# Configurer un NAS avec FreeBSD sur SBC

**Table of Contents**



**FreeBSD** fonctionne sur les cartes SBC et a un support natif pour ZFS, cet article présente l'installation de **FreeBSD** avec le support ZFS.

**Crochet FreeBSD**: Crochet est un outil pour aider à créer des images FreeBSD amorçables. Il prend en charge de nombreuses cartes (Alix, **BananaPi**, BeagleOs, Chromebook Snow, OrangePi, PandaBoard, Pine64, RaspberryPi, RaspberryPi 2, Soekris, systèmes x86 génériques, Wandboard, PB polyvalent, VMWare, ZedBoard , Zybo).

Les ports FreeBSD sont téléchargés sur la plateforme http://ftp.freebsd.org/pub/FreeBSD/ports/.

Le script crochet.sh peut créer une image FreeBSD amorçable complète prête à être

copiée sur un périphérique approprié (par exemple, une carte SDHC, une carte Compact Flash, un lecteur de disque, etc.). Le script s'exécute sur FreeBSD.

L'utilisation du script pour créer une image consiste en quelques étapes :
- CRÉER un fichier de configuration (Commencer par copier config.sh.sample)
- La première ligne spécifie la configuration de carte que l'on utilise. Le nom doit correspondre exactement à un répertoire sous “board/”.
- Le fichier de configuration peut spécifier une grande variété de personnalisations pour l'image générée. Le fichier config.sh.sample comprend une documentation complète sous forme de commentaires.
- Exécuter crochet.sh en tant que root `$ sudo /bin/sh crochet.sh -c <fichier de configuration>`
- Le script vérifiera d'abord que l'on dispose des sources nécessaires et dira exactement comment obtenir celles qui manque. Suivre les instructions et réexécuter le script jusqu'à ce qu'on ait tout.
- Dès qu'il aura trouvé toutes les ressources requises, le script compilera alors tout et construira l'image disque. Cette partie du processus peut prendre plusieurs heures.

Crochet conserve le système construit, le noyau et d'autres fichiers entre les exécutions. Dans de nombreux cas, on peut ajuster la configuration et réexécuter crochet pour créer une nouvelle image en quelques minutes seulement. Cependant, si on met à jour les sources de FreeBSD ou effectue un changement de configuration important, il faudra probablement supprimer le contenu du répertoire de travail pour forcer tout à reconstruire à partir de zéro.

## Installation de l’image sur la carte

Il existe de nombreuse façon d’installer un système pour une Rapsberry. De la création d’un kernel spécifique à partir des sources jusqu’à l’achat d’une carte SD préinstallé.

On va passer à un niveau de complexité intermédiaire en téléchargeant une image toute faite et en la transférant sur une carte SD.

### Récupérer une image installable

Télécharger la version de l’image sur le site FTP sur le site raspbsd.org

Graver l'image sur la carte SD avec l'une des méthodes décrites ici.

Une fois le transfert terminé, on peut retirer la carte SD de la machine, la mettre dans le Raspberry et démarrer. Il n'y a pas d'interface graphique, juste un écran de console.

Pour les SBC rk3399 (Radxa RockPi, PINE64) le mauvais dtb est chargé. Si on veut que cela fonctionne, on peut faire ce qui suit (sur une machine freebsd):
- Le dernier **u-boot** dans les ports prend en charge la carte 4c, télécharger la source

dans https://github.com/S199pWa1k9r/ports/tree/master/sysutils
- Sur la machine FreeBSD (ou dans une VM) aller dans `/usr/ports/sysutils/u-boot-rock-pi-4`
- Modifier le Makefile et remplacer `BOARD_CONFIG` par `rock-pi-4c-rk3399_defconfig`
- Construire ce port

```
pkg install -y gmake bison gsed swig dtc u-boot-tools python37 dialog4ports py37-pyelftools
pkg install -y atf-rk3399 atf-rk3328 atf-sun50i_a64
pkg install -y aarch64-none-elf-gcc
make u-boot-pinebook-pro-2020.07
```

- Télécharger l'image officielle rockpro64 pour freebsd et la décompresser
- Écrire les images sur la carte emmc/SD
- Ensuite, écrire l'uboot

```
dd if=/usr/local/share/u-boot/u-boot-rock-pi-4/idbloader.img of=/path/to/sdcarddevice seek=64 bs=512 conv=sync
dd if=/usr/local/share/u-boot/u-boot-rock-pi-4/u-boot.itb of=/path/to/sdcarddevice seek=16384 bs=512 conv=sync
```

Une version précompilée pour RAdxa Rock Pi 4 est disponible à l'adresse suivante https://www.freshports.org/sysutils/u-boot-rock-pi-4/. Pour l'installer sur la carte SD juste faire:
- `dd if=/usr/local/share/u-boot/u-boot-rock-pi-4/idbloader.img of=/path/to/sdcarddevice seek=64 bs=512 conv=sync`
`dd if=/usr/local/share/u-boot/u-boot-rock-pi-4/u-boot.itb of=/path/to/sdcarddevice seek=16384 bs=512 conv=sync`

# Configuration de base

À ce stade, on peut supprimer le clavier USB et le moniteur HDMI, me connecter à distance avec ssh en tant qu'utilisateur wkt et utiliser la commande su pour accéder à un shell root.

## Première connexion

À la première connexion, se connecter en tant que root :

```
login: root
```

Le mot de passe est soit vide (taper directement sur entrée pour se connecter à FreeBSD) soit root

## Changement du mot de passe

Il faut immédiatement changer (ou plus exactement mettre) le mot de passe de root :

```
# passwd
Changing local password for root
New Password:
Retype New Password:
```

## Création d'un nouvel utilisateur

Afin d'éviter de modifier la configuration de sshd et autoriser les connexions via le réseau de root, créer un utilisateur qui aura les mêmes privilèges que root.

La commande de gestion des utilisateurs est pw, mais on utilisera la commande adduser

```
# adduser
Username: jacques
Full name: jacques foucry
Uid (Leave empty for default):
Login group [jacques]:
Login group is jacques. Invite jacques into other groups? []: wheel
Login class [default]:
Shell (sh csh tcsh nologin) [sh]: sh
Home directory [/home/jacques]:
Home directory permissions (Leave empty for default):
Use password-based authentication? [yes]:
Use an empty password? (yes/no) [no]:
Use a random password ? (yes/no) [no]:
Enter password:
Enter password again:
Lock out the account after creation? [no]:
Username   : jacques
Password   : *****
Full Name  : jacques foucry
Uid        : 1002
Class      :
Groups     : jacques wheel
Home       : /home/jacques
Home Mode  :
Shell      : /bin/sh
Locked     : no
OK ? (yes/no): y
adduser: INFO: Successfully added (jacques) to the user database.
Add another user? (yes/no): n
Goodbye
```

Editer `/etc/group` pour mettre l'utilisateur dans le groupe wheel ; cela permet à wkt d'exécuter la commande su et de devenir root :

```
wheel:*:0:root,freebsd,wkt
```

## Configurer la connexion réseau

Il faut connaître l'adresse IP de la machine pour se connecter dessus par le réseau :

```
# ifconfig
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> metric 0 mtu 16384
    options=600003<RXCSUM,TXCSUM,RXCSUM_IPV6,TXCSUM_IPV6>
    inet6 ::1 prefixlen 128
    inet6 fe80::1%lo0 prefixlen 64 scopeid 0x1
    inet 127.0.0.1 netmask 0xff000000
    nd6 options=21<PERFORMNUD,AUTO_LINKLOCAL>
ue0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
    options=80001<RXCSUM,LINKSTATE>
    ether b8:27:eb:47:8a:41
    inet 192.168.1.186 netmask 0xffff0000 broadcast 192.168.1.255
    media: Ethernet autoselect (100baseTX <full-duplex>)
    status: active
    nd6 options=29<PERFORMNUD,IFDISABLED,AUTO_LINKLOCAL>
```

Pour conserver toujours la même adresse, définir une adresse IP statique pour le Pi en ajoutant ces lignes à /etc/rc.conf :

```
ifconfig_ue0="inet 10.10.1.90 netmask 255.255.255.0"
defaultrouter="10.10.1.1"
```

Ceci fait, on peut configurer l'interface ue0 :

```
/etc/rc.d/netif restart ue0
```

Et configurer le serveur DNS dans le fichier /etc/resolv.conf avec les détails des serveurs et le domaine à utiliser pour les noms sans points :

```
search local.net
nameserver 10.10.1.1
nameserver 8.8.8.8
```

Vérifier que la connection avec l'utilisateur jacques sur la machine en passant par ssh. Depuis une autre machine, tenter la commande suivante :

```
$ ssh jacques@192.168.1.186
Password for jacques@raspberry-pi:
Last login: Sun May 10 12:38:34 2015 from turing.example.com
FreeBSD 10.1-STABLE (RPI-B) #0 r282689: Sun May 10 07:19:57 UTC 2015

Welcome to FreeBSD!

Release Notes, Errata: https://www.FreeBSD.org/releases/
Security Advisories:   https://www.FreeBSD.org/security/
FreeBSD Handbook:      https://www.FreeBSD.org/handbook/
FreeBSD FAQ:           https://www.FreeBSD.org/faq/
Questions List:
https://lists.FreeBSD.org/mailman/listinfo/freebsd-questions/
FreeBSD Forums:        https://forums.FreeBSD.org/

Documents installed with the system are in the /usr/local/share/doc/freebsd/
directory, or can be installed later with:  pkg install en-freebsd-doc
For other languages, replace "en" with a language code like de or fr.

Show the version of FreeBSD installed:  freebsd-version ; uname -a
Please include that output and any error messages when posting questions.
Introduction to manual pages:  man man
FreeBSD directory layout:      man hier

Edit /etc/motd to change this login announcement.
To see the MAC addresses of the NICs on your system, type

    ifconfig -a
        -- Dru <genesis@istar.ca>
$
```

Et vérifier la capacité à devenir root :

```
$ su - root
Password:
```

Le mot de passe que l'on doit donner ici est le mot de passe de root2.

# Agrandir le système de fichier

Lorsque la carte SD utilisée a une taille bien plus large que les 120Mo utilisés pour l'installation (c'est suffisant pour faire booter le système, mais trop petit pour le faire tourner au quotidien), il faut agrandir le système de fichier.

## Repérer la bonne partition

La carte est découpée en deux partitions. La première (/boot), au format ntfs est utilisée pour

démarrer. La seconde au format ufs est utilisée par le système lui-même, c'est elle qu'il faut agrandir.

```
# df -h
Filesystem        Size    Used   Avail Capacity  Mounted on
/dev/mmcsd0s2a    907M    395M    440M    47%    /
devfs             1.0K    1.0K      0B   100%    /dev
/dev/mmcsd0s1      17M    3.6M     13M    21%    /boot/msdos
/dev/md0           29M     24K     26M     0%    /tmp
/dev/md1           14M     60K     13M     0%    /var/log
/dev/md2          4.4M     12K    4.0M     0%    /var/tmp
```

`/dev/mmcsd0s2a 907M 395M 440M 47% /` c'est elle. Son nom peut varier.

## Agrandir la partition

Pour commencer il faut obtenir des informations sur la partition :

```
# gpart show
=>      63  62148545  mmcsd0  MBR  (30G)
        63     34776       1  !12  [active]  (17M)
     34839   1918224       2  freebsd  (937M)
   1953063  60195545          - free -  (29G)

=>      0  1918224  mmcsd0s2  BSD  (937M)
        0      105           - free -  (53K)
      105  1918080         1  freebsd-ufs  (937M)
  1918185       39           - free -  (20K)
```

On voit que la carte dispose de 29Go de libre. On peut donc agrandir la partition de manière a prendre toute la place disponible :

```
# gpart resize -i 2 mmcsd0
mmcsd0s2 resized

# gpart show
=>      63  62148545  mmcsd0  MBR  (30G)
        63     34776       1  !12  [active]  (17M)
     34839  62109621       2  freebsd  (30G)
  62144460      4148          - free -  (2.0M)

=>       0  62109621  mmcsd0s2  BSD  (30G)
         0       105           - free -  (53K)
       105   1918080         1  freebsd-ufs  (937M)
   1918185  60191436           - free -  (29G)
```

Les 29Go libres ont été "transférés" de mmcsd0 à mmcsd0s2. Redémarrer la raspberry pour que les changements soient complètement pris en compte :

```
# reboot
```

Après le redémarrage, il faut “transférer” cette place libre sur la partition mmcsd0s2 celle qui est au format freebsd-ufs :

```
# gpart show
=>      63  62148545  mmcsd0  MBR  (30G)
        63     34776       1  !12  [active]  (17M)
     34839  62109621       2  freebsd  (30G)
  62144460      4148          - free -  (2.0M)

=>       0  62109621  mmcsd0s2  BSD  (30G)
         0       105            - free -  (53K)
       105   1918080         1  freebsd-ufs  (937M)
   1918185  60191436            - free -  (29G)

# gpart resize -i 1 mmcsd0s2
mmcsd0s2a resized
# gpart show
=>      63  62148545  mmcsd0  MBR  (30G)
        63     34776       1  !12  [active]  (17M)
     34839  62109621       2  freebsd  (30G)
  62144460      4148          - free -  (2.0M)

=>       0  62109621  mmcsd0s2  BSD  (30G)
         0       105            - free -  (53K)
       105  62101376         1  freebsd-ufs  (30G)
  62101481      8140            - free -  (4.0M)
```

On peut maintenant agrandir le système de fichier qui se trouve sur cette partition :

```
# growfs /
Device is mounted read-write; resizing will result in temporary write
suspension for /.
It's strongly recommended to make a backup before growing the file system.
OK to grow filesystem on /dev/mmcsd0s2a, mounted on /, from 937MB to 30GB?
[Yes/No] Yes
super-block backups (for fsck_ffs -b #) at:
 1918400, 2397952, 2877504, 3357056, 3836608, 4316160, 4795712, 5275264,
5754816, 6234368, 6713920, 7193472, 7673024, 8152576, 8632128,
 9111680, 9591232, 10070784, 10550336, 11029888, 11509440, 11988992,
12468544, 12948096, 13427648, 13907200, 14386752, 14866304, 15345856,
 15825408, 16304960, 16784512, 17264064, 17743616, 18223168, 18702720,
19182272, 19661824, 20141376, 20620928, 21100480, 21580032,
 22059584, 22539136, 23018688, 23498240, 23977792, 24457344, 24936896,
25416448, 25896000, 26375552, 26855104, 27334656, 27814208,
 28293760, 28773312, 29252864, 29732416, 30211968, 30691520, 31171072,
31650624, 32130176, 32609728, 33089280, 33568832, 34048384,
 34527936, 35007488, 35487040, 35966592, 36446144, 36925696, 37405248,
```

```
37884800, 38364352, 38843904, 39323456, 39803008, 40282560,
 40762112, 41241664, 41721216, 42200768, 42680320, 43159872, 43639424,
44118976, 44598528, 45078080, 45557632, 46037184, 46516736,
 46996288, 47475840, 47955392, 48434944, 48914496, 49394048, 49873600,
50353152, 50832704, 51312256, 51791808, 52271360, 52750912,
 53230464, 53710016, 54189568, 54669120, 55148672, 55628224, 56107776,
56587328, 57066880, 57546432, 58025984, 58505536, 58985088,
 59464640, 59944192, 60423744, 60903296, 61382848, 61862400
```

Et une fois de plus, redémarrer la machine pour avoir beaucoup de place :

```
$ df -h
Filesystem        Size    Used   Avail Capacity  Mounted on
/dev/mmcsd0s2a     29G    395M     26G     1%    /
devfs             1.0K    1.0K      0B   100%    /dev
/dev/mmcsd0s1      17M    3.6M     13M    21%    /boot/msdos
/dev/md0           29M     24K     26M     0%    /tmp
/dev/md1           14M     56K     13M     0%    /var/log
/dev/md2          4.4M    8.0K    4.0M     0%    /var/tmp
```

# Mise à jour du système et installation de logiciel

Maintenant qu'on a de la place sur la partition, on peut installer des paquets supplémentaires.

### Méthode 1: Récupérer et installer un arbre de paquet

On va télécharger et installer une arborescence décrivant les paquets que l'on veut installer par la suite.

```
# portsnap fetch
Looking up portsnap.FreeBSD.org mirrors... 7 mirrors found.
Fetching snapshot tag from ec2-eu-west-1.portsnap.freebsd.org... done.
Fetching snapshot metadata... done.
Fetching snapshot generated at Wed May 27 00:04:21 UTC 2015:
c85bcce40986c3d6b0527149349855bb797070a934082f100% of   75 MB 1455 kBps
00m53s
Extracting snapshot... done.
Verifying snapshot integrity... done.
Fetching snapshot tag from ec2-eu-west-1.portsnap.freebsd.org... done.
Fetching snapshot metadata... done.
Updating from Wed May 27 00:04:21 UTC 2015 to Wed May 27 15:09:33 UTC 2015.
Fetching 4 metadata patches... done.
Applying metadata patches... done.
Fetching 0 metadata files... done.
Fetching 125 patches.
(125/125) 100.00%  done.
```

```
done.
Applying patches...
done.
Fetching 8 new ports or files... done.
# portsnap extract
.../...
/usr/ports/x11/xwud/
/usr/ports/x11/xxkb/
/usr/ports/x11/xzoom/
/usr/ports/x11/yad/
/usr/ports/x11/yakuake-kde4/
/usr/ports/x11/yalias/
/usr/ports/x11/yeahconsole/
/usr/ports/x11/yelp/
/usr/ports/x11/zenity/
Building new INDEX files... done.
```

On a installé dans /usr/ports une arborescence de paquets. Ce sont les descriptions des logiciels, avec les URL de téléchargement des sources, des dépendances, où l'installer et comment.

Les répertoires `/usr/ports/>package Name>` contiennent les sources des paquets :

- **Makefile** c'est le fichier qui contient les instructions pour compiler le logiciel. Quels sont les dépendances, les besoins, etc. C'est le cœur du système de compilation ;
- **distinfo** un fichier au contenu un peu obscur. Il s'agit simplement du nom du fichier de source à télécharger (sudo-1.8.13.tar.gz«/code> et de la somme de contrôle de ce même fichier). On trouve également la taille que doit faire ce fichier. Lorsque nous allons demander la compilation de sudo, le fichier qui contient les sources sera téléchargé et ces éléments seront vérifiés
  - **files** est un répertoire qui contient des fichiers de configuration du logiciel. Ils seront installés au moment opportun
  - **pkg-desc** comme son nom semble l'indiquer, ce fichier contient une description du logiciel
  - **pkg-plist** contient une liste des fichiers qui seront installés, où le seront-ils.

  Pour installer un paquet il suffit de se placer dans un de ces répertoires dans l'arbre puis faire une installation avec la commande `make install`

Par exemple pour istaller tmux[1] il faut aller dans `/usr/ports/sysutils/tmux` et lancer la commande:

```
# make install clean
```

On peut faire de même pour sudo qui se trouve dans `/usr/ports/security`.

**Mises à jour des paquets**

Les logiciels qu'on utilise évoluent très vite. Certains nécessitent des mises à jour régulières (souvent des mises à jour de sécurité). On a déjà utilisé la commande

portsnap pour récupérer l'arborescence des paquets. On vautiliser à nouveau cette commande pour mettre à jour l'arbre.

```
# portsnap fetch update
```

Pour éviter d'oublier de faire cette mise à jour, on peut demander au logiciel cron (qui fait partie du système de base) de faire cette commande régulièrement. Editer le fichier /etc/crontab et y ajouter la ligne suivante

```
22 4 * * * root portsnap -I cron update && pkg_version -vIL=
```

Ce qui signifie qu'à 4 heures 22 minutes, tous les jours, l'utilisateur root va lancer la commande `portsnap -I cron update && pkg_versin -vIL=`. Cette commande va réaliser la mise à jour de l'arborescence et envoyer un courrier électronique si l'un des logiciels installés nécessite une mise à jour.

### Méthode 2: Installation avec pkg

La commande pkg permet d'installer les packages en tant que root :

```
pkg install apache24 bash minidlna proftpd rsnapshot rsync samba48 sudo vtun
```

## Configuration des paquets

À ce stade, on n'a effectué aucune configuration pour ces packages.

### Sudo

On l'habitude d'exécuter sudo sous Linux, On peut donc installé le package sudo. Pour que les membres du groupe wheel puissent sudo supprime le commentaire au début de cette ligne dans le fichier `/usr/local/etc/sudoers`:

```
## Décommenter pour permettre aux membres du groupe wheel d'exécuter
n'importe quelle commande
%wheel ALL=(ALL) ALL
```

### NTP

Le Pi n'a pas d'horloge alimentée par batterie, pour démarrer le démon NTP et régler l'horloge système au démarrage, ajouter ces lignes à /etc/rc.conf :

```
```

```
ntpd_enable="YES"
ntpdate_enable="YES"                    # Run ntpdate to sync time on boot
ntpdate_hosts="pool.ntp.org"            # List of ntpdate(8) servers.
```

Pour définir le fuseau horaire du système, lier le fichier de zone correspondant ainsi :

```
ln -s /usr/share/zoneinfo/Australia/Brisbane /etc/localtime
```

## Configuration ZFS

on ne peut pas connecter un pool de miroirs ZFS existant au RPi3 car ce sont des disques SATA, mais ZFS est déjà configuré et prêt à fonctionner dans FreeBSD. Tout ce quil y a à faire est de brancher la clé USB sur le RPi3 existant et de faire (en tant que root):

```
zfs import TAFE2T
```

et voici ce qu'on vois :

```
[root@naspi /etc]# df -h
Filesystem                Size    Used   Avail Capacity  Mounted on
/dev/ufs/rootfs            14G    3.0G     10G    23%    /
devfs                     1.0K    1.0K      0B   100%    /dev
/dev/msdosfs/MSDOSBOOT     50M     13M     37M    26%    /boot/msdos
tmpfs                      50M    4.0K     50M     0%    /tmp
TAFE2T                    1.7T    171G    1.6T    10%    /TAFE2T
TAFE2T/Henry              1.6T     20G    1.6T     1%    /TAFE2T/Henry
```

Maintenant, pour s'assurer que cela est monté automatiquement au démarrage, faire l'import zfs pour que le système connaisse les noms des pools, et ajouter cette ligne à /etc/rc.conf :

```
zfs_enable="YES"
```

Redémarrer le système et voir ce qui se passe :

```
[root@naspi /etc]# reboot
Connection to naspi closed by remote host.
```

Après avoir attendu environ une minute que le RPi3 redémarre, on me reconnecte avec ssh et on ne vois ... aucun pool ZFS. Lorsqu'on fait `zfs mount -a` les pools sont montés. Il se passe donc quelque chose:

Il semble que ZFS démarre avant que les clés USB ne soient connectées. Une solution est donc d'ajouter ce script shell Bourne à /etc/rc.local :

```
#!/bin/sh
# Essaye de monter les systèmes de fichiers USB ZFS au démarrage
done="no"
for i in 1 2 3 4 5 6 7 8 9 10
do if [ -c "/dev/da0" -a $done = "no" ]
   then logger "Mounting the ZFS stuff"
    zfs mount -a
        done="yes"; break
   else
     logger "Waiting for /dev/da0"; sleep 30
   fi
done
```

# Plus de paramètres ZFS

Quand on lis la sortie de dmesg. on vois ces avertissements ZFS :

```
ZFS NOTICE: Prefetch is disabled by default if less than 4GB of RAM is
present;
            to enable, add "vfs.zfs.prefetch_disable=0" to
/boot/loader.conf.
ZFS WARNING: Recommended minimum kmem_size is 512MB; expect unstable
behavior.
             Consider tuning vm.kmem_size and vm.kmem_size_max
             in /boot/loader.conf.
```

LLe consensus est de laisser la prélecture désactivée lorsqu'il y a moins de 4G de RAM. Espérons que lorsque le RPi4 arrivera (avec la 4G), ce ne sera pas un problème. On également ajusté le **kmem_size** en ajoutant ces lignes à `/boot/loader.conf` :

```
# Tune the VMS settings for ZFS
vm.kmem_size="512M"
vm.kmem_size_max="512M"
```

### ProFTPD

Pour avoir un serveur utilisant le protocole FTP, On peut installer proftpd sur le RPi3. Dans `/usr/local/etc/proftpd.conf`, désactiver UseIPv6 car on ne l'utilise pas et pour activer le service au démarrage ajouter dans `/etc/rc.conf` :

```
proftpd_enable="YES"
```

On peut aussi le démarrer manuellement en faisant: `service proftpd start`

Pour se connecter en ftp en tant qu'utilisateur et voir le répertoire personnel, on va configurer un deuxième utilisateur (avec adduser) qui aura `/usr/sbin/nologin` comme shell (donc ils ne pourra pas utiliser ssh) et avec un répertoire personnel où les vidéos et les images seront stockées.

Pour entrer en ftp en tant que deuxième utilisateur il faut indiquer à proftpd que ce shell est valide, éditer `/etc/shells` et ajouter `/usr/sbin/nologin` en tant que shell valide.

## Serveur Web Apache

Une fois que le package apache24 est installé, il existe un service Web vanille Apache 2.4 prêt à l'emploi. Le ServerRoot est `/usr/local/www/apache24/data` et il existe un utilisateur www et un groupe www pour posséder les fichiers. Pour le démarrer, ajouter la ligne

```
apache24_enable="YES"
```

dans /etc/rc.conf puis exécuter la commande

```
service apache24 start
```

## Samba

Pour configurer le partage de fichiers SMB avec Samba, copier le fichier de configuration Samba existant à partir du serveur Dell existant. Sous FreeBSD, il s'agit du fichier `/usr/local/etc/smb4.conf`. Voici l'essentiel du fichier :

```
[global]
wins support = yes
workgroup = WKTHOME
server string = %h server (Samba, FreeBSD)
log file = /var/log/samba4/log.%m
log level = 3
max log size = 1000
hosts allow = 10.10.1., 10.10.2., 192.168.2., 127.0.0.1
security = user
; Allow HP printer to login with XP-style authentication
ntlm auth = yes

[Music]
comment = Music
path = /usr/500/Music
public = yes
writable = no

[Scan]
comment = Public Stuff
path = /usr/Winshare/Scan
```

```
public = yes
writeable = yes
printable = no
write list = @staff
```

Samba conserve sa propre base de données de nom d'utilisateur/mot de passe, on dois donc maintenant configurer un utilisateur Samba :

```
# smbpasswd -a -U wkt
New SMB password:
Retype new SMB password:
Forcing Primary Group to 'Domain Users' for wkt
Forcing Primary Group to 'Domain Users' for wkt
Added user wkt.
```

Comme d'habitude, il faut activer le service en ajoutant une ligne à /etc/rc.conf :

```
samba_server_enable="YES"
```

et exécuter une commande pour démarrer le service :

```
service samba_server start
```

## Service MiniDLNA

Pour diffuser sur Smart TV avec minidlna les musiques et de vidéos stockées sur le NAS, créer les répertoires de mes fichiers :

```
mkdir -p /usr/Winshare/Photos /usr/500/Photos /usr/500/Video /usr/500/Music
chmod 755 /usr/Winshare/Photos /usr/500/Photos /usr/500/Video /usr/500/Music
```

Ensuite, éditer `/usr/local/etc/minidlna.conf`, supprimer la ligne `media_dir` existante et ajouté les lignes adéquates:

```
< media_dir=/opt
---
> media_dir=P,/usr/Winshare/Photos
> media_dir=P,/usr/500/Photos
> media_dir=V,/usr/500/Video
> media_dir=A,/usr/500/Music
```

Comme toujours, éditer `/etc/rc.conf` pour activer le service au démarrage :

```
minidlna_enable="YES"
```

et démarrer le service :

```
service minidlna start
```

## NFS

Pour monter des répertoires à partir du boîtier NAS en utilisant le protocole NFS. On peut avoir des problèmes avec la connexion du client Linux au serveur FreeBSD en utilisant NFSv4. Voici donc un exemple de configuration FreeBSD. Tout d'abord, `/etc/rc.conf` :

```
mountd_enable="YES"
mountd_flags="-r -n"
nfs_server_enable="YES"
nfsuserd_enable="YES"
nfsuserd_flags="-verbose"
nfsv4_server_enable="YES"
rpcbind_enable="YES"
```

`/etc/exports` ressemble à ceci :

```
/usr/Winshare /usr/500 -maproot=root 10.10.1.2 10.10.1.14
```

On exporte donc deux points de montage vers deux clients et on permet à l'UID racine du client de se mapper sur l'UID racine du serveur.

Sur le client Linux (en root), on peut monter manuellement les répertoires exportés :

```
mount.nfs -v -o nfsvers=3 naspi:/usr/500 /usr/500
mount.nfs -v -o nfsvers=3 naspi:/usr/Winshare /usr/Winshare
```

Noter que j'oblige le client à utiliser NFSv3. Une fois le serveur Dell parti, sur ma machine Linux, on changerai mon /etc/fstab pour avoir ceci :

```
naspi:/usr/Winshare   /usr/Winshare   nfs     soft,noauto,nfsvers=3   0    0
naspi:/usr/500        /usr/500        nfs     soft,noauto,nfsvers=3   0    0
```

## NFS et ZFS

OK, donc apparemment, il existe un fichier d'exportation séparé pour les systèmes de fichiers ZFS. Et comme mes /usr/Winshare et /usr/500 sont des systèmes de fichiers ZFS, voici ce que on devais faire.

Tout d'abord, vider le fichier /etc/exports existant :

```
```

```
echo -n > /etc/exports
```

Maintenant, créer ce fichier /etc/zfs/exports :

```
/usr/Winshare -maproot=root 10.10.1.2 10.10.1.14
/usr/500 -maproot=root 10.10.1.2 10.10.1.14
```

# Utilisation d'un volume ZFS en tant que cible iSCSI

On peut facilement créer un volume ZFS en tant que cible iSCSI en configurant la propriété **shareiscsi** sur le volume. Exemple :

```
# zfs create -V 2g tank/volumes/v2
# zfs set shareiscsi=on tank/volumes/v2
# iscsitadm list target
Target: tank/volumes/v2
    iSCSI Name: iqn.1986-03.com.sun:02:984fe301-c412-ccc1-cc80-cf9a72aa062a
    Connections: 0
```

Une fois la cible iSCSI créée, configurer l'initiateur iSCSI.

La commande **iscsitadm** permet la création et la gestion de cibles Solaris iSCSI. Si on a configuré la propriété **shareiscsi** dans un volume ZFS, ne pas utiliser la commande **iscsitadm** pour créer le même périphérique cible. On peut également dupliquer les informations des cibles sur ce même périphérique.

On peut gérer un volume ZFS configuré en tant que cible iSCSI de la même façon qu'un autre jeu de données ZFS. Cependant, les opérations de renommage, d'exportation et d'importation fonctionnent de façon différente pour les cibles iSCSI.

- Lors du renommage d'un volume ZFS, le nom de la cible iSCSI ne change pas. Exemple :

```
    # zfs rename tank/volumes/v2 tank/volumes/v1
    # iscsitadm list target
    Target: tank/volumes/v1
        iSCSI Name: iqn.1986-03.com.sun:02:984fe301-c412-ccc1-cc80-
cf9a72aa062a
        Connections: 0
```

- L'exportation d'un pool contenant un volume ZFS entraîne la suppression de la cible. L'importation d'un pool contenant un volume ZFS entraîne le partage de la cible. Exemple :

```
    # zpool export tank
    # iscsitadm list target
```

```
    # zpool import tank
    # iscsitadm list target
    Target: tank/volumes/v1
        iSCSI Name: iqn.1986-03.com.sun:02:984fe301-c412-ccc1-cc80-
cf9a72aa062a
        Connections: 0
```

L'ensemble des informations de configuration de cible iSCSI est stocké dans le jeu de données. Tout comme un système de fichiers NFS partagé, une cible iSCSI importée dans un système différent est partagée adéquatement.

1)

Tmux, à l'instar de Screen, est un multiplexeur de terminaux, outil permettant d'exploiter plusieurs terminaux au sein d'un seul et même affichage.