Guide rapide pour débuter avec FreeBSD à l'attention des utilisateurs de Linux®

John Ferrell

Version: 43184

Copyright © 2008 The FreeBSD Documentation Project

FreeBSD is a registered trademark of the FreeBSD Foundation.

Linux is a registered trademark of Linus Torvalds.

Intel, Celeron, Centrino, Core, EtherExpress, i386, i486, Itanium, Pentium, and Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Red Hat, RPM, are trademarks or registered trademarks of Red Hat, Inc. in the United States and other countries.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this document, and the FreeBSD Project was aware of the trademark claim, the designations have been followed by the « ™ » or the « ® » symbol.

Résumé

Ce document a pour but de familiariser rapidement les utilisateurs de Linux® de niveau intermédiaire à avancé avec les fondamentaux de FreeBSD.

Version française de Frederic Culot <culot@freebsd.org >.

Table des matières

1. Introduction 2
2. Interpréteurs de commandes: Pas de Bash? 2
3. Paquetages et logiciels portés: ajouter des logiciels sous FreeBSD 3
4. Démarrage Système: où sont les niveaux d'exécution (*run-levels*)? 5
5. Configuration Réseau 6
6. Pare-feu 7
7. Mettre à jour FreeBSD 8
8. procfs: disparu mais pas oublié 8
9. Commandes Usuelles 10
10. Conclusion 10

1. Introduction

Ce document met en évidence les différences entre FreeBSD et Linux® de telle sorte que les utilisateurs de Linux® d'un niveau intermédiaire jusqu'à avancé puissent se familiariser rapidement avec les fondamentaux de FreeBSD. Il s'agit ici simplement d'une courte introduction technique qui n'a pas pour objectif d'expliciter les différences « philosophiques » entre les deux systèmes d'exploitation.

Ce document part du principe que vous avez déjà installé FreeBSD. Si vous n'avez pas installé FreeBSD ou que vous avez besoin d'aide pour mener à terme le processus d'installation, vous pouvez vous référer au chapitre Installer FreeBSD du Manuel de Référence FreeBSD.

2. Interpréteurs de commandes: Pas de Bash?

Ceux qui ont l'habitude de Linux® sont souvent surpris de voir que Bash n'est pas l'interpréteur de commandes par défaut de FreeBSD. En fait, Bash n'est même pas présent dans l'installation par défaut. À la place, FreeBSD utilise tcsh(1) comme interpréteur par défaut. Cependant, Bash ainsi que vos autres interpréteurs de commandes favoris sont disponibles dans les Paquetages et logiciels portés de FreeBSD.

Si vous installez d'autres interpréteurs de commandes vous pouvez utiliser chsh(1) pour définir un interpréteur par défaut pour un utilisateur. Il est cependant recommandé de ne pas modifier l'interpréteur de commandes par défaut de `root`. La raison en est que les interpréteurs de commandes qui ne sont pas inclus dans la distribution de base sont normalement installés dans `/usr/local/bin` ou `/usr/bin` . Dans le cas d'un problème les systèmes de fichiers contenant `/usr/local/bin` et `/usr/bin` peuvent ne pas être montés. Dans ce cas `root` n'aurait pas accès à son interpréteur de commandes par défaut, ce qui empêcherait `root` de pouvoir se connecter. Pour cette raison un second compte `root`, le compte `toor`, a été créé pour l'utilisation avec des interpréteurs de commandes qui ne sont pas ceux par défaut. Vous pouvez consulter les questions fréquemment posées sur la sécurité concernant le compte toor pour plus d'information.

3. Paquetages et logiciels portés: ajouter des logiciels sous FreeBSD

En plus de la traditionnelle méthode UNIX® d'installation de logiciels (télécharger les sources, extraire, éditer le code source, et compiler), FreeBSD offre deux autres méthodes pour installer des applications: les paquetages et les logiciels portés. Une liste complète de tous les logiciels portés et paquetages disponibles se trouve ici.

3.1. Paquetages

Les paquetages sont des applications pré-compilées, les équivalents FreeBSD des fichiers `.deb` pour les systèmes basés sur Debian/Ubuntu et des fichiers `.rpm` pour les systèmes basés sur Red Hat/Fedora. Par exemple, la commande suivante installe Apache 2.2:

```
# pkg_add /tmp/apache-2.2.6_2.tbz
```

Utiliser l'option `-r` indique à pkg_add(1) de télécharger automatiquement le paquetage et de l'installer, ainsi que toutes ses dépendances:

```
# pkg_add -r apache22
Fetching ftp://ftp.freebsd.org/pub/FreeBSD/ports/i386/packages-6.2-
release/Latest/apache22.tbz... Done.
Fetching ftp://ftp.freebsd.org/pub/FreeBSD/ports/i386/packages-6.2-
release/All/expat-2.0.0_1.tbz... Done.
Fetching ftp://ftp.freebsd.org/pub/FreeBSD/ports/i386/packages-6.2-
release/All/perl-5.8.8_1.tbz... Done.
[snip]

To run apache www server from startup, add apache22_enable="YES"
in your /etc/rc.conf. Extra options can be found in startup script.
```

Note

Si vous utilisez une version *RELEASE* de FreeBSD (6.2, 6.3, 7.0, etc., généralement installée depuis un CD-ROM) `pkg_add -r` téléchargera les paquetages compilés spécifiquement pour cette version. Ces paquetages peuvent *ne pas* être la version la plus récente de l'application. Vous pouvez utiliser la variable `PACKAGESITE` pour surcharger ce comportement par défaut. Par exemple, assignez `ftp://ftp.freebsd.org/pub/FreeBSD/ports/i386/packages-6-stable/Latest/` à `PACKAGESITE` pour télécharger les paquetages les plus récents construits pour les versions 6.X.

Pour plus d'information concernant les paquetages vous pouvez vous référer à la section 4.4 du Manuel FreeBSD:Utiliser le système des logiciels pré-compilés.

3.2. Les logiciels portés

Le catalogue des logiciels portés est la seconde méthode proposée par FreeBSD pour installer des applications. Le catalogue des logiciels portés a pour architecture un ensemble de `Makefiles` et de fichiers correctifs spécifiquement adaptés pour pouvoir installer depuis les sources des applications diverses sur FreeBSD. Lors de l'installation d'un logiciel porté le système téléchargera le code source, appliquera tous les correctifs nécessaires, compilera le code, et installera l'application (et fera de même pour toutes ses dépendances).

Le catalogue des logiciels portés, parfois appelée l'arbre des ports (*ports tree* en anglais), peut être trouvé dans `/usr/ports` . Nous partons ici du principe que le catalogue des logiciels portés a été installé pendant le processus d'installation de FreeBSD. Si le catalogue des logiciels portés n'a pas été installé, il peut être ajoutée à partir des disques d'installation en utilisant sysinstall(8), ou bien récupéré depuis les serveurs FreeBSD en utilisant csup(1) ou portsnap(8). Des instructions détaillées concernant l'installation du catalogue des logiciels portés peuvent être consultées dans la section 4.5.1 du Manuel.

Installer un logiciel porté est aussi simple (en général) que de se déplacer dans le répertoire du logiciel porté et de lancer le processus de compilation. L'exemple suivant installe Apache 2.2 depuis le catalogue des logiciels portés:

```
# cd /usr/ports/www/apache22
# make install clean
```

Un des avantages majeurs d'utiliser les logiciels portés pour installer des logiciels est de pouvoir adapter les options d'installation. Par exemple, lors de l'installation d' Apache 2.2 depuis une version portée, il vous est possible d'activer mod_ldap en fixant la variable `WITH_LDAP` de make(1):

```
# cd /usr/ports/www/apache22
# make WITH_LDAP="YES" install clean
```

Vous pouvez vous référer à la section 4.5 du Manuel FreeBSD, Utiliser le catalogue des logiciels portés, pour obtenir plus d'information concernant le catalogue des logiciels portés.

3.3. Logiciel porté ou paquetage, lequel devrais-je utiliser?

Les paquetages sont simplement des logiciels portés pré-compilés, donc il s'agit vraiment de choisir entre une installation depuis les sources (logiciels portés) ou une installation utilisant des binaires pré-compilés. Chaque méthode présente ses avantages:

- Installation plus rapide (compiler de grosses applications peut prendre du temps).
- Inutile de comprendre comment compiler un logiciel.
- Inutile d'installer des compilateurs sur votre système.

- Possibilité d'adapter les options d'installation (les paquetages sont normalement compilés avec les options standards alors qu'avec les logiciels portés vous pouvez adapter diverses options comme la compilation de modules additionnels ou le changement de l'emplacement par défaut).

- Vous pouvez appliquer vos propres fichiers correctifs si vous le souhaitez.

Si vous n'avez pas de besoins particuliers, les paquetages seront probablement tout à fait adaptés à votre situation. S'il vous arrivait d'avoir des besoins spécifiques, les logiciels portés seraient plus appropriés (et n'oubliez pas que si vous devez effectuer des adaptations mais que vous préférez les paquetages, vous pouvez toujours compiler un paquetage personnalisé en utilisant `make package` et en copiant le paquetage sur d'autres serveurs).

4. Démarrage Système: où sont les niveaux d'exécution (run-levels)?

Linux® utilise le système d'initialisation SysV alors que FreeBSD utilise le style traditionnel BSD avec init(8). Avec init(8) il n'existe pas de niveaux d'exécution et pas de `/etc/inittab`, mais à la place le démarrage est contrôlé par l'utilitaire rc(8). Le script `/etc/rc` lit `/etc/defaults/rc.conf` et `/etc/rc.conf` pour déterminer quels services doivent être démarrés. Les services déclarés sont alors démarrés en lançant les scripts d'initialisation du service considéré que l'on trouve dans `/etc/rc.d/` et `/usr/local/etc/rc.d/` . Ces scripts sont similaires aux scripts que l'on trouve dans `/etc/init.d/` sur les systèmes Linux®.

> *Pourquoi existe-t-il deux emplacements distincts pour les scripts d'initialisation de services ?* Les scripts que l'on trouve dans `/etc/rc.d/` sont pour les applications qui font partie du système de base (cron(8), sshd(8), syslog(3), et d'autres). Les scripts dans `/usr/local/etc/rc.d/` sont pour les applications installées par les utilisateurs telles que Apache, Squid, etc.
>
> Quelle est la différence entre le système de « base » et les applications installées par l'utilisateur? FreeBSD est développé comme un système d'exploitation complet. En d'autres termes, le noyau, les bibliothèques système, et les utilitaires présents dans l'espace utilisateur (tels que ls(1), cat(1), cp(1), etc.) sont développés et distribués conjointement. C'est à cela que l'on fait référence en parlant de système de « base ». Les applications installées par l'utilisateur sont des applications qui ne font pas partie du système de « base », telles que Apache, X11, Mozilla Firefox, etc. Ces applications sont généralement installées en utilisant le Catalogue des logiciels portés et les paquetages de FreeBSD. Afin de les conserver à l'écart du système de « base », les applications installées par l'utilisateur sont normalement placées dans `/usr/local/` . Ainsi les binaires installés par l'utilisateur se trouvent dans `/usr/local/bin/` , les fichiers de configuration dans `/usr/local/etc/` , et ainsi de suite.

Les services sont activés en spécifiant `NomDuService_enable="YES"` dans `/etc/rc.conf` (rc.conf(5)). Pour vous faire une idée, vous pouvez consulter les valeurs par défaut du système dans `/etc/defaults/rc.conf` , ces valeurs par défaut sont surchargées par celles trouvées dans `/etc/rc.conf` . De plus, lors de l'installation de logiciels additionnels, veillez à consulter leur documentation pour déterminer de quelle manière sont activés les services associés, le cas échéant.

Le bout de code suivant extrait de `/etc/rc.conf` active sshd(8) et Apache 2.2. Il spécifie également que Apache devrait être lancé avec SSL.

```
# enable SSHD
sshd_enable="YES"
# enable Apache with SSL
apache22_enable="YES"
apache22_flags="-DSSL"
```

Dès lors qu'un service a été activé dans `/etc/rc.conf` , ce service peut être démarré depuis la ligne de commande (sans avoir à redémarrer le système):

```
# /etc/rc.d/sshd start
```

Si un service n'a pas été activé il peut être démarré depuis la ligne de commande en utilisant `forcestart`:

```
# /etc/rc.d/sshd forcestart
```

5. Configuration Réseau

5.1. Interfaces Réseau

À la place d'un identifiant générique *ethX* que Linux® utilise pour identifier une interface réseau, FreeBSD utilise le nom du pilote suivi d'un nombre en tant qu'identifiant. La sortie suivante de ifconfig(8) montre deux interfaces réseau Intel® Pro 1000 (`em0` et `em1`):

```
% ifconfig
em0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        options=b<RXCSUM,TXCSUM,VLAN_MTU>
        inet 10.10.10.100 netmask 0xffffff00 broadcast 10.10.10.255
        ether 00:50:56:a7:70:b2
        media: Ethernet autoselect (1000baseTX <full-duplex>)
        status: active
em1: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        options=b<RXCSUM,TXCSUM,VLAN_MTU>
        inet 192.168.10.222 netmask 0xffffff00 broadcast ↺
192.168.10.255
        ether 00:50:56:a7:03:2b
        media: Ethernet autoselect (1000baseTX <full-duplex>)
        status: active
```

5.2. Configuration IP

Une adresse IP peut être assignée à une interface en utilisant ifconfig(8). Cependant, pour assurer la persistence même après un redémarrage, la configuration IP doit être enregistrée dans `/etc/rc.conf` . Dans l'exemple suivant sont spécifiés le nom de la machine, l'adresse IP, et la passerelle par défaut:

```
hostname="server1.example.com"
ifconfig_em0="inet 10.10.10.100  netmask 255.255.255.0"
defaultrouter="10.10.10.1"
```

Utilisez ceci pour configurer une interface pour DHCP:

```
hostname="server1.example.com"
ifconfig_em0="DHCP"
```

6. Pare-feu

Tout comme IPTABLES pour Linux®, FreeBSD offre également un pare-feu au niveau du noyau; en fait FreeBSD offre trois pare-feux distincts:

- IPFIREWALL
- IPFILTER
- PF

IPFIREWALL ou IPFW (la commande pour gérer un jeu de règles IPFW est ipfw(8)) est le pare-feu développé et maintenu par les développeurs FreeBSD. IPFW peut être couplé avec dummynet(4) pour fournir des possibilités de régulation du trafic et simuler différents types de connections réseau.

Voici un exemple de règle IPFW pour autoriser SSH en entrée:

```
ipfw add allow tcp from any to me 22 in via $ext_if
```

IPFILTER est le pare-feu applicatif développé par Darren Reed. Celui-ci n'est pas spécifique à FreeBSD et a été porté sur de nombreux systèmes d'exploitation tels que NetBSD, OpenBSD, SunOS, HP/UX, et Solaris.

Voici un exemple de règle IPFILTER pour autoriser SSH en entrée:

```
pass in on $ext_if proto tcp from any to any port = 22
```

Le dernier pare-feu, PF, est développé par le projet OpenBSD. PF a été créé pour remplacer IPFILTER, ce qui fait que la syntaxe de PF est très similaire à celle de IPFILTER. PF peut être couplé avec altq(4) pour fournir des possibilités de QoS.

Voici un exemple de règle PF pour autoriser SSH en entrée:

```
pass in on $ext_if inet proto tcp from any to ($ext_if) port 22
```

7. Mettre à jour FreeBSD

Il existe trois méthodes différentes pour mettre à jour un système FreeBSD: à partir des sources, les mises à jour binaires, et les disques d'installation.

Mettre à jour depuis les sources est la méthode la plus compliquée mais elle offre la plus grande flexibilité. Le processus implique de synchroniser une copie locale du code source de FreeBSD avec les serveurs CVS (*Concurrent Versioning System* - Système de gestion de Versions Concurrentes) de FreeBSD. Une fois que le code source local est à jour vous pouvez compiler les nouvelles versions du noyau et du système de base. Pour plus d'informations concernant les mises à jour depuis les sources vous pouvez consulter le chapitre sur la mise à jour du manuel FreeBSD.

Les mises à jour binaires ressemblent aux commandes `yum` ou `apt-get` utilisées pour mettre à jour un système Linux®. La commande freebsd-update(8) téléchargera les nouvelles mises à jour et les installera. Les mises à jour peuvent être programmées par l'intermédiaire de cron(8).

Note

Si vous utilisez cron(8) pour programmer vos mises à jour, assurez-vous d'utiliser la commande `freebsd-update cron` dans votre crontab(1) pour réduire le nombre de machines récupérant les mises à jour au même moment.

```
0 3 * * * root /usr/sbin/freebsd-update cron
```

La dernière méthode, qui permet de mettre à jour en utilisant les disques d'installation, est simple: démarrez depuis les disques d'installation et sélectionnez l'option de mise à jour.

8. procfs: disparu mais pas oublié

Avec Linux®, il vous est peut-être arrivé de consulter `/proc/sys/net/ipv4/ip_forward` pour déterminer si le routage IP était activé. Avec FreeBSD vous devriez utiliser sysctl(8) pour voir ce réglage ainsi que d'autres réglages système parce que procfs(5)

a été abandonné dans les versions actuelles de FreeBSD (notez que `sysctl` est disponible aussi sous Linux®).

Dans l'exemple du routage IP voici ce que vous utiliseriez pour déterminer si le routage IP est activé sur votre système FreeBSD:

```
% sysctl net.inet.ip.forwarding
net.inet.ip.forwarding: 0
```

L'option `-a` est utilisée pour lister tous les réglages du système:

```
% sysctl -a
kern.ostype: FreeBSD
kern.osrelease: 6.2-RELEASE-p9
kern.osrevision: 199506
kern.version: FreeBSD 6.2-RELEASE-p9 #0: Thu Nov 29 04:07:33 UTC 2007
    root@i386-builder.daemonology.net:/usr/obj/usr/src/sys/GENERIC

kern.maxvnodes: 17517
kern.maxproc: 1988
kern.maxfiles: 3976
kern.argmax: 262144
kern.securelevel: -1
kern.hostname: server1
kern.hostid: 0
kern.clockrate: { hz = 1000, tick = 1000, profhz = 666, stathz = ↺
133 }
kern.posix1version: 200112
...
```

Note

Certaines de ces valeurs `sysctl` sont uniquement accessibles en lecture.

procfs est parfois nécessaire comme pour faire fonctionner de vieux logiciels, pour examiner des appels système en utilisant truss(1), et pour la Compatibilité Binaire avec Linux® (celle-ci utilise cependant son propre procfs, linprocfs(5)). Si vous avez besoin de monter procfs vous pouvez ajouter la ligne suivante dans `/etc/fstab` :

```
proc                /proc           procfs  rw,noauto       0       0
```

Note

noauto fera en sorte que /proc ne soit pas monté automatiquement lors du démarrage.

Et ensuite montez procfs avec:

```
# mount /proc
```

9. Commandes Usuelles

9.1. Gestion des Paquetages

Commande Linux® (Red Hat/Debian)	Equivalent FreeBSD	Rôle
yum install paquetage / apt-get install paquetage	pkg_add -r paquetage	Installation de *paquetage* depuis un dépôt distant
rpm -ivh paquetage / dpkg -i paquetage	pkg_add -v paquetage	Installation de *paquetage*
rpm -qa / dpkg -l	pkg_info	Lister les paquetages installés

9.2. Gestion Système

Commande Linux®	Equivalent FreeBSD	Rôle
lspci	pciconf	Lister les périphériques PCI
lsmod	kldstat	Lister les modules noyau chargés
modprobe	kldload / kldunload	Charger/Décharger les modules noyau
strace	truss	Examiner les appels système

10. Conclusion

Nous esperons que ce document vous a fourni suffisamment d'informations pour que vous puissiez faire vos premiers pas avec FreeBSD. Vous pouvez consulter le Manuel FreeBSD

pour une couverture plus complète des sujets abordés ici ainsi que de tous les autres sujets non abordés dans ce document.