Full Circle

LE MAGAZINE INDÉPENDANT DE LA COMMUNAUTÉ UBUNTU LINUX

ÉDITION SPÉCIALE

Command & Conquer

Volume Un

Jusqu'au numéro 25

Spécial Full Circle Magazine

Full Circle

LE MAGAZINE INDÉPENDANT DE LA COMMUNAUTÉ UBUNTU LINUX

Au sujet du Full Circle

Le Full Circle est un magazine gratuit, libre et indépendant, consacré à toutes les versions d'Ubuntu, qui fait partie des systèmes d'exploitation Linux. Chaque mois, nous publions des tutoriels, que nous espérons utiles, et des articles proposés par des lecteurs. Le Podcast, un complément du Full Circle, parle du magazine même, mais aussi de tout ce qui peut vous intéresser dans ce domaine.

Bienvenue dans un autre « numéro spécial »

Une autre série, une autre compilation d'articles pour plus de commodité. Voici une réédition directe de la série Command & Conquer, des numéros 15 à 25.

Veuillez garder à l'esprit la date de publication originale ; les versions actuelles du matériel et des logiciels peuvent être différentes de celles illustrées. Il convient donc de vérifier la version de votre matériel et de vos logiciels avant d'essayer d'émuler les tutoriels dans ces numéros spéciaux. Il se peut que les logiciels que vous avez installés soient plus récents ou qu'il y ait des versions plus récentes disponibles dans les dépôts de votre distribution.

Amusez-vous !

Clause de non-responsabilité :

Cette édition spéciale vous est fournie sans aucune garantie ; les auteurs et le magazine Full Circle déclinent toute responsabilité pour des pertes ou dommages éventuels si des lecteurs choisissent d'en appliquer le contenu à leurs ordinateur et matériel ou à ceux des autres.

Sommaire

COMMAND AND CONQUER

Écrit par Robert Clipsham

Le mois dernier, on vous a montré comment rester en sécurité en utilisant la ligne de commande. Maintenant que vous savez faire cela, vous devriez pouvoir commencer à bénéficier de vos nouvelles connaissances ! Au cours des prochains numéros, on va vous expliquer les bases de la gestion des fichiers, ce qui vous sera utile plus tard quand nous ferons des choses plus avancées.

La première commande dans ce numéro ne fera que prouver un argument du début de l'article du mois dernier : « Vous vous trouvez dans votre répertoire /home ». Chaque fois que vous voyez « ~ », cela veut dire votre répertoire /home. Allez-y, pour le démontrer, saisissez (dans un terminal) :

```
pwd
```

J'ai ce résultat :

```
$ pwd
/home/robert
```

Bien sûr, le résultat chez vous sera votre répertoire /home et non pas le mien. Mais à quoi ça sert ? Qu'est-ce que cela veut dire « être dans son répertoire /home » ? Toute commande que vous lancerez s'exécutera dans le répertoire dans lequel vous vous trouvez. Cela ne veut pas dire grand chose pour le moment, mais fera toute la différence plus tard. Une commande qui utilise le répertoire actuel est « ls », qui vous retournera une liste de fichiers dans le répertoire voulu ou dans le répertoire actuel si vous n'en précisez pas un autre.

Cependant, il n'est pas toujours utile d'être dans votre répertoire /home, alors on va bouger. Pour ce faire, nous utiliserons la commande cd, « change directory » (changer de répertoire).

```
$ cd ~/Documents
```

Si, maintenant, vous saisissez « pwd », vous verrez que vous êtes dans votre répertoire documents. Vous n'aviez pas besoin du « ~/ » pour cette commande, bien que cela puisse être un raccourci utile pour gagner du temps. Dans cet exemple, vous vous trouvez déjà dans votre répertoire /home et la commande « cd Documents » aurait fait l'affaire. Toutefois, si vous aviez été dans un autre répertoire, par exemple « /home/robert/Pictures/2007/December/Christmas », il vous aurait fallu pas mal de temps pour aller dans le répertoire Documents sans « ~/ ». Cela étant dit, maintenant que vous vous trouvez dans votre répertoire Documents, comment retourner dans le répertoire /home ? Il y a plusieurs façons de faire.

```
$ cd
$ cd ..
$ cd ~/
$ cd /home/robert
```

Toutes ces commandes font la même chose si vous vous trouvez dans votre répertoire Documents. « cd » sans argument vous amènera toujours dans votre répertoire /home. « cd .. » vous ramène dans le répertoire où vous étiez précédemment et donc, ici, nous sommes allés de « /home/robert/Documents » au répertoire précédent, « /home/robert ». La troisième utilise le raccourci « ~ » et vous pouvez vous en servir avec ou sans le « / » après. La commande finale utilise le chemin complet, qui vous amènera toujours vers l'emplacement précisé, à supposer qu'il existe.

Et maintenant, je vais essayer de vous faire gagner du temps ! Au lieu de taper un chemin très long tel que « ~/Pictures/2007/December/Christmas » vous pouvez en saisir juste les deux ou trois premières lettres !

```
$ cd ~/Pi<tab>
```

Le <tab> veut dire que vous devez appuyer sur la touche de tabulation et vous verrez que « Pi » est remplacé automatiquement par « Pictures ». Vous pouvez uti-

liser cette méthode avec la plupart des répertoires pour gagner du temps.

Conseils en cas de panne

Vous avez peut-être rencontré quelques problèmes avec ces commandes de base. Ne vous inquiétez surtout pas, c'est sans doute quelque chose de très simple. Le premier problème que vous avez pu rencontrer s'est sans doute présenté alors que vous essayiez de changer pour le répertoire « Documents ».

```
-bash: cd: documents: No such file
or directory. (Le fichier ou le
répertoire n'existe pas.)
```

Tout ce que vous saisissez en ligne de commande est sensible à la casse ! « Documents » et « documents » sont deux répertoires distincts aux yeux du terminal, alors il faut faire très attention aux majuscules et minuscules ! Vous pouvez avoir rencontré cette erreur aussi si vous n'avez pas de répertoire Documents parce que, par exemple, vous l'avez supprimé. L'autre erreur que vous avez rencontrée est peut-être survenue quand vous avez essayé de compléter la commande au moyen de la touche de tabulation. Si votre ordinateur a lâché un bip quand vous avez appuyé sur cette touche, cela peut vouloir dire deux choses. La première, c'est que le répertoire n'existe pas. Si le répertoire n'existe pas, l'ordinateur ne va pas pouvoir compléter la commande ! L'autre possibilité, c'est que, dans votre répertoire /home, il y a plus d'un répertoire qui commence avec Pi. Si c'est le cas, il suffit d'appuyer sur la touche tab à nouveau et vous aurez une liste de fichiers et de répertoires possibles, vous permettant de saisir quelques lettres de plus et ensuite de réappuyer sur la touche de tabulation. S'il y a vraiment beaucoup de fichiers et répertoires qui semblent possibles, vous verrez quelque chose comme :

```
Display all 388 possibilities? (y or
n)
```

(Afficher toutes les 388 possibilités ? (y - pour yes/oui - ou n - pour no/non))

À moins de vouloir toutes les voir, tapez « n », puis Entrée et tapez encore quelques lettres pour réduire le nombre des résultats possibles.

Robert Clipsham est un geek qui s'assume, dont les hobbys comprennent : la programmation et l'écriture de scripts, le chat sur IRC et ne pas écrire ses articles dans les temps.

Goal
GetDeb extends the existing software options for Ubuntu (and derived) Linux distributions by providing major updates and software not yet available on the official Ubuntu repositories.

Quality
GetDeb packages are built using Debian/Ubuntu building rules, this reduces development effort and assures the same level of quality. However when new packages are developed or major upgrades are performed we do not follow a strict quality assurance process, this is the accepted cost required to achieve shorter release times. Still with a broader user base problems are quickly identified as fixed. It should also be noted that we do not provide system core packages or major libraries which could cause dependency problems or other major issues, in case you find a broken package recovery should be easily achieved by reinstalling the Ubuntu official package.

www.getdeb.net

COMMAND & CONQUER

Écrit par Robert Clipsham

Maintenant que vous savez piloter la ligne de commande, repérer où vous êtes et lister les fichiers dans le répertoire courant, nous allons vous montrer comment gérer vos fichiers à l'aide d'un shell. Pour commencer, ouvrez un terminal et tapez les commandes suivantes :

```
$ touch foo
$ mkdir bar
```

La première de ces commandes créera un fichier vide nommé « foo ». C'est souvent utile si vous faites tourner un serveur web avec des applications web et qu'on vous demande de créer un fichier avec un nom précis pour prouver que vous avez la permission d'installer une application. La deuxième commande est utilisée pour créer des répertoires, ici on en crée un nommé « bar ». Si vous voulez être sûr que ces commandes ont fonctionné, vous pouvez utiliser la commande que vous avez apprise dans le numéro précédent pour voir la liste des fichiers et répertoires du répertoire courant (non je ne vous dirai pas de quelle commande il s'agit !). Maintenant que nous avons un fichier et un répertoire pour nous entraîner, il est temps les utiliser pour faire quelque chose. La première chose que je vais vous montrer est comment copier un fichier ou un répertoire.

```
$ cp foo foo2
```

Ceci copiera le fichier « foo » (que nous avons créé plus tôt) en « foo2 ». Maintenant, déplaçons le nouveau fichier « foo2 » dans le répertoire « bar » et en même temps renommons-le « foo ». Pour cela, utilisons l'outil mv.

```
$ mv foo2 bar/foo
```

Notez comment vous pouvez utiliser l'outil mv pour renommer des fichiers, et aussi pour les déplacer. Si vous vouliez simplement déplacer « foo2 » dans le répertoire « bar » sans le renommer, vous pourriez utiliser l'une des deux commandes suivantes :

```
$ mv foo2 bar/
$ mv foo2 bar/foo2
```

Puisque vous ne le renommez pas, vous n'avez pas besoin d'ajouter le nom du fichier au répertoire, mais vous pouvez si vous voulez être sûr de ce que vous faites. Vous devez faire attention en utilisant mv et cp, car ces commandes écraseront tout fichier existant. Ce n'est pas un problème avec nos fichiers de test, mais quand vous manipulez de vrais fichiers cela pourrait poser des problèmes. Vous devriez faire une sauvegarde avant d'agir sur les fichiers avec la ligne de commande si vous avez peur de perdre des fichiers. Si vous voulez être prévenu quand votre commande va écraser un fichier, utilisez l'argument -i. Par exemple :

```
$ cp -i foo bar/
```

Si vous avez fait « mv foo2 bar/foo » précédemment, vous verrez une question demandant de confirmer que vous souhaitez écraser le fichier. Il y a plusieurs autres options dans les pages de manuel que vous pourrez trouver utiles. L'une d'elles est -v, qui affichera tous les fichiers au fur et à mesure qu'ils sont copiés ou déplacés. Vous pouvez utiliser des caractères joker pour copier ou déplacer plusieurs fichiers.

```
$ mkdir new-directory/
$ mv bar/* new-directory/
```

Cela déplacera tout le contenu de « bar » dans « new-directory ». Si vous préférez déplacer le répertoire entier, et pas seulement les fichiers et répertoires qu'il contient, utilisez la même commande mais sans l'étoile. Pour finir, supprimons tous les fichiers d'exemple avec la commande rm.

```
$ rm -rf bar/
$ rm -rf new-directory/
$ rm foo
```

Utiliser rm avec -rf efface récursivement les fichiers et répertoires, sans poser aucune question. On l'utilise en général pour effacer des répertoires et tout ce qu'ils contiennent. Faites très attention d'entrer le bon nom de fichier ou de répertoire en utilisant rm ; si vous utilisez TAB et ne vérifiez pas la complétion, vous pourriez effacer quelque chose par erreur ! Si quelqu'un vous aide et vous dit d'utiliser rm, assurez-vous que vous savez ce que vous effacez, surtout si la ligne commence par « sudo ». N'oubliez pas de sauvegarder les fichiers importants, même si vous pensez ne plus en avoir besoin !

Robert Clipsham est un geek qui s'assume, il passe son temps à : programmer/écrire des scripts, discuter sur IRC et à ne pas rendre ses articles à temps.

COMMAND AND CONQUER

Écrit par Robert Clipsham

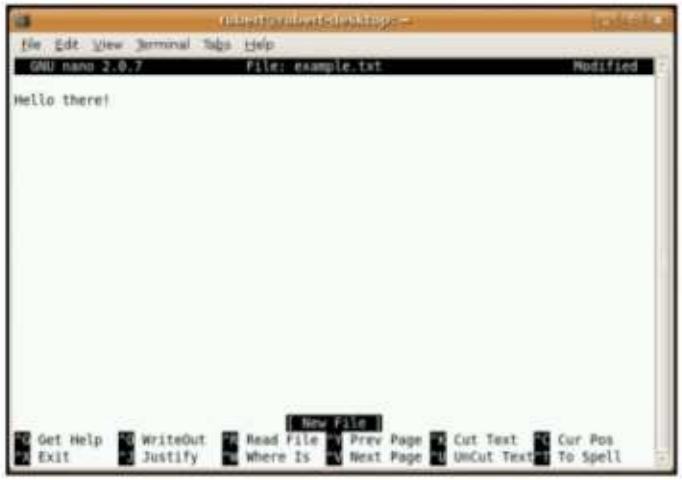
Si vous avez suivi les tutoriels jusqu'à maintenant, vous devez désormais connaître les bases pour survivre et gérer des fichiers en ligne de commande. Ce mois-ci, nous allons vous apprendre à éditer des fichiers en utilisant Nano et Vi/Vim.

Pour commencer, je vais vous montrer le plus simple des deux éditeurs à utiliser, Nano (à droite).

```
$ nano example.txt
```

En haut, vous devez voir trois choses : sur la gauche « GNU Nano » suivi d'un numéro de version, au centre le nom du fichier, et à droite l'état du fichier. En bas, il y a deux rangées de commandes pour l'éditeur, puis une ligne d'état juste au-dessus. Pour éditer le fichier, commencez à écrire comme vous le feriez avec n'importe quel autre éditeur de texte !

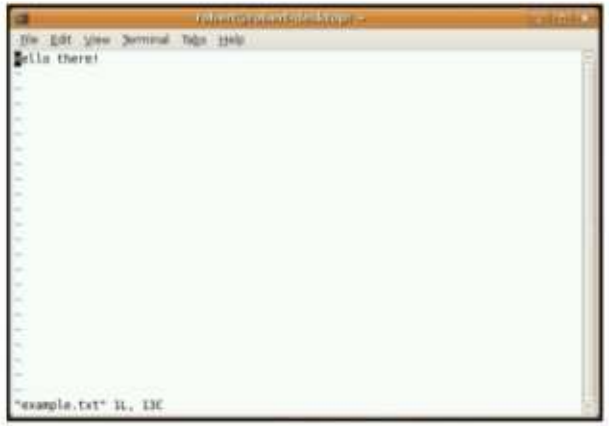
En plus de l'éditeur, les fonctions communes (et leurs combinaisons de touches) sont listées en bas. Par exemple, ^O (Ctrl+O) va « Écrire »



ou sauvegarder le fichier et ^W (Ctrl+W) vous permet de chercher un terme dans le fichier. Entrez du texte dans le fichier puis sauvegardez-le. Nano est le plus pratique à utiliser des deux éditeurs, mais il regroupe moins de fonctionnalités que Vi ou Vim.

Par défaut, Ubuntu contient uniquement le paquet Vim basique, sans tous ses « artifices ». Ce qui est bien pour ce tutoriel, mais pour de l'édition de fichier plus avancée, vous aurez besoin du paquet complet (nous vous montrerons comment l'obtenir grâce à une ligne de commande dans le prochain tutoriel).

Les éditeurs de texte permettent principalement deux choses : éditer des fichiers et automatiser des tâches. Les tâches peuvent être variées, depuis chercher/remplacer jusqu'à sauvegarder un fichier. Dans un éditeur graphique, pour sauvegarder, il faut utiliser les menus et cliquer sur des boutons. Dans Nano, vous utilisez certaines combinaisons de touches pour faire ces tâches. Vim comprend deux modes pour ceci : un mode



insertion et un mode commande. Vim (page suivante) démarre en mode commande par défaut, donc vous n'êtes pas à même d'éditer le document.

```
$ vim exemple.txt
```

Pour passer en mode d'insertion, appuyez sur « i » ou la touche

Commande	Fonction
:w	Sauvegarder le document
:q	Quitter Vim
:q!	Quitter Vim sans sauvegarder
h,j,k,l	Se déplacer dans le document, vous pouvez aussi utiliser les flèches
:e [nomdefichier]	Ouvre le fichier donné
:help [commande]	ouvre la page d'aide principale ou affiche une aide spécifique
:set [option] [valeur]	Donne une liste d'options qui ont été configurées ou crée une option

« insert ». Vous pouvez maintenant éditer le fichier. Entrez quelques lignes de texte supplémentaires puis retournez en mode commande en appuyant sur la touche Echap (Esc). Il existe des centaines, si ce n'est des milliers de commandes dans Vim, fournissant pas mal de fonctions. Quelques-une des commandes basiques sont présentées dans la fig.1 (à gauche).

Ainsi, si vous voulez par exemple sauvegarder vos changements, entrez « :w » (toutes les commandes de Vim commencent par « : »). Vous pouvez aussi mélanger les commandes, par exemple « :wq » sauvegarde et quitte Vim.

Voici quelques-unes des commandes basiques pour Vim. Il y a aussi des commandes pour copier/coller, chercher/remplacer, surligner, montrer les numéros de ligne et quelques fonctions plus avancées. Si vous voulez plus d'informations, utilisez « :help » ou lisez la documentation sur http://www.vim.org/docs.php.

Robert Clipsham est un geek qui s'assume, il passe son temps à : programmer/écrire des scripts, discuter sur IRC et à ne pas rendre ses articles à temps.

COMMAND & CONQUER

Écrit par Robert Clipsham

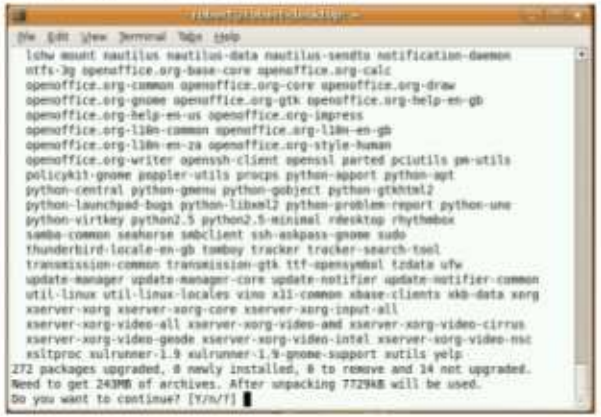
Ce mois-ci, nous allons vous montrer comment faire de la gestion de paquets à l'aide de la ligne de commande. Quand il s'agit de gérer des paquets en utilisant la ligne de commande, il y a deux outils principaux à votre disposition, apt-get et aptitude. Je ne vais pas entrer ici dans une comparaison de ceux-ci. En effet, une rapide recherche dans Google vous donnera plus d'informations que nécessaire à leur sujet. J'ai choisi d'utiliser aptitude pour ce tutoriel, mais vous êtes libre de faire votre propre choix basé sur les informations que vous aurez récoltées.

Tout d'abord, nous allons mettre à jour notre système. Ouvrez votre émulateur de terminal favori, puis saisissez ceci :

```
$ sudo aptitude update && sudo aptitude safe-upgrade
```

Au premier abord, ça peut sembler assez compliqué, par conséquent nous allons décomposer cette commande. Tout d'abord, vous remarquerez que nous avons utilisé « sudo ». Si vous suivez le tutoriel depuis le début, vous devez savoir que « sudo » lance la commande en tant qu'utilisateur « root », ce qui vous donne les pleins pouvoirs sur votre système. C'est indispensable sinon vous ne serez pas en mesure d'installer les paquets. Le morceau suivant dit à aptitude de mettre à jour (update) sa base de données des paquets, afin d'avoir la liste des derniers paquets disponibles. Le « && » est un opérateur de ligne de commande, il indique que la ligne de commande doit lancer une autre commande. La commande suivante indique à aptitude de lancer une mise à jour intelligente (safe-upgrade) de tous les paquets présents sur le système. C'est cette commande qui réalise vraiment la mise à jour des paquets. Il existe également une autre option disponible qui correspond à une mise à jour complète (full-upgrade). Celle-ci est moins conservatrice que la précédente et peut provoquer des effets non souhaités. Référez-vous à la page man pour de plus amples informations sur cette option. Il vous sera demandé d'entrer votre mot de passe « sudo », qui est le même que celui que vous utilisez pour vous connecter (vous devez avoir un compte privilégié pour faire cela, ça ne fonctionnera pas avec des comptes aux droits limités).

Vous allez voir beaucoup de texte défiler, celui-ci vous indique les sources où aptitude va chercher ses listes de paquets, ainsi que certaines informations sur leur statut. Les prochaines informations dans la console seront en rapport avec la deuxième commande. Ça ressemblera à quelque chose comme :

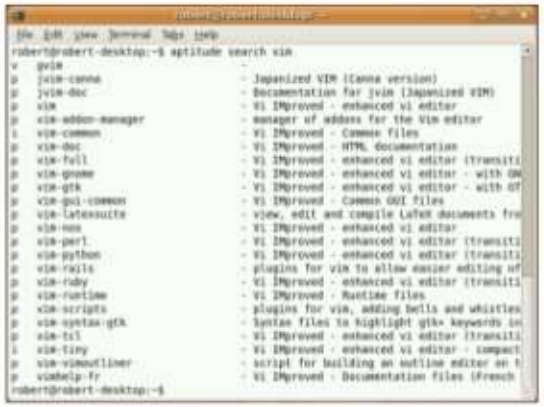


Lisez les informations qu'elle donne, et si vous êtes d'accord, saisissez « y » et appuyez sur Entrée. Dans le cas contraire, saisissez « n », et rien ne se passera. Comme vous pouvez le constater sur la capture d'écran, j'ai besoin de faire plusieurs mises à jour ; cependant, si vous avez gardé votre système à jour, elle vous retournera directement l'invite de commande puisqu'il n'y aura rien à faire.

La prochaine chose que vous vous devez de connaître avec aptitude, c'est comment ajouter et supprimer des paquets. Toutefois, c'est inutile si vous ne connaissez pas ceux qui sont disponibles. Commençons par rechercher un paquet. Dans le dernier numéro, j'ai fait mention du fait qu'Ubuntu n'avait pas par défaut tous les paquets vim, alors recherchons les.

```
$ aptitude search vim
```

Avez-vous remarqué que sudo n'est pas obligatoire lors d'une recherche de paquets ? Vous n'en avez pas besoin puisque vous ne modifiez aucun fichier du système. Vous devez voir apparaitre une liste à peu près comme celle-ci :



C'est une liste de paquets correspondant à votre critère de recherche, avec leur description et leur état. Voici un tableau (en haut à droite) vous montrant la signification de certains de ces codes

Statut	Description
p	Aucune trace de ce paquet sur le système.
c	Le paquet est supprimé, mais ses fichiers de configuration sont toujours sur le système.
i	Le paquet est installé.
v	Le paquet est virtuel.

d'états (vous pouvez également les trouver dans les pages man).

Maintenant, installons vim :

```
$ sudo aptitude install vim
```

Ça fonctionne de la même façon que safe-upgrade dans la mesure où il affiche les dépendances qu'il va installer. Si vous êtes sûr de vouloir l'installer, tapez « y » et appuyez sur Entrée. Vim va maintenant s'installer sur votre système, prêt à l'emploi. Pour le supprimer plus tard, vous pouvez faire :

```
$ sudo aptitude remove vim
```

Il sera alors supprimé. C'est aussi simple !

Robert Clipsham est un geek qui s'assume, il passe son temps à : programmer/écrire des scripts, discuter sur IRC et à ne pas rendre ses articles à temps.

COMMAND AND CONQUER

Écrit par Robert Clipsham

S'il vous est déjà arrivé d'égarer un fichier, la section « Command and Conquer » de ce mois-ci est faite pour vous. Il est possible de trouver un grand nombre d'outils de recherche graphique un peu partout, mais pourquoi utiliser une interface graphique (GUI) quand on peut utiliser la ligne de commande !

La première commande que nous allons examiner est « grep ». « Grep » est utilisé pour rechercher une chaîne de texte particulière dans un fichier. Par exemple, supposons que je possède un fichier texte appelé cookies.txt contenant une recette de cookies et que je désire savoir combien d'œufs sont nécessaires, je peux saisir :

```
$ grep oeuf cookies.txt
```

Cela affichera la liste de toutes les lignes contenant le mot « oeuf ». Prenons un exemple plus réaliste : supposons que je possède un fichier de configuration (je prendrai ici comme exemple « apache » bien que cela puisse être n'importe quoi d'autre) et que je veuille modifier le nom du fichier journal contenant les erreurs. Il n'est pas nécessaire de connaître tout ce qu'il y a dans le fichier, je veux juste savoir quelle ligne rechercher afin de pouvoir la modifier.

```
# grep errorlog
/etc/apache2/apache2.conf
```

Remarquez bien que je suis root pour lancer cette commande. Il se peut que vous soyez capable de la lancer en tant qu'utilisateur normal, cela dépend des permissions associées à votre fichier de configuration. Essayez de la lancer en tant qu'utilisateur normal dans un premier temps ! Vous pouvez remarquer que cette commande ne donne aucune réponse. La raison en est que la commande grep est sensible à la casse.

```
# grep -n ErrorLog
/etc/apache2/apache2.conf
```

L'option -n force grep à donner les numéros de ligne afin que vous puissiez trouver la ligne où se trouve la directive recherchée. Remarquez bien que j'ai mis des majuscules à ErrorLog dans cet exemple. On aurait également pu utiliser l'option -i pour forcer grep à ignorer la casse. Il est également possible de faire des recherches dans tous les fichiers d'un répertoire en utilisant l'option -r :

```
$ grep -ir oeuf recettes/
```

recherche toutes les recettes contenant le mot oeuf dans le répertoire recettes. Et si vous voulez rechercher dans les noms de fichiers ? C'est ici que la commande find montre tout son intérêt.

```
$ find recettes/ -type f -name
'*.jpg'
```

Cette commande permet de rechercher tous les fichiers (-type f) qui se terminent par .jpg. Remarquez la façon dont j'ai entouré *.jpg par des apostrophes. Cela empêche le shell de remplacer le caractère *. Essayez les commandes suivantes et observez la différence.

```
$ echo *
```

```
$ echo '*'
```

La première devrait afficher la liste de tous les fichiers du répertoire actuel alors que la seconde devrait afficher *. Il se peut que vous ne trouviez pas la commande find utilisée seule très utile, mais lorsqu'elle est utilisée conjointement avec xargs elle se révèle être un outil très puissant.

```
$ find recettes -type f -name '*-
gateau.txt' | xargs -I % cp %
anciennes-recettes/
```

Cette commmande récupère la sortie de « find recettes -type f -name '*-gateau.txt' » puis l'envoie (|) dans xargs.

L'option -I % indique à xargs de remplacer % par chaque ligne qu'il reçoit.

```
$ find recettes -type f -name '*-gateau.txt'
```

```
recettes/chocolat-gateau.txt

recettes/fromage-gateau.txt

recettes/magique-gateau.txt
```

Si la sortie est celle donnée ci-dessus, alors voici les commandes que xargs lancera :

```
$ cp recettes/chocolat-gateau.txt anciennes-recettes/
```

```
$ cp recettes/fromage-gateau.txt anciennes-recettes/
```

```
$ cp recettes/magique-gateau.txt anciennes-recettes/
```

Si vous avez beaucoup de fichiers qui possèdent le même motif, cela peut être une méthode très utile pour automatiser des tâches telles qu'une sauvegarde sélective. Si la sortie de find doit être ajoutée à la fin de la commande alors vous pouvez supprimer -I % et elle sera automatiquement ajoutée. La dernière commande que nous allons présenter est « locate ». Il se peut que locate ne soit pas installée donc installez-la pour que cela fonctionne. Locate permet de trouver très rapidement des fichiers portant un nom donné.

Si apache est installé sur votre machine, essayez ceci :

```
$ locate apache
```

Une très longue liste de fichiers apparaît donc il est sûrement judicieux de combiner la sortie avec grep afin de trouver exactement ce que vous recherchez.

```
$ locate apache | grep etc
```

Cela va restreindre la liste aux résultats contenant le mot « etc ».

Robert Clipsham est un mordu avoué de l'informatique dont les passe-temps sont : programmation/script, discussion sur IRC et retard dans l'écriture de ses articles.

COMMAND AND CONQUER

Ecrit par Philip Royer

Ça va bientôt faire une heure que vous êtes assis dans le cabinet médical. Chaque seconde semble durer une éternité. L'inquiétude commence à vous saisir les tripes. Les résultats des tests sont en cours d'examen par deux infirmières et leurs mines sombres n'annoncent rien de bon. Vous vous demandez ce qu'ils ont trouvé et vous espérez que votre vie ne soit pas en danger. Vous avez trop à perdre : votre famille, votre travail, votre belle voiture. Vous sursautez au moment où un médecin entre dans la salle d'attente, un bloc-notes dans les mains. Il lève la tête et vous regarde droit dans les yeux. « Je suis désolé Monsieur, mais il semble que cette maladie soit en phase terminale ». Votre cœur se serre, sachant que cette maladie n'aboutira qu'à la mort. Tout ce travail que vous avez fait pour... rien.

Je me souviens que ma première expérience avec le terminal de l'ordinateur était similaire à cela. J'étais TRÈS timide quand il fallait manuellement entrer du code dans l'ordinateur. Avec tout ce code qui défilait, je craignais que quelque chose ne tourne mal. Que faire si j'avais mal orthographié quelque chose, et que ça détruise mon ordinateur ? Au fil du temps cependant, cette inquiétude fut remplacée par une confiance que j'aurais pu avoir au départ.

C'est ce que je pensais à chaque fois que j'entendais le mot « terminal ». Ce n'est jamais un bon mot. Ainsi, j'eus un mouvement de recul quand un tutoriel m'a demandé d'ouvrir le terminal. Mais est-ce que la crainte d'entrer une série de lettres et de mots techniquement puissants est une raison suffisante pour être effrayé ? Est-ce une peur rationnelle ? Pourquoi est-ce qu'un si grand nombre a peur à l'idée de passer à Linux à cause de « devoir utiliser la ligne de commande ? » Permettez-moi d'essayer de briser certaines de ces craintes et de corriger toutes fausses pensées en vous aidant vous, l'utilisateur, à comprendre un peu mieux le terminal.

Qu'est-ce que le terminal ?

Le terminal est une application qui vous permet de parler à l'ordinateur à l'aide de commandes à base de texte. Cela signifie que vous éliminez le besoin d'utiliser une interface graphique, ou un tas de boutons conviviaux, au profit de commandes à lancer. On l'appelle la ligne de commande, ce qui veut dire qu'à la place de cliquer sur des boutons et des icônes, vous entrez des commandes avec du texte. Par exemple, pour mettre à jour votre système, vous devez taper :

```
sudo apt-get update
```

Il existe de nombreuses autres commandes que vous pouvez utiliser pour lancer des applications. En revenant sur les premiers jours de Linux, nous pouvons remarquer que la plupart des choses étaient faites en utilisant la ligne de commande. C'est une des nombreuses raisons pour laquelle beaucoup de non-geeks ne veulent pas, ou plutôt pas encore, passer à Linux.

La vérité, c'est que Linux s'est tellement éloigné de cet état primitif où tout devait être fait à l'aide de la ligne de commande, qu'il est actuellement au stade où il peut être utilisé par n'importe qui, expert comme non-expert. Mais alors si Linux, ou plus spécifiquement, Ubuntu, a atteint un niveau où la ligne de commande n'est plus nécessaire, pourquoi continuer à l'utiliser?

Pourquoi utiliser le terminal ?

Au moment où vous explorerez plus en profondeur Linux, le terminal se révèlera être votre meilleur ami. Il vous dira ce qui ne va pas. Peut-être pas de la même façon dont interagissent les humains, mais d'une manière très proche. Permettez-moi de développer un peu. Lorsque vous cliquez sur une icône du Bureau, ou sur un bouton quelconque, il envoie des commandes variées à l'ordinateur. Toutes ces commandes vous ne les voyez pas parce qu'elles se passent en arrière-plan. Si, par exemple, je devais cliquer sur le bouton de mise à jour de mon menu, afin de mettre à jour le système, la seule chose que je verrais est une barre d'état indiquant combien de temps il me reste avant que la mise à jour ne soit accomplie. Maintenant si je devais taper sudo apt-get update (l'équivalent en ligne de commande), j'obtiendrais une très longue liste complète d'adresses Web alors que mon ordinateur serait lui, en même temps, en pleine recherche de mises à jour. Vous pensez probablement, « pourquoi voudrais-je voir tout ça ? » La réponse est simple : il me dit ce qui se passe dans les coulisses. Eh bien, pourquoi aurais-je besoin de savoir ça ? Parce que s'il y avait un problème avec une installation, l'empêchant d'être menée à terme, alors il y aurait des messages d'erreurs lisibles dans le terminal mais que je ne pourrais pas voir sur le Bureau. Si j'avais juste obtenu « Je suis désolé, votre installation de ... n'a pas pu être terminée », et bien, je ne saurais pas ce qui s'est mal passé. Mais, en l'installant à l'aide du terminal, des messages d'erreurs apparaitront, m'informant du problème tout en me permettant de le résoudre, ou d'obtenir de l'aide. Mais, l'utilisation du terminal n'est pas pour tout le monde.

Devrais-je utiliser le terminal ?

Bien que le terminal soit très utile dans de nombreuses situations, je ne le recommande pas à tout le monde. Pour les utilisateurs de base, il pourrait être trop difficile à prendre en main, et selon les commandes concernées, elles pourraient accidentellement détériorer la machine. Mais, si vous avez un problème avec votre ordinateur, un programme qui ne s'exécute pas, ou une installation qui se bloque, vous pouvez poster la sortie du terminal sur le Web afin d'obtenir de l'aide d'autres personnes.

Dois-je utiliser le terminal ?

L'utilisation du terminal n'est pas nécessaire sous Ubuntu, c'est un plus pour vous aider quand vous avez des problèmes avec votre ordinateur. Même en tant qu'utilisateur avancé, je me sers rarement du terminal. Tout simplement parce qu'avoir roue de secours dans votre voiture, ça ne signifie pas qu'il faille l'utiliser tout le temps. Vous ne l'utilisez que si vous avez un pneu à plat. C'est aussi simple que ça.

En espérant que vous ne soyez plus effrayé par le terminal et que vous le voyiez maintenant comme un outil utile dont vous pouvez vous servir facilement. Seulement si vous osez, bien entendu. Il ne doit pas cependant prédominer dans l'utilisation que vous faites de votre ordinateur. On ne devrait jamais avoir peur de ce qu'on ne connait pas car sinon, nous n'apprendrions jamais rien de nouveau.

Malheureusement, Robert est intimidé par la vie réelle ces derniers temps et n'est pas en mesure de continuer à écrire C&C. Par conséquent nous sommes à la recherche d'un remplaçant pour quelques mois. Contactez Robert à cette adresse : mrmonday@fullcirclemagazine.org

COMMAND AND CONQUER

Par Lucas Westermann

Dans cet épisode de Command & Conquer, nous traiterons des utilisations de base de grep, sed, awk, cat, and cut pour les sorties formatées. Ceci peut se révéler utile quand vous rassemblez des données telles que celles de Conky, ou des scripts qui affichent des informations dans le terminal.

La première commande que nous devrions voir est cut. Si, par exemple, nous voulons afficher le nom de la distribution dans un script, on le trouverait dans /etc/issue. Si, toutefois, nous lançons cat /etc/issue nous voyons qu'il y a une ligne de trop et des caractères d'échappement inclus dans cette ligne. Ainsi si on lance /etc/issue|head -n 1, on retire la ligne superflue en redirigeant la sortie de cat vers head, qui n'affiche que la première ligne de la sortie. Pour l'instant tout va bien, mais quid des caractères d'échappement? C'est là que cut devient pratique. Pour utiliser cut, nous devons fournir un délimiteur, et lui dire ensuite quoi faire avec. La commande que nous utiliserons est :

```
cat /etc/issue|head -n 1|cut --delimiter=' ' -f 1,2
```

Cette commande indique à cut que le délimiteur à utiliser est un espace blanc, et d'afficher les deux premiers champs (concrètement, cut découpe la sortie en segments selon le délimiteur fourni, ainsi les champs 1 et 2 sont les premiers champs avant et après le délimiteur de la sortie, dans notre cas, Ubuntu 8.10). Cut peut aussi être utilisé pour afficher seulement un certain nombre de caractères en utilisant l'option -c.

On peut faire de même avec sed :

```
cat /etc/issue|sed '{s/\\n// ; s/\\l// ; /^$/d}'
```

Ça peut paraître confus, mais les deux premières expressions (chaque expression est séparée par un point-virgule) indiquent à sed de remplacer « \n » par «» (rien), et de même pour « \l », supprimant ces caractères de la sortie. « /^$d » est une commande qui demande a sed d'effacer toutes les lignes blanches (« ^$ » est l'expression usuelle pour une ligne qui commence par un blanc et qui termine par un blanc et rien entre — une ligne blanche). Ainsi « 's/\\n\b' » indique simplement à sed de substituer (« s/ ») « \n » (« \\n ») par «» (« »). La raison pour laquelle la commande est entre accolades, est parce que nous demandons trois expressions sur la sortie, et nous voulons qu'elle n'en retourne qu'une, donc nous plaçons l'expression entre accolades (« {} »), et séparée par des points-virgules.

Enfin, le même résultat peut être obtenu avec awk :

```
cat /etc/issue|awk '/\\n/ {print $1,$2}'
```

Cette commande utilise encore des expressions régulières, mais est légèrement plus facile à comprendre que sed. Fondamentalement, awk '/\n/ {print $1, $2}' trouve toute ligne contenant « \n », et affiche ensuite les deux premiers champs (le séparateur par défaut est un espace, mais vous pouvez spécifier le vôtre en utilisant l'option -f). Ceci nous évite de formater la ligne superflue et le \l de la sortie. Vous pouvez aussi renoncer à re-

diriger la sortie de cat /etc/issue (ou de toute autre commande), comme elles peuvent être appliquées à un fichier spécifié en fin de commande. J'ai utilisé cat afin de laisser les commandes moins désordonnées.

Ceci est seulement destiné à être une approche introductive aux possibilités de awk, sed et cut. Leur mise en application flexible rend difficile l'écriture d'un tutoriel court et approfondi sur leur usage. Les explications précédentes tendent à illustrer comment les commandes fonctionnent et n'explorent pas tout leur potentiel. Une réelle application de ces commandes serait dans la première moitié d'un script (l'exemple ci-après montre les informations d'un thème, mais cette partie n'est pas nécessairement pertinente avec cet article; on a renoncé à garder le script complet). L'exemple contient également un défi pour quiconque souhaite le relever : découvrez comment utiliser l'une des trois commandes pour supprimer l'indentation dans la partie mémoire du script, et si vous désirez vous entrainer davantage, essayez de remplacer toutes les occurrences de cut, sed, ou awk avec une autre commande qui fait la même chose (i.e. remplacer cut avec awk). Il n'y a aucun prix, mais c'est un bon entrainement pour comprendre le fonctionnement intrinsèque des commandes.

http://fullcirclemagazine.org/issue-21-shell-script/

Davantage de lecture:

Sed - http://www.gry-moire.com/Unix/Sed.html

awk - http://www.linuxjournal.com/article/8913 or http://www.linuxfocus.org/English/September1999/article103.html

cut - http://learnlinux.tsf.org.za/courses/build/shell-scripting/ch03s04.html

La page man (manuel) de chaque commande peut être consultée avec :

```
man [command]
```

Cela peut être pratique si vous n'êtes pas sûr(e) de savoir comment utiliser une commande.

Lucas a appris tout ce qu'il sait en plantant régulièrement son système et en n'ayant d'autre solution que de découvrir un moyen de le réparer. Quand il a le temps, il publie aussi un blog à :

http://lswest-ubuntu.blogspot.com.

COMMAND AND CONQUER

Par Lucas Westermann

Avez-vous déjà voulu réduire la taille d'une vidéo et l'ajouter à une présentation ? Ou peut-être la convertir dans un autre format pour pouvoir la regarder ? Aujourd'hui, je vais vous montrer comment réaliser ces deux opérations en utilisant l'outil en ligne de commande 'ffmpeg'. Je vous présenterai aussi la commande 'mogrify' qui est un outil d'édition d'image contenu dans le paquet imagemagik. Il permet de faire beaucoup de choses, mais je ne traiterai que des bases – principalement changer la taille d'une image (par exemple pour des aperçus réduits ou des petites images).

Pour utiliser ces outils, vous devrez installer ffmpeg et imagemagick à l'aide du gestionnaire de paquets Synaptic, dans le menu Ajouter/Enlever des applications ou, dans l'esprit de cet article, grâce à la ligne de commande avec :

```
sudo apt-get install ffmpeg
imagemagick
```

Il n'y a pas de risque à lancer cette commande d'installation (si vous ne savez pas si vous avez déjà installé ces paquets) car apt-get n'écrasera pas le programme existant, mais vous informera simplement qu'il est déjà installé. La commande va aussi vous demander votre mot de passe (parce que vous utilisez sudo). Si c'est la première fois que vous le faites, vous serez peut-être surpris de voir que rien n'est affiché quand vous saisissez votre mot de passe. C'est normal, tapez simplement votre mot de passe, puis pressez la touche Entrée.

Pour cet article, je vais convertir une courte vidéo de « Freedom Downtime » que j'ai utilisée dans une présentation. Ffmpeg offre une quantité d'options (pour lesquelles vous trouverez plus de détails dans la « page » du manuel de 13 pages – en utilisant la commande « man ffmpeg »), mais l'option que j'utilise le plus souvent est l'option pour convertir des fichiers. Le format de cette commande est :

```
ffmpeg -i fichier-source.type
fichier-cible.type
```

Cette commande va juste convertir le fichier source vers le fichier que vous avez spécifié avec 'fichier-cible.type' – sans changer la taille (puisque, sauf instruction contraire, ffmpeg conserve la taille de la source). Dans notre cas, pour convertir « Freedom Downtime » (taille de départ 640×480) vers une vidéo plus petite (disons 320×240), la commande serait :

```
ffmpeg -i freedom\ down-
time.mpg -s 320x240 freedom\
downtime\ retaillé.mpg
```

Mogrify est un outil très utile, particulièrement s'il vous arrive de poster de nombreuses images sur des forums qui n'autorisent pas l'envoi d'images excédant une certaine taille, ou qui interdisent de lier une image de grande taille comme pour les prévisualisations dans les messages. J'utilise mogrify principalement pour faire des aperçus d'images, mais le programme peut faire beaucoup de choses, comme ajouter du texte ou des

effets (fusain, colorisation, etc) et bien plus encore (à nouveau, tout est expliqué dans la page du manuel, que vous pouvez lire avec la commande « man mogrify »). Mogrify supporte des arguments de changement de taille soit en pourcentage, soit en pixels. Donc, si vous avez une image de 1280×800 pixels que vous voulez réduire à 640×400 pixels, vous pouvez le faire avec :

```
mogrify -resize 50 source.jpg
cible.jpg
```

ou :

```
mogrify -resize 640x400
source.jpg cible.jpg
```

ou même juste :

```
mogrify -resize 50% source.jpg
cible.jpg
```

Cependant, si la taille en pixels et le ratio d'aspect voulus sont différents, le résultat pourrait être plus petit qu'espéré, parce que mogrify réduira l'image aux valeurs les plus proches qui restent dans les mêmes proportions. Mogrify a aussi une option aperçu, « thumbnail », qui fait presque la même chose que l'option « resize », mais supprime les commentaires superflus, etc. des en-têtes du fichier pour en réduire la taille. La commande pour utiliser cette option serait :

```
mogrify -thumbnail 50 source.jpg
cible.jpg
```

Vous pouvez aussi utiliser mogrify pour convertir des images avec l'option « -format ». Ainsi,

```
mogrify -format jpg *.png
```

convertira tous les fichiers .png du dossier courant dans le format .jpg (les noms seront conservés).

Comme vous pouvez le voir, contrairement à ce que l'on croit, la ligne de commande peut servir dans des projets graphiques, et le fait souvent plus vite ou plus efficacement qu'une interface graphique avec des menus compliqués ou des présentations différentes en fonction des versions. La commande restera (habituellement) la même, et les arguments ne sont que très rarement modifiés. Et donc, les outils en ligne de commande sont beaucoup plus universels – c'est pourquoi les utilisateurs de ubuntuforums.org (ou forum.ubuntu-fr.org, ndt) proposent généralement les commandes au lieu des méthodes graphiques pour leurs solutions, car ces commandes sont les mêmes sous Kubuntu, Xubuntu et Ubuntu, aussi bien que pour d'autres systèmes. Avec un peu de chance, vous aurez trouvé cet article utile, et la prochaine fois que vous devrez convertir une vidéo ou une image, vous vous souviendrez de mogrify et de ffmpeg. Après tout, c'est en forgeant qu'on devient forgeron.

Pour en savoir plus :

http://www.imagemagick.org/www/mogrify.html – Guide très utile sur imagemagick en général, sur Ubuntu-fr.
http://www.ffmpeg.org/documentation.html - Une documentation française de ffmpeg sur Ubuntu-fr.

Lucas a appris tout ce qu'il sait en cassant son système, et en n'ayant plus alors d'autre choix que de trouver comment le réparer. Quand il en trouve le temps, il publie aussi un blog à l'adresse http://lswest-ubuntu.blogspot.com.

COMMAND AND CONQUER

Par Lucas Westermann

J'ai remarqué qu'il y avait un certain nombre d'articles postés sur les forums Ubuntu au cours des deux dernières semaines sur la manière de faire des rapports d'anomalies. Par conséquent, j'ai décidé de présenter certaines choses fondamentales à faire pour essayer de trouver l'origine du problème, afin de pouvoir faire une recherche dans google pour y trouver une solution (à moins de pouvoir la trouver sans cela). Pour information générale, les journaux (« logs ») sont stockés dans /var/log/. Il existe des journaux systèmes (pour tout) et puis un ensemble de journaux pour les applications ou les processus.

La première chose qui doit être faite si une application se bloque au démarrage (par exemple Firefox se fige et se bloque juste après son lancement) est de lancer l'application à partir du terminal pour ainsi afficher les messages d'erreurs dans celui-ci. Si vous obtenez un message d'erreur, la meilleure solution est de copier/coller l'essentiel de l'erreur dans google pour faire une recherche ou bien, si vous comprenez le message d'erreur, d'utiliser cette information pour savoir ce que vous devez faire afin de résoudre le problème.

Un problème plus difficile à résoudre est, par exemple, quand vous insérez une clé USB et qu'elle n'est pas reconnue par Nautilus. La première commande qui doit être exécutée est la suivante :

```
dmesg|tail
```

Regardez si la sortie fait référence à l'insertion d'une clé USB, ou toute autre chose qui relève de votre problème spécifique. Si ça n'apparait pas dans la sortie, vous pouvez soit essayer d'augmenter la quantité de lignes de la sortie en ajoutant le argument -n à tail suivi du nombre de lignes à afficher. Par exemple, pour afficher 14 lignes dans la sortie, saisissez :

```
dmesg | tail -n 14
```

sinon vous pouvez enlever et réinsérer la clé dans un nouveau port USB ou vérifiez les sorties de :

```
sudo fdisk -l
lsusb
```

pour regarder s'il n'y a pas quelques indications concernant la reconnaissance d'une clé USB. Si la clé est reconnue par le système, vous pouvez, pour obtenir un message d'erreur plus spécifique, essayer de la monter manuellement et regarder pourquoi ça échoue.

Les commandes et idées ci-dessus peuvent s'appliquer à presque n'importe quel problème que vous pourriez rencontrer, du moment que vous savez un minimum où chercher. La prochaine suggestion, cependant, concerne la lenteur du démarrage et l'examen de ce qui se passe lors du lancement de l'ordinateur, au cas où quelque chose traînerait et causerait ainsi un grand retard.

Ceci peut être fait par un programme appelé « bootchart » qui est dans les dépôts d'Ubuntu. Vous pouvez l'installer en saisissant :

```
sudo apt-get install bootchart
```

Une fois installé, vous n'aurez plus qu'à redémarrer votre ordinateur et ainsi vous pourrez visualiser

les résultats graphiques dans eye of gnome (le visionneuse d'images par défaut), en naviguant dans le répertoire /var/log/bootchart/ et en ouvrant la bonne image (nommées en fonction de la date).

En outre, les problèmes matériels peuvent être vérifiés par le programme lshw, qui énumère les informations sur le matériel. La façon la plus pratique pour l'utiliser est de l'exécuter avec l'option -C suivi de la section (display, network, etc.).

Ainsi, par exemple, les problèmes liés à la connexion sans fil sont vérifiés avec :

```
sudo lshw -C Network
```

Cette commande donne des détails sur vos périphériques réseaux (ethernet et sans fil) et autant d'informations que possible, allant des capacités jusqu'aux pilotes et ainsi de suite. Le plus important est sans doute de vérifier qu'il n'est pas désactivé et que le pilote apparaît bien (il est dans la dernière ligne de la section « device » et sera désigné par « driver=[nomDuDisque] »).

Dernier petit conseil, si vous rencontrez des erreurs ou des problèmes que vous ne parvenez pas à résoudre ou à corriger, donnez autant d'informations utiles que possible à toutes demandes que vous faites. Trop d'informations est mieux que pas assez. Par exemple, si vous travaillez sur un problème de connexion sans fil, ou sur un périphérique sans fil non reconnu, envoyez la sortie de commandes telles que ifconfig, iwconfig, lshw -C Network ; si c'est un dongle USB sans fil, postez également la sortie de lsusb, si c'est un PCI, celle de lspci, etc. Tout cela facilite la tâche car celui qui décide de vous aider n'aura pas à vous demander plus de renseignements si tout est déjà présent dans le premier message et toutes réponses ultérieures peuvent être à même de vous aider, sans long va-et-vient qui habituellement peuvent durer un jour ou deux (selon le fuseau horaire et l'heure de la journée du premier message). Il suffit de garder à l'esprit que plus d'informations sont fournies, plus les personnes auront matière à travailler afin de résoudre un problème qu'elles ne peuvent ni voir ni dépanner physiquement.

Cet article a été créé dans le but d'être un guide utile afin d'offrir aux gens un point de départ afin de résoudre leurs problèmes par eux-mêmes, ou d'augmenter leurs chances d'obtenir une aide lorsqu'ils en font la demande. Il n'est pas exhaustif et les commandes présentées n'ont pas été expliquées en détail. Toutes les commandes peuvent être examinées dans les pages du manuel (en utilisant la commande « man » qui a été présentée dans un précédent article de C&C) et c'est en utilisant les commandes que vous les comprendrez.

Lecture complémentaire :
http://www.troubleshooters.com/tpromag/200007/200007.htm

Lucas a appris tout ce qu'il sait en endommageant régulièrement son système, et en n'ayant alors plus d'autre choix que de trouver un moyen de le réparer. Quand il en trouve le temps, il publie également un blog à l'adresse http://lswest-ubuntu.blogspot.com.

COMMAND & CONQUER

Écrit par Lucas Westermann

Cet article sera dédié à l'automatisation de choses que vous faites tous les jours et à vous aider à créer une solution de sauvegarde qui fonctionne pour vous. Pour commencer, un « script bash » est un script qui contient des commandes bash de tous les jours et des fonctionnalités qui ne sont pas souvent utilisées en ligne de commandes (comme des boucles if-then-else, des boucles while, etc.). Cron est un démon qui exécute toutes les commandes listées dans le fichier crontab d'un utilisateur (donc si vous souhaitez exécuter quelque chose qui nécessite les droits de root, il faut éditer le crontab de l'utilisateur root). Pour cet article je vais utiliser un script bash simple d'une seule ligne qui va regrouper tous les fichiers .java d'un dossier en un seul fichier texte. Toutefois vous pouvez l'adapter pour regrouper des fichiers de log. Je vais vous expliquer comment fonctionne le script bash et comment utiliser cron pour qu'il exécute le script toutes les 6 heures.

Pour créer un script bash, mon éditeur préféré étant nano, c'est lui que j'utiliserai dans les exemples. Si vous préférez un éditeur avec une interface graphique, remplacez « nano » par « gedit » (sans les guillemets). Pour commencer, tapez ce qui suit :

```
nano FileCondenser
```

Cela lance une nouvelle interface dans un terminal avec un fichier totalement vide. Tapez ensuite les lignes suivantes (je vais les expliquer dans un moment) :

```
#!/bin/bash find $HOME/workspace/Year\ 11 -name '*.java'  | while read line; do cat "${line}"; done
```

Sauvegarder le fichier avec ctrl+x et appuyez sur Entrée. Pour rendre ce script exécutable, lancez la commande suivante :

```
chmod +x FileCondenser
```

Vous pouvez maintenant exécuter votre script en lançant dans un terminal :

```
./FileCondenser
```

ou en créant un lien symbolique vers /usr/bin (et en l'exécutant comme tout autre programme que vous utilisez dans le terminal) en tapant :

```
sudo ln -s /<chemin vers le script>/FileCondenser /usr/bin/FileCondenser
```

où vous remplacez « <chemin vers le script> » par le chemin réel (le chemin absolu est requis pour un lien symbolique).

Le script utilise essentiellement find pour trouver tous les fichiers qui se terminent par .java dans le dossier eclipse, et utilise pipe (redirection de la sortie) vers la boucle while qui indique que tant qu'il y a une ligne à lire

> **Le script utilise essentiellement find pour trouver tous les fichiers qui se terminent par .java...**

dans l'un de ces fichiers on la restitue et une fois qu'il n'y en a plus, on arrête.

Comme je l'ai dit plus haut vous pouvez changer le chemin en /var/logs/ ce qui donne :

```
find /var/logs/ -name "*.log"|while read line; do cat "${line}"; done
```

Cela va regrouper tous les fichiers qui se terminent par .log dans le dossier /var/log (et ses sous-dossiers) et les afficher dans le terminal. Mais vous pouvez rediriger la sortie vers un fichier.

Pour ajouter ce script à votre fichier crontab, je vous suggère de créer le lien symbolique afin de rendre la commande plus courte mais ce n'est pas obligatoire.

Pour éditer le fichier crontab, exécutez la commande suivante :

```
crontab -e
```

Cela va ouvrir le fichier crontab de l'utilisateur. Si vous souhaitez l'ajouter dans le fichier crontab de l'utilisateur root (pour des scripts de sauvegarde et des actions similaires) exécutez la commande suivante :

```
sudo crontab -e root
```

Une fois le fichier crontab ouvert, il est important de connaître le format de l'entrée. C'est le suivant : <minute> <heure> <jour> <mois> <jour de la semaine> <commande>. Toutes les sections doivent être remplies, soit par des caractères génériques, soit par des valeurs. Voici deux exemples :

Toutes les 5 minutes :

```
*/5 * * * * FileCondenser > $HOME/condenseFile.txt
```

Tous les dimanches à 18 heures :

```
00 18 * * sun FileCondenser > $HOME/condenseFile.txt
```

L'entrée que je suggère d'utiliser pour une sauvegarde périodique des logs serait :

```
* */2 * * * FileCondenser > $HOME/condenseFile.txt
```

ce qui va exécuter le script toutes les deux heures et sauvegarder la sortie dans un fichier de votre répertoire HOME appelé condenseFile.txt. Une petite note pour terminer, un script simple de sauvegarde auquel je peux penser, en serait un qui sauvegarde un dossier particulier ou un ensemble de dossiers (comme votre dossier home ou votre répertoire racine) et qui les enregistre sur une partition de sauvegarde, sur un disque dur externe, etc. Un exemple de ce script serait :

```
tar cvvzf /media/Backup/Musique/backup.tar.gz $HOME/Musique
```

Cette commande suppose que votre disque dur/partition de sauvegarde est monté sur /media/Backup et qu'il contient un dossier nommé « Musique ». La commande tar crée une archive tar gzippée dans /media/Backup/Musique/ nommée backup.tar.gz qui contient le contenu de votre dossier Musique. Vous pourriez exécuter cette commande depuis un crontab sans script, mais je suppose que si vous souhaitez faire des sauvegardes d'un système complet (de votre partition racine, par exemple) vous devrez inclure une longue liste de dossiers à ignorer, en lisant probablement cette liste dans un autre fichier, ainsi un script vous permettrait de garder tout ceci propre. Mais pas seulement, cela vous permettrait aussi de facilement l'exécuter manuellement dans un terminal. Je vous recommande de tester tous vos scripts et commandes dans un petit dossier de test avant de les utiliser en vrai sinon vous pourriez avoir des soucis.

Lectures complémentaires :

Une version Python du script pour les personnes intéressées : http://lswest.pastebin.com/m5b536464

Un tutoriel pour les scripts bash : http://www.linux.org/docs/ldp/howto/Bash-Prog-Intro-HOWTO.html

« *Linux Shell Scripting with Bash* » *de* Ken O. Burtch (ISBN: 978-0-672-32642-4)

Tutoriel sur cron : http://www.clickmojo.com/code/cron-tutorial.html

Lucas a appris tout ce qu'il sait en endommageant régulièrement son système, et en n'ayant alors plus d'autre choix que de trouver un moyen de le réparer. Quand il en trouve le temps, il publie également un blog à l'adresse http://lswest-ubuntu.blogspot.com.

COMMAND & CONQUER

Écrit par Lucas Westermann

Erratum au FCM nº 24 : /var/logs/ mentionné le mois dernier devrait être en fait : /var/log/ – toutes mes excuses.

Ne vous êtes vous jamais dit : « Il doit y avoir un moyen plus simple de faire ça », alors que vous fouillez dans l'historique des anciennes commandes à la recherche d'une seule bien spécifique ? Non seulement il y a un moyen plus simple, mais en plus il y a quelques trucs utiles que vous pouvez faire avec votre terminal et le shell pour vous rendre la vie plus simple. Mais, pour commencer, vous remarquerez que j'ai utilisé les termes « terminal » et « shell » et comme j'ai connu des gens qui pensaient que c'était la même chose, je vais par conséquent prendre un moment pour m'expliquer.

Un terminal est le programme qui affiche l'invite du shell, mais vous pouvez facilement changer le shell qu'il affiche. Ainsi, toute configuration concernant l'invite doit être faite dans le shell, et non dans le terminal. Deux shells couramment utilisés sont le Bash (Bourne Again Shell) qui est celui par défaut dans la plupart des cas et le Zsh (nommé de manière originale « Z Shell »). Si vous souhaitez essayer un nouveau shell, tout ce que vous avez à faire c'est l'installer puis saisir, par exemple :

```
zsh
```

à partir du shell bash actuel. Si vous décidez qu'il vous convient tout particulièrement et que vous souhaitez en changer (comme moi), saisissez :

```
sudo chsh <nomdutilisateur>
```

où bien sûr « <nomdutilisateur> » désigne votre nom d'utilisateur actuel. On vous demande alors quelque chose qui ressemble à ceci :

```
Changing the login shell
for <user>
Enter the new value, or
press ENTER for the default
Login Shell [/bin/zsh]:
```

où vous avez juste à saisir le chemin vers l'exécutable du shell (habituellement dans /bin/). L'entrée entre crochets est votre shell actuel (comme vous pouvez le voir, j'utilise Z shell).

Revenons à notre sujet : l'historique de votre shell peut parfois être très long et prêter à confusion (je crois que le mien en est rendu à environ 1000 commandes depuis mon dernier nettoyage, et il n'y a pas si longtemps que cela). Je lance très souvent les mêmes commandes, encore et encore, parfois avec beaucoup d'arguments, parfois avec très peu. Pour les plus longues, j'ai pris l'habitude de remonter dans l'historique pour retrouver la commande que je souhaite. Par exemple :

```
history|grep cd
```

renvoie :

```
996 cd Dropbox/Scripts/C
```

qui m'affiche la dernière commande cd que j'ai exécutée, ou toutes les commandes cd que j'ai exécutées s'il y en a plus d'une. Vous remarquerez que ça affiche également un nombre sur la gauche (dans mon cas, c'est 996). Donc si vous souhaitez exécuter à nouveau cette commande, deux possibilités. Vous pouvez copier/coller la ligne (ce qui n'est franchement pas très efficace puisque cela vous oblige à retirer les mains de votre clavier), ou vous pouvez saisir :

> Il doit y avoir un moyen plus simple de faire ça ...

```
!996
```

qui va automatiquement exécuter la commande de l'historique numéro 996. Waouh, me voici de retour dans le dossier de mes scripts C, trop facile ! Seulement 4 caractères pour une commande entière. Cela fait définitivement appel à mon côté (très) paresseux. C'est également bien plus rapide que d'utiliser les flèches haut/bas pour naviguer dans l'historique ligne par ligne pour retrouver la bonne commande. J'espère que cela vous aura plu à vous aussi. Une dernière remarque, vous pouvez également exécuter :

```
!cd
```

pour exécuter à nouveau la commande cd précédente.

Une autre chose très utile que j'ai apprise, c'est que vous pouvez ajouter des fonctions personnalisées à votre shell Bash (mais aussi au Z shell). Pour cela, vous devez éditer votre fichier .bashrc (avec gedit, vim, nano ou avec votre éditeur préféré). Moi, j'utilise vim.

```
vim .bashrc
```

Vous ouvrez alors un fichier très long (et probablement peu clair). J'ajoute généralement mes entrées personnalisées à la fin du fichier afin de bien les séparer de ce qui devrait s'y trouver et en ajoutant un commentaire à chaque fois. Mais faites comme bon vous semble. Si vous souhaitez simplement exécuter une commande plus facilement, comme par exemple utiliser la commande :

```
update
```

au lieu de saisir ceci :

```
sudo apt-get upgrade
```

alors je vous recommande d'utiliser des alias pour vous économiser des caractères à taper (pour moi, le nombre de touches à utiliser est très important quand on fait des raccourcis). Donc, si vous souhaitez créer un alias pour ça, il suffit d'ajouter la ligne suivante dans votre .bashrc (j'ai ajouté un commentaire pour être plus clair mais vous pouvez le retirer si vous voulez).

```
#Alias pour mettre à jour le système

alias update = "sudo apt-get upgrade"
```

Voici un autre alias que j'utilise très souvent :

```
#Un alias pour détailler la commande ls

alias ls = "ls -la --color=always --classify"
```

Comme vous l'avez probablement remarqué, j'ai en réalité remplacé la commande ls pour avoir une sortie bien plus détaillée. Maintenant vous pouvez vous demander « mais comment faire pour utiliser la vraie commande ls sans arguments ? ». La réponse est :

```
\ls
```

L'antislash passe outre n'importe quel alias lié à ce nom, et exécute la commande telle quelle.

Reconcentrons-nous maintenant sur les fonctions. Ce sont tout simplement des scripts, qui peuvent être très utiles, ajoutés directement au fichier de configuration de votre shell. L'exemple que je vais utiliser se trouve sur la page suivante. Ne vous inquiétez pas, je vais l'expliquer.

J'utilise ce script à l'occasion pour convertir des fichiers audios .m4a en .mp3 car je ne me vois pas taper toutes ces commandes à la main (même occasionnellement). La fonction est définie à la première ligne puis après la première accolade, c'est réellement le script. Il vérifie si les arguments sont vides, et si c'est le cas, il affiche le message d'erreur de la fin du script (l'antépénultième ligne). S'il y a bien des arguments, il vérifie que le premier fichier existe puis, crée alors le fichier de sortie (le .mp3 dans ce cas). S'il n'existe pas, il affiche « le fichier <fichier> n'existe pas ! ». Une fois que c'est fait, il vérifie que le fichier de sortie existe (le .mp3), pour s'assurer que la première boucle a été exécutée avec succès. Dans le cas contraire il ne déplace pas le fichier. Il déplace alors le fichier .m4a dans votre dossier Musique, en prenant le nom de la sortie (pour que vous puissiez savoir quel .m4a va avec quel .mp3), et échange le mp3 avec le m4a, pour qu'il puisse toujours être joué. Il dit alors qu'il a été bougé, et déplace le fichier mp3 dans le dossier Musique. Il vérifie également que le dossier Musique/m4a existe (le test if avant le commentaire m4a). S'il n'existe pas, il est créé avant de poursuivre. Avec un peu de chance, les gens vont trouver ce script utile, comme moi qui ai quelques fichiers m4a qui me restent de ma collection iTunes, et que je convertis quand je tombe dessus. La seule chose que je dois préciser c'est que la fonction nécessite que les noms de fichier soient entre guillemets (les antislashs, les espaces... ne fonctionnent pas). Ainsi la fonction doit être exécutée ainsi :

```
m4a "2-10 You're the Inspiration.m4a" "You're the Inspiration.mp3"
```

Les fichiers résultants se trouveront dans les dossiers ~/Musique et ~/Musique/m4a. Toutefois, il manquera au fichier mp3 les étiquettes id3 (id3tags).

Je laisse ça comme un défi à tout lecteur qui souhaitera le relever. Il existe des outils en ligne de commande qui vous permettent d'accéder aux données

des étiquettes, et tel un affront je vous annonce que le paquet perl-mp4-info d'archlinux (disponible dans le dépôt Arch User sur le web) est un outil qui lit les étiquettes des fichiers m4a (du moins, il devrait) et qui devrait être disponible pour Ubuntu. Si vous pouvez étendre le script pour qu'il copie les étiquettes du fichier m4a dans le fichier mp3 et que vous souhaitez partager votre solution, envoyez-moi le code par courriel et je l'ajouterai au prochain article (en vous citant bien sûr). Je n'ai pas encore écrit ce code, mais j'ai une bonne idée de comment y arriver (je n'ai pas senti l'utilité de tester mes algorithmes vu que je crois avoir converti tous mes fichiers m4a maintenant !). Je comparerai tout code que je recevrai avec mon algorithme, et je vous dirai dans le prochain article si cela correspond à la façon dont je voulais procéder, ou si c'est une manière à laquelle je n'avais pas du tout pensé. Autant que je sache, ffmpeg ne transfèrera pas l'information.

J'espère que ces astuces vous auront aidé et que vous les aurez trouvées au moins partiellement utiles. Je sais qu'elles m'ont évité beaucoup de frappes au clavier et qu'elles m'ont permis de travailler plus efficacement avec la ligne de commandes (j'utilise même Awesome [Ed : un gestionnaire de fenêtres dynamique et en mosaïque : il est dans les dépôts] sur mon système principal, ainsi je ne me sers presque plus de la souris ces temps-ci). J'attends avec impatience de voir si quelqu'un osera relever le défi de compléter la fonction ci-dessus.

```
m4a(){
if [[ "$1" != "" && "$#" == 2 ]]; then
    #vérifie si le fichier existe
    if [ -e "$1" ]; then
        #convertit l'audio
        ffmpeg -i "$1" "$2"
    else
        echo "Le fichier "$1" n'existe pas !"
    fi
    if [ ! -d "$HOME/Musique/m4a" ]; then
        mkdir $HOME/Musique/m4a
    fi
    #déplace le fichier .m4a dans le dossier m4a si le fichier
existe dans le répertoire courant
    if [ -e "$2" ]; then
        mv "$1" $HOME/Musique/m4a/"`echo "$2"|sed 's/mp3/m4a/'`"
        echo "Le fichier m4a a été déplacé dans le dossier
~/Musique"
        mv "$2" $HOME/Musique/
        echo "Le mp3 a été déplacé dans le dossier ~/Musique"
    else
        echo "Le fichier "$2" n'existe pas !"
    fi
else
    echo "Arguments invalides (ou trop/pas assez), merci de
lancer ce script comme suit : \"m4a <input> <output>\""
fi
}
```

Lucas a appris tout ce qu'il sait en endommageant régulièrement son système, et en n'ayant alors plus d'autre choix que de trouver un moyen de le réparer. Quand il en trouve le temps, il publie également un blog à l'adresse http://lswest-ubuntu.blogspot.com.

COMMENT CONTRIBUER

FULL CIRCLE A BESOIN DE VOUS !

Un magazine n'en est pas un sans articles et Full Circle n'échappe pas à cette règle. Nous avons besoin de vos opinions, de vos bureaux et de vos histoires. Nous avons aussi besoin de critiques (jeux, applications et matériels), de tutoriels (sur K/X/Ubuntu), de tout ce que vous pourriez vouloir communiquer aux autres utilisateurs de *buntu. Envoyez vos articles à :

articles@fullcirclemagazine.org

Nous sommes constamment à la recherche de nouveaux articles pour le Full Circle. Pour de l'aide et des conseils, veuillez consulter l'Official Full Circle Style Guide :
http://url.fullcirclemagazine.org/75d471

Envoyez vos **remarques** ou vos **expériences** sous Linux à : letters@fullcirclemagazine.org

Les tests de **matériels/logiciels** doivent être envoyés à : reviews@fullcirclemagazine.org

Envoyez vos **questions** pour la rubrique Q&R à : questions@fullcirclemagazine.org

et les **captures d'écran** pour « Mon bureau » à : misc@fullcirclemagazine.org

Si vous avez des questions, visitez notre forum : fullcirclemagazine.org

Merci de noter :

Les éditions spéciales sont un assemblage des numéros d'origine et les indications peuvent ne pas fonctionner dans les versions actuelles d'Ubuntu.

Équipe Full Circle

Rédacteur en chef - Ronnie Tucker
ronnie@fullcirclemagazine.org

Webmaster - Lucas Westermann
admin@fullcirclemagazine.org

Éditions spéciales - Jonathan Hoskin

Correction et Relecture
Mike Kennedy, Gord Campbell, Robert Orsino, Josh Hertel, Bert Jerred, Jim Dyer et Emily Gonyer

Remerciements à Canonical, aux nombreuses équipes de traduction dans le monde entier et à **Thorsten Wilms** pour le logo du FCM.

Pour la traduction française :
http://www.fullcirclemag.fr

Pour nous envoyer vos articles en français pour l'édition française :
webmaster@fullcirclemag.fr

Pour les Actus hebdomadaires du Full Circle :

Vous pouvez vous tenir au courant des Actus hebdomadaires en utilisant le flux RSS : http://fullcirclemagazine.org/feed/podcast

Ou, si vous êtes souvent en déplacement, vous pouvez obtenir les Actus hebdomadaires sur Stitcher Radio (Android/iOS/web) : http: www.stitcher.com/s?fid=85347&refid=stpr

et sur TuneIn à : http://tunein.com/radio/Full-Circle-Weekly-News-p855064/

Obtenir le Full Circle en français :

http://www.fullcirclemag.fr/?pages/Numéros

Obtenir le Full Circle Magazine :

Format EPUB - Les éditions récentes du Full Circle comportent un lien vers le fichier epub sur la page de téléchargements. Si vous avez des problèmes, vous pouvez envoyer un courriel à : mobile@fullcirclemagazine.org

Issuu - Vous avez la possibilité de lire le Full Circle en ligne via Issuu : http://issuu.com/fullcirclemagazine. N'hésitez surtout pas à partager et à noter le FCM, pour aider à le faire connaître ainsi qu' Ubuntu Linux.

Magzster - Vous pouvez aussi lire le Full Circle online via Magzster : http://www.magzter.com/publishers/Full-Circle. N'hésitez surtout pas à partager et à noter le FCM, pour aider à le faire connaître ainsi qu'Ubuntu Linux.