimale de la clé est de 1024 bits (2048 par défaut).

DSA

L'algorithme *Digital Signature Algorithm* s'est avéré peu sûr et a été supprimé à partir d'OpenSSH 7.0. Les clés DSA doivent avoir une longueur exacte de 1024 bits.

ecdsa

L'algorithme de signature numérique à courbe elliptique (*Elliptic Curve Digital Signature Algorithm*) est une amélioration du DSA et — peut donc — être considéré comme plus sûr. Il utilise la cryptographie à courbe elliptique. La longueur de la clé ECDSA est déterminée par l'une des trois tailles possibles de la courbe elliptique en bits : 256, 384 ou 521.

ed25519

C'est une implémentation de EdDSA — *Edwards-curve Digital Signature Algorithm* — qui utilise la courbe 25519 plus robuste. Il est considéré comme le plus sûr de tous. Toutes les clés Ed25519 ont une longueur fixe de 256 bits.

NOTE En l'absence du paramètre `-t`, `ssh-keygen` va générer une paire de clés RSA par défaut.

Fonction des clés d'hôte du serveur OpenSSH

Le répertoire de configuration globale d'OpenSSH se trouve dans l'arborescence `/etc` :

```
halof:~ # tree /etc/ssh
/etc/ssh
├── moduli
├── ssh_config
└── ssh_host_dsa_key
```

```

├─ ssh_host_dsa_key.pub
├─ ssh_host_ecdsa_key
├─ ssh_host_ecdsa_key.pub
├─ ssh_host_ed25519_key
├─ ssh_host_ed25519_key.pub
├─ ssh_host_rsa_key
├─ ssh_host_rsa_key.pub
└─ sshd_config

```

0 directories, 11 files

En dehors de `moduli` et des fichiers de configuration pour le client (`ssh_config`) et pour le serveur (`sshd_config`), vous y trouverez quatre paires de clés—une paire pour chaque algorithme supporté—qui sont créées lorsque le serveur *OpenSSH* est installé. Comme nous l'avons déjà noté, le serveur utilise ces *clés d'hôte* pour s'identifier auprès des clients selon la demande. Voici leur schéma de nommage :

Clés privées

préfixe `ssh_host_` + *algorithme* + suffixe `key` (par exemple : `ssh_host_rsa_key`)

Clés publiques (ou empreintes de clés publiques)

préfixe `ssh_host_` + *algorithme* + suffixe `key.pub` (par exemple : `ssh_host_rsa_key.pub`)

NOTE

Une empreinte (*fingerprint*) est créée en appliquant une fonction de hachage cryptographique à une clé publique. Les empreintes sont plus courtes que les clés auxquelles elles se réfèrent, ce qui permet de simplifier la gestion des clés.

Les permissions sur les fichiers qui contiennent les clés privées sont `0600` ou `-rw-----` : le propriétaire (`root`) est le seul à pouvoir les lire et les écrire. D'autre part, tous les fichiers de clés publiques sont également lisibles par les membres du groupe propriétaire et tous les autres (`0644` ou `-rw-r--r--`) :

```

halof:~ # ls -l /etc/ssh/ssh_host_*
-rw----- 1 root root 1381 Dec 21 20:35 /etc/ssh/ssh_host_dsa_key
-rw-r--r-- 1 root root  605 Dec 21 20:35 /etc/ssh/ssh_host_dsa_key.pub
-rw----- 1 root root  505 Dec 21 20:35 /etc/ssh/ssh_host_ecdsa_key
-rw-r--r-- 1 root root  177 Dec 21 20:35 /etc/ssh/ssh_host_ecdsa_key.pub
-rw----- 1 root root  411 Dec 21 20:35 /etc/ssh/ssh_host_ed25519_key
-rw-r--r-- 1 root root   97 Dec 21 20:35 /etc/ssh/ssh_host_ed25519_key.pub
-rw----- 1 root root 1823 Dec 21 20:35 /etc/ssh/ssh_host_rsa_key
-rw-r--r-- 1 root root  397 Dec 21 20:35 /etc/ssh/ssh_host_rsa_key.pub

```

Vous pouvez voir les empreintes des clés en passant l'option `-l` à `ssh-keygen`. Vous devez également fournir l'option `-f` pour spécifier le chemin vers le fichier de clé :

```
halof:~ # ssh-keygen -l -f /etc/ssh/ssh_host_ed25519_key
256 SHA256:8cnPrinC49ZHc+/9Ai5pV+1JfZ4WBRZhd3rD0sc2z1A root@halof (ED25519)
halof:~ # ssh-keygen -l -f /etc/ssh/ssh_host_ed25519_key.pub
256 SHA256:8cnPrinC49ZHc+/9Ai5pV+1JfZ4WBRZhd3rD0sc2z1A root@halof (ED25519)
```

Pour afficher l'empreinte de la clé ainsi que son dessin aléatoire, il suffit d'ajouter l'option `-v` comme ceci :

```
halof:~ # ssh-keygen -lv -f /etc/ssh/ssh_host_ed25519_key.pub
256 SHA256:8cnPrinC49ZHc+/9Ai5pV+1JfZ4WBRZhd3rD0sc2z1A root@halof (ED25519)
+--[ED25519 256]--+
|                 +oo|
|                 .+o.|
|                 ..E.|
|                 + . +.o|
|                 S + + *o|
|                 ooo 0o=|
|                 . . . =o+.==|
|                 = o =oo o=o|
|                 o.o +o+..o.+|
+-----[SHA256]-----+
```

Tunnels de ports SSH

OpenSSH propose une fonctionnalité puissante de redirection de port, qui permet de faire transiter —et de chiffrer— le trafic d'un port source à travers un processus SSH, avant de le rediriger vers un port sur une machine de destination. Ce mécanisme, appelé *tunnel de port* ou *redirection de port*, présente des avantages majeurs :

- Il permet de contourner les pare-feux pour accéder à des ports sur des machines distantes.
- Il autorise l'accès depuis l'extérieur vers une machine de votre réseau privé.
- Il assure le chiffrement de tous les échanges de données.

De manière générale, on distingue deux types de tunnels de port : les redirections locales et les redirections distantes.

Tunnel de port local

Vous définissez un port local pour rediriger le trafic vers la machine de destination via le processus SSH, qui fait office d'intermédiaire. Ce processus SSH peut s'exécuter sur la machine locale ou sur un serveur distant. Par exemple, si vous souhaitez créer un tunnel vers `www.gnu.org` en utilisant le port 8585 sur votre machine locale, vous procéderiez comme suit :

```
carol@debian:~$ ssh -L 8585:www.gnu.org:80 debian
carol@debian's password:
Linux debian 4.19.0-9-amd64 #1 SMP Debian 4.19.118-2 (2020-04-29) x86_64

The programs included with the Debian GNU/Linux system are free software;
(...)
Last login: Sun Jun 28 13:47:27 2020 from 127.0.0.1
```

Explication : avec l'option `-L`, nous spécifions que le port local 8585 doit être redirigé vers le port 80 (HTTP) de `www.gnu.org` via le processus SSH s'exécutant sur `debian` (notre machine locale). Nous aurions pu utiliser la commande `ssh -L 8585:www.gnu.org:80 localhost` avec le même résultat. Si vous ouvrez maintenant un navigateur web et accédez à <http://localhost:8585>, vous serez redirigé vers `www.gnu.org`. Pour illustrer cela, utilisons `lynx` (le navigateur web classique en mode texte) :

```
carol@debian:~$ lynx http://localhost:8585
(...)
* Back to Savannah Homepage
  * Not Logged in
  * Login
  * New User
  * This Page
  * Language
  * Clean Reload
  * Printer Version
  * Search
  * -
(...)
```

Si vous souhaitez faire exactement la même chose, mais en passant par un processus SSH s'exécutant sur `halof`, vous utiliseriez la commande suivante :

```
carol@debian:~$ ssh -L 8585:www.gnu.org:80 -Nf ina@192.168.1.77
Enter passphrase for key '/home/carol/.ssh/id_ecdsa':
```

```
carol@debian:~$
carol@debian:~$ lynx http://localhost:8585
(...)
* Back to Savannah Homepage
  * Not Logged in
  * Login
  * New User
  * This Page
  * Language
  * Clean Reload
  * Printer Version
  * Search
  * _
(...)
```

Trois détails importants dans cette commande :

- Grâce à l'option `-N`, nous n'ouvrons pas de session interactive sur `halof` ; nous effectuons uniquement la redirection de port.
- L'option `-f` indique à SSH de s'exécuter en arrière-plan.
- Nous avons spécifié l'utilisatrice `ina` pour effectuer la redirection : `ina@192.168.1.77`.

Tunnel de port distant

Dans le cas d'un tunnel de port distant, le trafic arrivant sur un port du serveur distant est redirigé vers le processus SSH s'exécutant sur votre machine locale, puis de là vers le port spécifié sur la machine de destination (qui peut également être votre machine locale). Par exemple, si vous souhaitez permettre à quelqu'un en dehors de votre réseau d'accéder au serveur web Apache de votre machine locale via le port 8585 du serveur SSH exécuté sur `halof` (192.168.1.77), vous utiliserez la commande suivante :

```
carol@debian:~$ ssh -R 8585:localhost:80 -Nf ina@192.168.1.77
Enter passphrase for key '/home/carol/.ssh/id_ecdsa':
carol@debian:~$
```

Désormais, toute personne qui établit une connexion sur `halof` sur le port 8585 de l'interface `localhost` verra la page d'accueil par défaut d'Apache2 de Debian :

```
carol@halof:~$ lynx localhost:8585
(...)
```

Apache2 Debian Default

Page: It works (p1 of 3)

Debian Logo Apache2 Debian Default Page

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after

installation on Debian systems. If you can read this page, it means that the Apache HTTP server

installed at this site is working properly. You should replace this file (located at /var/www/html/index.html) before continuing to operate your HTTP server.

(...)

Pour autoriser l'hôte distant `halof` à créer un tunnel sur toutes ses interfaces réseau pour un port spécifique, il est nécessaire de s'assurer que le paramètre `GatewayPorts yes` est configuré dans le fichier `sshd_config`. Par défaut, ce paramètre est défini sur `no`, ce qui limite la redirection à l'interface `localhost` uniquement.

```
carol@debian:~$ ssh -R 0.0.0.0:8585:localhost:80 -Nf ina@192.168.1.77
Enter passphrase for key '/home/carol/.ssh/id_ecdsa':
carol@debian:~$
```

Désormais, toute personne qui se connecte à `halof` sur le port 8585, quelle que soit l'interface IP utilisée, verra la page d'accueil par défaut d'Apache2 de Debian :

```
carol@debian:~$ lynx 192.168.1.77:8585
(...)
```

Apache2 Debian Default

Page: It works (p1 of 3)

Debian Logo Apache2 Debian Default Page

It works!

(...)

NOTE

Il existe un troisième type de redirection de port, plus complexe et hors du cadre de cette leçon : la *redirection de port dynamique*. Contrairement aux redirections locales ou distantes, ce type de redirection utilise divers flux TCP sur une plage de ports, plutôt que de cibler un seul port.

Tunnels X11

Maintenant que vous maîtrisez les tunnels de ports, terminons cette leçon en abordant le tunnel X11 (ou *X11 Forwarding*). Grâce à un tunnel X11, le système graphique *X Window* de l'hôte distant est redirigé vers votre machine locale. Pour cela, il suffit d'invoquer `ssh` avec l'option `-X` :

```
carol@debian:~$ ssh -X ina@halof
...
```

Vous pouvez désormais lancer une application graphique comme le navigateur `firefox` avec le résultat suivant : l'application s'exécute sur le serveur distant, mais son affichage est redirigé vers votre machine locale.

Si vous démarrez une nouvelle session SSH avec l'option `-x`, le *X11 Forwarding* sera désactivé. Essayez de lancer `firefox` dans ces conditions, et vous obtiendrez une erreur similaire à celle-ci :

```
carol@debian:~$ ssh -x ina@halof
carol@192.168.0.106's password:
(...)
ina@halof:~$ firefox

(firefox-esr:1779): Gtk-WARNING **: 18:45:45.603: Locale not supported by C library.
    Using the fallback 'C' locale.
Error: no DISPLAY environment variable specified
```

NOTE

Les trois directives de configuration liées respectivement à la redirection de port locale, la redirection de port distante et le *X11 Forwarding* sont `AllowTcpForwarding`, `GatewayPorts` et `X11Forwarding`. Pour en savoir plus, consultez `man ssh_config` et, ou `man sshd_config`.

Exercices guidés

1. Connectée en tant qu'utilisatrice `sonya` sur votre machine client, effectuez les tâches SSH suivantes sur le serveur distant `halof` :

- Exécutez la commande pour afficher le contenu de `~/ .ssh` en tant qu'utilisatrice `serena` sur l'hôte distant ; puis revenez à votre terminal local.

- Connectez-vous en tant qu'utilisatrice `serena` sur l'hôte distant.

- Connectez-vous en tant qu'utilisatrice `sonya` sur l'hôte distant.

- Supprimez toutes les clés appartenant à `halof` de votre fichier local `~/ .ssh/known_hosts`.

- Sur votre machine client, créez une paire de clés `ecdsa` de 256 bits.

- Sur votre machine client, créez une paire de clés `ed25519` de 256 bits.

2. Mettez les étapes suivantes dans le bon ordre pour établir une connexion SSH en utilisant l'agent d'authentification SSH :

- Sur le client, lancez un nouveau interpréteur de commandes Bash pour l'agent d'authentification avec `ssh-agent /bin/bash`.
- Sur le client, créez une paire de clés en utilisant `ssh-keygen`.
- Sur le client, ajoutez votre clé privée à une zone sécurisée de la mémoire avec `ssh-add`.
- Ajoutez la clé publique de votre client au fichier `~/ .ssh/authorized_keys` de l'utilisateur avec lequel vous voulez vous connecter sur l'hôte distant.
- S'il n'existe pas déjà, créez `~/ .ssh` pour l'utilisateur avec lequel vous voulez vous connecter sur le serveur.
- Connectez-vous au serveur distant.

L'ordre correct est :

Étape 1 :	
Étape 2 :	
Étape 3 :	
Étape 4 :	
Étape 5 :	
Étape 6 :	

3. Dans un contexte de redirection de port, quelle option et quelle directive sont utilisées pour les types de tunnels suivants :

Type de tunnel	Option	Directive
Local		
Distant ou inversé		
X		

4. Admettons que vous tapez la commande `ssh -L 8888:localhost:80 -Nf ina@halof` dans le terminal de votre machine client. Toujours sur la machine client, vous dirigez un navigateur vers `http://localhost:8888`. Quel résultat obtiendrez-vous ?

Exercices d'approfondissement

1. Pour ce qui est des directives de sécurité SSH :

- Quelle directive est utilisée dans `/etc/ssh/sshd_config` pour activer les connexions `root` :

- Quelle directive pourriez-vous utiliser dans `/etc/ssh/sshd_config` pour spécifier que seulement les comptes locaux peuvent accepter les connexions SSH :

2. Lorsque vous utilisez le même utilisateur sur le client et le serveur, quelle commande `ssh` pouvez-vous utiliser pour transférer la clé publique du client sur le serveur de sorte que vous puissiez vous connecter par clé publique ?

3. Créez deux tunnels de ports locaux en une seule commande, en redirigeant les ports locaux non privilégiés 8080 et 8585 via le serveur distant `halof` vers les sites web `www.gnu.org` et `www.meltpa.org`, respectivement. Utilisez le compte `ina` sur le serveur distant et n'oubliez pas d'utiliser les options `-Nf` :

Résumé

Dans cette leçon, nous avons abordé *OpenSSH 2* qui utilise le protocole *Secure Shell* pour chiffrer les communications entre le serveur et le client. Voici ce que vous avez appris :

- Comment se connecter à un serveur distant.
- Comment exécuter des commandes à distance.
- Comment créer des paires de clés.
- Comment se connecter avec une clé.
- Comment utiliser l'agent d'authentification pour plus de sécurité et de confort.
- Les algorithmes de clés publiques supportés par *OpenSSH* : `RSA`, `DSA`, `ecdsa`, `ed25519`.
- Le rôle des clés d'hôte *OpenSSH*.
- Comment créer des tunnels de port : local, distant et X.

Voici les commandes que nous avons abordées dans cette leçon :

ssh

Se connecter ou exécuter des commandes sur une machine distante.

ssh-keygen

Générer, gérer et convertir les clés d'authentification.

ssh-agent

Agent d'authentification OpenSSH.

ssh-add

Ajoute des identités de clés privées à l'agent d'authentification.

Réponses aux exercices guidés

1. Connectée en tant qu'utilisatrice `sonya` sur votre machine client, effectuez les tâches SSH suivantes sur le serveur distant `halof` :

- Exécutez la commande pour afficher le contenu de `~/ .ssh` en tant qu'utilisatrice `serena` sur l'hôte distant ; puis revenez à votre terminal local.

```
ssh serena@halof ls .ssh
```

- Connectez-vous en tant qu'utilisatrice `serena` sur l'hôte distant.

```
ssh serena@halof
```

- Connectez-vous en tant qu'utilisatrice `sonya` sur l'hôte distant.

```
ssh halof
```

- Supprimez toutes les clés appartenant à `halof` de votre fichier local `~/ .ssh/known_hosts`.

```
ssh-keygen -R halof
```

- Sur votre machine client, créez une paire de clés `ecdsa` de 256 bits.

```
ssh-keygen -t ecdsa -b 256
```

- Sur votre machine client, créez une paire de clés `ed25519` de 256 bits.

```
ssh-keygen -t ed25519
```

2. Mettez les étapes suivantes dans le bon ordre pour établir une connexion SSH en utilisant l'agent d'authentification SSH :

- Sur le client, lancez un nouveau shell Bash pour l'agent d'authentification avec `ssh-agent /bin/bash`.
- Sur le client, créez une paire de clés en utilisant `ssh-keygen`.
- Sur le client, ajoutez votre clé privée à une zone sécurisée de la mémoire avec `ssh-add`.

- Ajoutez la clé publique de votre client au fichier `~/ .ssh/authorized_keys` de l'utilisateur avec lequel vous voulez vous connecter sur l'hôte distant.
- S'il n'existe pas déjà, créez `~/ .ssh` pour l'utilisateur avec lequel vous voulez vous connecter sur le serveur.
- Connectez-vous au serveur distant.

Voici l'ordre correct :

Étape 1 :	Sur le client, créez une paire de clés en utilisant <code>ssh-keygen</code> .
Étape 2 :	S'il n'existe pas déjà, créez <code>~/ .ssh</code> pour l'utilisateur avec lequel vous voulez vous connecter sur le serveur.
Étape 3 :	Ajoutez la clé publique de votre client au fichier <code>~/ .ssh/authorized_keys</code> de l'utilisateur avec lequel vous voulez vous connecter sur l'hôte distant.
Étape 4 :	Sur le client, lancez un nouveau interpréteur de commandes Bash pour l'agent d'authentification avec <code>ssh-agent /bin/bash</code> .
Étape 5 :	Sur le client, ajoutez votre clé privée à une zone sécurisée de la mémoire avec <code>ssh-add</code> .
Étape 6 :	Connectez-vous au serveur distant.

3. Dans un contexte de redirection de port, quelle option et quelle directive sont utilisées pour les types de tunnels suivants :

Type de tunnel	Option	Directive
Local	<code>-L</code>	<code>AllowTcpForwarding</code>
Distant ou inversé	<code>-R</code>	<code>GatewayPorts</code>
X	<code>-X</code>	<code>X11Forwarding</code>

4. Admettons que vous tapez la commande `ssh -L 8888:localhost:80 -Nf ina@halof` dans le terminal de votre machine client. Toujours sur la machine client, vous dirigez un navigateur vers `http://localhost:8888`. Quel résultat obtiendrez-vous ?

La page d'accueil du serveur web de halof, étant donné que localhost est interprété du point de vue du serveur.

Réponses aux exercices d'approfondissement

1. Pour ce qui est des directives de sécurité SSH :

- Quelle directive est utilisée dans `/etc/ssh/sshd_config` pour activer les connexions `root` :

`PermitRootLogin`

- Quelle directive pourriez-vous utiliser dans `/etc/ssh/sshd_config` pour spécifier que seulement les comptes locaux peuvent accepter les connexions SSH :

`AllowUsers`

2. Lorsque vous utilisez le même utilisateur sur le client et le serveur, quelle commande `ssh` pouvez-vous utiliser pour transférer la clé publique du client sur le serveur de sorte que vous puissiez vous connecter par clé publique ?

`ssh-copy-id`

3. Créez deux tunnels de ports locaux en une seule commande, en redirigeant les ports locaux non privilégiés 8080 et 8585 via le serveur distant `halof` vers les sites web `www.gnu.org` et `www.melpa.org`, respectivement. Utilisez le compte `ina` sur le serveur distant et n'oubliez pas d'utiliser les options `-Nf` :

`ssh -L 8080:www.gnu.org:80 -L 8585:www.melpa.org:80 -Nf ina@halof`



110.3 Leçon 2

Certification :	LPIC-1
Version :	5.0
Thème :	110 Sécurité
Objectif :	110.3 Sécuriser les données avec le chiffrement
Leçon :	2 sur 2

Introduction

Dans la précédente leçon, nous avons appris à utiliser *OpenSSH* pour chiffrer les connexions à distance ainsi que tout échange d'informations ultérieur. Il existe d'autres scénarios dans lesquels vous souhaitez chiffrer des fichiers ou des e-mails pour qu'ils parviennent à leur destinataire en toute sécurité et à l'abri des regards indiscrets. Vous pourrez également avoir besoin de signer numériquement ces fichiers ou ces messages pour éviter qu'ils ne soient falsifiés.

Un excellent outil pour ce genre d'utilisation est *GNU Privacy Guard* (alias *GnuPG* ou simplement *GPG*), une implémentation libre et open source de l'outil propriétaire *Pretty Good Privacy* (*PGP*). *GPG* utilise le standard *OpenPGP* tel qu'il a été défini par le *OpenPGP Working Group* de l'IETF (*Internet Engineering Task Force*) dans la RFC 4880. Dans cette leçon, nous allons voir les bases de *GNU Privacy Guard*.

Configuration de base de GnuPG, utilisation et révocation

Tout comme pour SSH, le mécanisme sous le capot de GPG est la *cryptographie asymétrique* ou *cryptographie à clé publique*. Un utilisateur génère une paire de clés composée d'une *clé privée* et d'une *clé publique*. Les clés sont liées mathématiquement de manière à ce que ce qui est chiffré

par l'une ne puisse être déchiffré que par l'autre. Pour une communication réussie, l'utilisateur devra envoyer sa clé publique au destinataire concerné.

Configuration et utilisation de GnuPG

La commande pour travailler avec GPG est `gpg`. Vous pouvez lui fournir un certain nombre d'options pour effectuer différentes tâches. Commençons par générer une paire de clés pour l'utilisatrice `carol`. Pour ce faire, vous utiliserez la commande `gpg --gen-key` :

```
carol@debian:~$ gpg --gen-key
gpg (GnuPG) 2.2.12; Copyright (C) 2018 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

gpg: directory '/home/carol/.gnupg' created
gpg: keybox '/home/carol/.gnupg/pubring.kbx' created
Note: Use "gpg --full-generate-key" for a full featured key generation dialog.

GnuPG needs to construct a user ID to identify your key.

Real name:

(...)

```

Après vous avoir informé — entre autres choses — que le répertoire de configuration `~/.gnupg` et votre trousseau public `~/.gnupg/pubring.kbx` ont été créés, `gpg` vous invite à fournir votre nom et votre adresse e-mail :

```
(...)
Real name: carol
Email address: carol@debian
You selected this USER-ID:
    "carol <carol@debian>"

Change (N)ame, (E)mail, or (O)kay/(Q)uit?

```

Si vous êtes d'accord avec le USER- ID résultant et que vous appuyez sur `o`, on vous demandera une phrase d'authentification (qui devra être suffisamment complexe de préférence) :

Please enter the passphrase to

```
| protect your new key
|
| Passphrase: |
(...)

```

Un message final s’affiche pour vous informer de la création d’une série de fichiers ainsi que les clés elles-mêmes, et le processus de génération des clés est terminé :

```
(...)
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
gpg: /home/carol/.gnupg/trustdb.gpg: trustdb created
gpg: key 19BBEFD16813034E marked as ultimately trusted
gpg: directory '/home/carol/.gnupg/openpgp-revocs.d' created
gpg: revocation certificate stored as '/home/carol/.gnupg/openpgp-
revocs.d/D18FA0021F644CDAF57FD0F919BBEFD16813034E.rev'
public and secret key created and signed.

pub  rsa3072 2020-07-03 [SC] [expires: 2022-07-03]
     D18FA0021F644CDAF57FD0F919BBEFD16813034E
uid                               carol <carol@debian>
sub  rsa3072 2020-07-03 [E] [expires: 2022-07-03]

```

Le contenu du répertoire `~/.gnupg` (le répertoire de configuration de GPG) peut désormais être affiché :

```
carol@debian:~/.gnupg$ ls -l
total 16
drwx----- 2 carol carol 4096 Jul  3 23:34 openpgp-revocs.d
drwx----- 2 carol carol 4096 Jul  3 23:34 private-keys-v1.d
-rw-r--r--  1 carol carol 1962 Jul  3 23:34 pubring.kbx
-rw-----  1 carol carol 1240 Jul  3 23:34 trustdb.gpg

```

Voyons le rôle de chacun de ces fichiers :

openpgp-revocs.d

Le certificat de révocation créé avec la paire de clés est conservé ici. Les permissions sur ce répertoire sont assez restrictives étant donné que toute personne qui aurait accès au certificat

pourrait révoquer la clé (nous reviendrons sur la révocation de la clé dans la prochaine section).

private-keys-v1.d

Ce répertoire contient vos clés privées, les permissions sont donc restrictives.

pubring.kbx

C'est votre trousseau de clés publiques. Il contient votre propre clé publique ainsi que toutes les clés publiques importées.

trustdb.gpg

La base de données de confiance. Elle concerne le concept de *toile de confiance* (*Web of Trust* qui dépasse le cadre de cette leçon).

NOTE L'arrivée de *GnuPG 2.1* a entraîné quelques changements importants, notamment la disparition des fichiers `secring.gpg` et `pubring.gpg` au profit de `private-keys-v1.d` et `pubring.kbx`, respectivement.

Une fois que votre paire de clés a été créée, vous pouvez voir vos clés publiques avec `gpg --list-keys` — qui affichera le contenu de votre trousseau de clés publiques :

```
carol@debian:~/.gnupg$ gpg --list-keys
/home/carol/.gnupg/pubring.kbx
-----
pub   rsa3072 2020-07-03 [SC] [expires: 2022-07-03]
      D18FA0021F644CDAF57FD0F919BBEFD16813034E
uid           [ultimate] carol <carol@debian>
sub   rsa3072 2020-07-03 [E] [expires: 2022-07-03]
```

La chaîne hexadécimale `D18FA0021F644CDAF57FD0F919BBEFD16813034E` est votre *empreinte de clé publique*.

NOTE En dehors du USER-ID (`carol` dans l'exemple), on trouve aussi le KEY-ID. Le KEY-ID est constitué des 8 derniers chiffres hexadécimaux de l'empreinte de votre clé publique (`6813 034E`). Vous pouvez vérifier l'empreinte de votre clé avec la commande `gpg --fingerprint USER-ID`.

Distribution et révocation des clés

Maintenant que vous disposez de votre clé publique, vous devez l'enregistrer (ou l'*exporter*) dans un fichier afin de la mettre à disposition de vos futurs destinataires. Ils pourront alors l'utiliser

pour chiffrer des fichiers ou des messages qui vous sont destinés (puisque vous êtes le seul à posséder la clé privée, vous serez également le seul à pouvoir les déchiffrer et les lire). De même, vos destinataires l'utiliseront pour déchiffrer et vérifier vos messages et fichiers chiffrés et, ou signés. La commande à utiliser est `gpg --export` suivi du `USER-ID` et d'une redirection vers le nom du fichier résultant de votre choix :

```
carol@debian:~/.gnupg$ gpg --export carol > carol.pub.key
carol@debian:~/.gnupg$ ls
carol.pub.key  openpgp-revocs.d  private-keys-v1.d  pubring.kbx  trustdb.gpg
```

NOTE

L'ajout de l'option `-a` ou `--armor` à `gpg --export` (par exemple : `gpg --export --armor carol > carol.pub.key`) va créer un résultat ASCII armuré (au lieu du format OpenPGP binaire par défaut) qui pourra être envoyé par e-mail en toute sécurité.

Comme nous l'avons déjà noté, vous devez maintenant envoyer votre fichier de clé publique (`carol.pub.key`) au destinataire avec lequel vous souhaitez échanger des informations. Par exemple, envoyons le fichier de clé publique à `ina` sur le serveur distant `halof` en utilisant `scp` (*secure copy*) :

```
carol@debian:~/.gnupg$ scp carol.pub.key ina@halof:/home/ina/
Enter passphrase for key '/home/carol/.ssh/id_ecdsa':
carol.pub.key
100% 1740  775.8KB/s  00:00
carol@debian:~/.gnupg$
```

`ina` est maintenant en possession de `carol.pub.key`. Elle va l'utiliser pour chiffrer un fichier et l'envoyer à `carol` dans la section suivante.

NOTE

Une autre manière de distribuer les clés publiques consiste à utiliser des *serveurs de clés* : vous envoyez votre clé publique sur le serveur avec la commande `gpg --keyserver keyserver-name --send-keys KEY-ID` et d'autres utilisateurs pourront les récupérer (ou les *importer*) avec `gpg --keyserver keyserver-name --recv-keys KEY-ID`.

Pour conclure cette partie, nous allons parler de la révocation des clés. La révocation de clés sera utilisée lorsque vos clés privées ont été compromises ou retirées. La première étape consiste à créer un certificat de révocation en passant à `gpg` l'option `--gen-revoke` suivie du `USER-ID`. Vous pouvez précéder `--gen-revoke` de l'option `--output` suivie d'un nom de fichier cible pour enregistrer le certificat résultant dans un fichier (au lieu de l'afficher sur l'écran du terminal). Les

messages affichés tout au long du processus de révocation sont plutôt explicites :

```
sonya@debian:~/.gnupg$ gpg --output revocation_file.asc --gen-revoke sonya
```

```
sec  rsa3072/0989EB7E7F9F2066 2020-07-03 sonya <sonya@debian>
```

```
Create a revocation certificate for this key? (y/N) y
```

```
Please select the reason for the revocation:
```

```
0 = No reason specified
```

```
1 = Key has been compromised
```

```
2 = Key is superseded
```

```
3 = Key is no longer used
```

```
Q = Cancel
```

```
(Probably you want to select 1 here)
```

```
Your decision? 1
```

```
Enter an optional description; end it with an empty line:
```

```
> My laptop was stolen.
```

```
>
```

```
Reason for revocation: Key has been compromised
```

```
My laptop was stolen.
```

```
Is this okay? (y/N) y
```

```
ASCII armored output forced.
```

```
Revocation certificate created.
```

Please move it to a medium which you can hide away; if Mallory gets access to this certificate he can use it to make your key unusable. It is smart to print this certificate and store it away, just in case your media become unreadable. But have some caution: The print system of your machine might store the data and make it available to others!

Le certificat de révocation a été enregistré dans le fichier `revocation_file.asc` (asc pour le format ASCII) :

```
sonya@debian:~/.gnupg$ ls
```

```
openpgp-revocs.d  private-keys-v1.d  pubring.kbx  revocation_file.asc  trustdb.gpg
```

```
sonya@debian:~/.gnupg$ cat revocation_file.asc
```

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
```

```
Comment: This is a revocation certificate
```

```
iQHDBCABCgAtFiEEiIVjfDnnpieFi0wvnlcN6yLCeHEFA18ASx4PHQJzdG9sZW4g
bGFwdG9wAAoJEJ5XDesiwnhxT9YMAKkjQiMpo9Uyiy9hyvukPPSrlcmtAGLk4pKS
pLZfzA5kxa+HPQwBglAEvfNRR6VMxqXUgUGYC/IAyQQM62oNACy2PCPrxyJNgVF7
8l4mMZKvW++5ikjZwyg6WWV0+w6oroeo9qruJFjcu752p4T+9gsHVa2r+KRqcPQe
```

```
aZ65sAvsBJlcsUDZqfWUXg2kQp9mNPCdQuqvDaKRgNCHA1zbzNFzXWVd2X5RgFo5
nY+tUP8ZQA9DTQPBLPcggICmfLopMPZYB2bft5geb2mMi2oNpf9CNPdQkdcimNV
aRjqdUP9C89PwTafBQkQiONlsR/dWTFcqprG5KOWQPA7xjeMV8wretdEgsyTxqHp
v1iRzwjshiJCKBXxz7wSmQrJ40fiMDHeS4ipR0AYd08QCzm0zmcFQKikGSHGMy1
z/YRl1td6NZIKjf1TD0nTrFnRvPdsZ0lKYSArbfqNrHRBQkgirOD4JPI1tYKTffq
i0eZFx25K+fj2+0AJjvrbe4HDo5m+Q==
=umI8
-----END PGP PUBLIC KEY BLOCK-----
```

Pour révoquer la clé privée, il vous faudra fusionner le certificat avec la clé, ce qui se fait en important le fichier du certificat de révocation dans votre trousseau de clés :

```
sonya@debian:~/.gnupg$ gpg --import revocation_file.asc
gpg: key 9E570DEB22C27871: "sonya <sonya@debian>" revocation certificate imported
gpg: Total number processed: 1
gpg:    new key revocations: 1
gpg: marginals needed: 3  completes needed: 1  trust model: pgp
gpg: depth: 0  valid:   1  signed:    0  trust: 0-, 0q, 0n, 0m, 0f, 1u
gpg: next trustdb check due at 2022-07-04
```

Affichez vos clés et vous verrez que la clé a été révoquée :

```
sonya@debian:~/.gnupg$ gpg --list-keys
/home/sonya/.gnupg/pubring.kbx
pub   rsa3072 2020-07-04 [SC] [revoked: 2020-07-04]
      8885637C39E7A627858B4C2F9E570DEB22C27871
uid           [ revoked] sonya <sonya@debian>
```

Enfin, assurez-vous de mettre la clé révoquée à disposition de toute partie ayant des clés publiques associées (y compris les serveurs de clés).

Utiliser GPG pour chiffrer, déchiffrer, signer et vérifier des fichiers

Dans la précédente section, carol a envoyé sa clé publique à ina. Nous allons maintenant l'utiliser pour expliquer comment GPG peut chiffrer, déchiffrer, signer et vérifier des fichiers.

Chiffrer et déchiffrer des fichiers

Tout d'abord, ina doit importer la clé publique de carol (carol.pub.key) dans son trousseau de

clés afin de pouvoir travailler avec :

```
ina@halof:~> gpg --import carol.pub.key
gpg: /home/ina/.gnupg/trustdb.gpg: trustdb created
gpg: key 19BBEFD16813034E: public key "carol <carol@debian>" imported
gpg: Total number processed: 1
gpg:             imported: 1
ina@halof:~> gpg --list-keys
/home/ina/.gnupg/pubring.kbx
-----
pub   rsa3072 2020-07-03 [SC] [expires: 2022-07-03]
       D18FA0021F644CDAF57FD0F919BBEFD16813034E
uid           [ unknown] carol <carol@debian>
sub   rsa3072 2020-07-03 [E] [expires: 2022-07-03]
```

Ensuite, vous allez créer un fichier en y écrivant du texte et le chiffrer en utilisant `gpg` (vu que vous n'avez pas signé la clé de `carol`, on vous demandera explicitement si vous voulez utiliser cette clé) :

```
ina@halof:~> echo "This is the message ..." > unencrypted-message
ina@halof:~> gpg --output encrypted-message --recipient carol --armor --encrypt unencrypted-
message
gpg: 0227347CC92A5CB1: There is no assurance this key belongs to the named user
sub   rsa3072/0227347CC92A5CB1 2020-07-03 carol <carol@debian>
      Primary key fingerprint: D18F A002 1F64 4CDA F57F  D0F9 19BB EFD1 6813 034E
      Subkey fingerprint: 9D89 1BF9 39A4 C130 E44B  1135 0227 347C C92A 5CB1

It is NOT certain that the key belongs to the person named
in the user ID.  If you really know what you are doing,
you may answer the next question with yes.

Use this key anyway? (y/N) y
```

Détaillons la commande `gpg` :

--output encrypted-message

Spécification du nom de fichier pour la version chiffrée du fichier original (`encrypted-message` dans l'exemple).

--recipient carol

Spécification du `USER-ID` du destinataire (`carol` dans notre exemple). S'il n'est pas fourni,

GnuPG le demandera (à moins que `--default-recipient` ne soit spécifié).

--armor

Cette option produit un texte ASCII armuré, qui pourra être copié dans un e-mail.

--encrypt unencrypted-message

Spécification du nom de fichier du fichier original à chiffrer.

Vous pouvez maintenant envoyer le message chiffré `encrypted-message` à `carol` sur `debian` en utilisant `scp` :

```
ina@halof:~> scp encrypted-message carol@debian:/home/carol/
carol@debian's password:
encrypted-message                                100% 736
1.8MB/s 00:00
```

Si vous vous connectez maintenant en tant que `carol` et que vous essayez de lire le message chiffré `encrypted-message`, vous confirmerez qu'il est effectivement chiffré et — par conséquent — illisible :

```
carol@debian:~$ cat encrypted-message
-----BEGIN PGP MESSAGE-----

hQGMaWInNHZJKlyxAQv/brJ8Ubs/xya35sbv6kdRkm1C70NLxL30ueWA4mCs0Y/P
GBna6ZEUcrMEgl/rCyByj3Yq74kuiTmzxAIRUDdvHfj0Ttr0WjVAqIn/fPSfMkjk
dTxKo1i55tLJ+sJ17dGMZDcNBInBTP4U1atuN71A5w7vH+XpcesRcFQLKiS0mYtt
F7SN3/5x5J6io4ISn+b0KbJgiJNNx+Ne/ub4Uzk4NlK7tmBklyC1VRualtxcG7R9
1klBPYSld6fTdDwT1Y4MofpyILAiGMZvUR1RXauEKf70IzwC5gWU+UQPSgeCdKQu
X7QL0ZIBS0Ug2XKr01k93lmDj f8PwsRIm16n/hNe1a0BA3HMP0b60zv1gFeEsFvC
IxhUYPb+rfuNFTMEB7xIO94AAmWB9N4qknMxdDqNE8WhA728Plw6y8L2ngsp1Y15
MR4lIFDpljA/CcVh4BXVe9j0TdFDWUkrFMfaIfcPQwKLXEYJp19XYIaaEazk0s5D
W4pENN0Y0cX0KWyAYX6r0l8BF0rq/HMenQwqAVXMG3s8ATuU0eqjBbR1x1qCvRQP
CR/3V73aQwc2j5ioQmhWYpqxiro0yKX2Ar/E6rZyJtJYrq+Cuk803JoBaudknNFj
pwuRwF1amwnSZ/MZ/9kMKQ==
=g1jw
-----END PGP MESSAGE-----
```

Cependant, comme vous êtes en possession de la clé privée, vous pouvez facilement déchiffrer le message en passant à `gpg` l'option `--decrypt` suivie du chemin vers le fichier chiffré (la phrase d'authentification de la clé privée sera requise) :

```
carol@debian:~$ gpg --decrypt encrypted-message
gpg: encrypted with 3072-bit RSA key, ID 0227347CC92A5CB1, created 2020-07-03
"carol <carol@debian>"
This is the message ...
```

Vous pouvez également spécifier l'option `--output` pour enregistrer le message dans un nouveau fichier non chiffré :

```
carol@debian:~$ gpg --output unencrypted-message --decrypt encrypted-message
gpg: encrypted with 3072-bit RSA key, ID 0227347CC92A5CB1, created 2020-07-03
"carol <carol@debian>"
carol@debian:~$ cat unencrypted-message
This is the message ...
```

Signer et vérifier des fichiers

En dehors du chiffrement, GPG peut être utilisé pour signer des fichiers. L'option `--sign` est pertinente ici. Commençons par créer un nouveau message (message) et signons-le avec l'option `--sign` (la phrase de passe de votre clé privée vous sera demandée) :

```
carol@debian:~$ echo "This is the message to sign ..." > message
carol@debian:~$ gpg --output message.sig --sign message
(...)
```

Voici le détail de la commande `gpg` :

`--output message`

Spécification du nom de fichier de la version signée du fichier original (message.sig dans notre exemple).

`--sign message`

Chemin vers le fichier original.

NOTE

En utilisant `--sign`, le document est compressé puis signé. Le résultat est au format binaire.

Ensuite, nous allons transférer le fichier vers `ina` sur `halof` en utilisant `scp message.sig ina@halof:/home/ina`. De retour en tant que `ina` sur `halof`, vous pouvez désormais le vérifier en utilisant l'option `--verify` :

```
ina@halof:~> gpg --verify message.sig
gpg: Signature made Sat 04 jul 2020 14:34:41 CEST
gpg:          using RSA key D18FA0021F644CDAF57FD0F919BBEFD16813034E
gpg: Good signature from "carol <carol@debian>" [unknown]
(...)
```

Si vous voulez lire le fichier, vous devez le déchiffrer vers un nouveau fichier (message dans notre cas) avec l'option `--output` :

```
ina@halof:~> gpg --output message --decrypt message.sig
gpg: Signature made Sat 04 jul 2020 14:34:41 CEST
gpg:          using RSA key D18FA0021F644CDAF57FD0F919BBEFD16813034E
gpg: Good signature from "carol <carol@debian>" [unknown]
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: D18F A002 1F64 4CDA F57F D0F9 19BB EFD1 6813 034E
ina@halof:~> cat message
This is the message to sign ...
```

GPG-Agent

Nous allons conclure cette leçon en abordant brièvement `gpg-agent`. `gpg-agent` est le démon qui gère les clés privées pour GPG (il est lancé à la demande par `gpg`). Pour voir un résumé des options les plus utiles, lancez `gpg-agent --help` ou `gpg-agent -h` :

```
carol@debian:~$ gpg-agent --help
gpg-agent (GnuPG) 2.2.4
libgcrypt 1.8.1
Copyright (C) 2017 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <https://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.

Syntax: gpg-agent [options] [command [args]]
Secret key management for GnuPG

Options:

    --daemon                run in daemon mode (background)
    --server                run in server mode (foreground)
    --supervised            run in supervised mode
```

<code>-v, --verbose</code>	<code>verbose</code>
<code>-q, --quiet</code>	<code>be somewhat more quiet</code>
<code>-s, --sh</code>	<code>sh-style command output</code>
<code>-c, --csh</code>	<code>csh-style command output</code>
<code>(...)</code>	

NOTE | Pour en savoir plus, consultez la page de manuel `gpg-agent`.

Exercices guidés

1. Complétez le tableau en renseignant le nom de fichier approprié :

Description	Nom de fichier
Base de données de confiance	
Répertoire des certificats de révocation	
Répertoire des clés privées	
Trousseau de clés publiques	

2. Répondez aux questions suivantes :

- Quel type de cryptographie est utilisé par *GnuPG* ?

- Quels sont les deux principaux composants de la cryptographie à clé publique ?

- Quel est le KEY-ID de l'empreinte de la clé publique 07A6 5898 2D3A F3DD 43E3 DA95 1F3F 3147 FA7F 54C7 ?

- Quelle méthode permet de distribuer les clés publiques à l'échelle mondiale ?

3. Mettez les étapes suivantes dans le bon ordre pour la révocation de la clé privée :

- Mettez la clé révoquée à disposition de vos correspondants
- Créez un certificat de révocation
- Importez le certificat de révocation dans votre trousseau de clés

L'ordre correct est :

Étape 1 :	
Étape 2 :	
Étape 3 :	

4. Dans un contexte de chiffrement de fichiers, que signifie l'option `--armor` dans la commande `gpg --output encrypted-message --recipient carol --armor --encrypt`

unencrypted-message ?

Exercices d'approfondissement

1. La plupart des options de `gpg` ont une version longue et une version courte. Complétez le tableau avec la version courte correspondante :

Version longue	Version courte
<code>--armor</code>	
<code>--output</code>	
<code>--recipient</code>	
<code>--decrypt</code>	
<code>--encrypt</code>	
<code>--sign</code>	

2. Répondez aux questions suivantes concernant les exports de clés :

- Quelle commande pouvez-vous utiliser pour exporter toutes vos clés publiques vers un fichier nommé `all.key` ?

- Quelle commande pouvez-vous utiliser pour exporter toutes vos clés privées vers un fichier nommé `all_private.key` ?

3. Quelle option de `gpg` permet d'effectuer la plupart des tâches de gestion des clés dans un menu ?

4. Quelle option de `gpg` vous permet de faire une signature en clair ?

Résumé

Cette leçon a abordé *GNU Privacy Guard*, un excellent choix pour chiffrer/déchiffrer et signer/vérifier numériquement des fichiers. Voici ce que vous avez appris :

- Comment générer une paire de clés.
- Comment afficher les clés de votre trousseau.
- Le contenu du répertoire `~/ .gnupg`.
- La signification de `USER-ID` et `KEY-ID`.
- Comment distribuer des clés publiques à vos correspondants.
- Comment distribuer des clés publiques à l'échelle mondiale par l'intermédiaire de serveurs de clés.
- Comment révoquer des clés privées.
- Comment chiffrer et déchiffrer des fichiers.
- Comment signer et vérifier des fichiers.
- Les bases de *GPG-Agent*.

Voici les commandes qui ont été abordées dans cette leçon :

gpg

Outil de chiffrement et de signature *OpenPGP*.

Réponses aux exercices guidés

1. Complétez le tableau en renseignant le nom de fichier approprié :

Description	Nom de fichier
Base de données de confiance	trustdb.gpg
Répertoire des certificats de révocation	opengp-revocs.d
Répertoire des clés privées	private-keys-v1.d
Trousseau de clés publiques	pubring.kbx

2. Répondez aux questions suivantes :

- Quel type de cryptographie est utilisé par *GnuPG* ?

Cryptographie à clé publique ou cryptographie asymétrique.

- Quels sont les deux principaux composants de la cryptographie à clé publique ?

La clé publique et la clé privée.

- Quel est le KEY-ID de l'empreinte de la clé publique 07A6 5898 2D3A F3DD 43E3 DA95 1F3F 3147 FA7F 54C7 ?

FA7F 54C7

- Quelle méthode permet de distribuer les clés publiques à l'échelle mondiale ?

Les serveurs de clés.

3. Mettez les étapes suivantes dans le bon ordre pour la révocation de la clé privée :

- Mettez la clé révoquée à disposition de vos correspondants
- Créez un certificat de révocation
- Importez le certificat de révocation dans votre trousseau de clés

L'ordre correct est :

Étape 1 :	Créez un certificat de révocation
Étape 2 :	Importez le certificat de révocation dans votre trousseau de clés

Étape 3 :

Mettez la clé révoquée à disposition de vos correspondants

4. Dans un contexte de chiffrement de fichiers, que signifie l'option `--armor` dans la commande `gpg --output encrypted-message --recipient carol --armor --encrypt unencrypted-message` ?

Elle produit un résultat ASCII armuré, ce qui vous permet de copier le fichier chiffré résultant dans un e-mail.

Réponses aux exercices d'approfondissement

1. La plupart des options de `gpg` ont une version longue et une version courte. Complétez le tableau avec la version courte correspondante :

Version longue	Version courte
<code>--armor</code>	<code>-a</code>
<code>--output</code>	<code>-o</code>
<code>--recipient</code>	<code>-r</code>
<code>--decrypt</code>	<code>-d</code>
<code>--encrypt</code>	<code>-e</code>
<code>--sign</code>	<code>-s</code>

2. Répondez aux questions suivantes concernant les exports de clés :

- Quelle commande pouvez-vous utiliser pour exporter toutes vos clés publiques vers un fichier nommé `all.key` ?

```
gpg --export --output all.key ou gpg --export -o all.key
```

- Quelle commande pouvez-vous utiliser pour exporter toutes vos clés privées vers un fichier nommé `all_private.key` ?

```
gpg --export-secret-keys --output all_private.key ou gpg --export-secret-keys -o all_private.key
```

(`--export-secret-keys` peut être remplacé par `--export-secret-subkeys` avec un résultat légèrement différent — consultez `man gpg` pour plus d'informations).

3. Quelle option de `gpg` permet d'effectuer la plupart des tâches de gestion des clés dans un menu ?

```
--edit-key
```

4. Quelle option de `gpg` vous permet de faire une signature en clair ?

```
--clearsign
```

Impression

© 2026 par Linux Professional Institute: Supports de cours, “LPIC-1 (102) (Version 5.0)”.

PDF généré: 2026-03-26

Cette oeuvre est placée sous licence Creative Commons Attribution - Pas d'Utilisation Commerciale - Pas de Modification 4.0 International (CC-BY-NC-ND 4.0). Pour consulter une copie de cette licence, visitez

<https://creativecommons.org/licenses/by-nc-nd/4.0/>

Bien que le Linux Professional Institute ait fait tout son possible pour s'assurer que les informations et les instructions contenues dans cette publication soient exactes, le Linux Professional Institute décline toute responsabilité en cas d'erreurs ou d'omissions, y compris, mais sans s'y limiter, la responsabilité des dommages résultant de l'utilisation ou de la confiance accordée à cette publication. L'utilisation des informations et des instructions contenues dans cet ouvrage se fait à vos risques et périls. Si les exemples de code ou toute autre technologie décrite ou contenue dans cet ouvrage sont soumis à des licences open source ou aux droits de propriété intellectuelle de tiers, il vous incombe de veiller à ce que leur utilisation soit conforme à ces licences et/ou ces droits.

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.