

# Configurer un serveur d'application et un terminal X

Nicolas LS

27 mars 2002

# Table des matières

|          |   |           |
|----------|---|-----------|
| <b>1</b> | <b>Configurer le terminal X</b>                               | <b>4</b>  |
| 1.1      | Etherboot . . . . .   | 4         |
| 1.1.1    | Création de la disquette . . . . .                            | 4         |
| 1.1.2    | Liste des drivers . . . . .                                   | 4         |
| 1.2      | Récupération de l'adresse MAC . . . . .                       | 5         |
| 1.3      | Compilation du noyau pour le Terminal X . . . . .             | 5         |
| 1.3.1    | Sous FreeBSD . . . . .  | 5         |
| 1.3.2    | Sous linux . . . . .  | 6         |
| <b>2</b> | <b>Configurer le serveur</b>                                  | <b>7</b>  |
| 2.1      | Configurer le serveur DHCPD . . . . .                         | 7         |
| 2.2      | Créer la racine du système pour le terminal x . . . . .       | 8         |
| 2.2.1    | Copie des fichiers . . . . .                                  | 8         |
| 2.2.2    | Dossiers spécifiques a la racine du terminal X . . . . .      | 9         |
| 2.2.3    | Copie du noyau . . . . .                                      | 9         |
| 2.2.4    | Création du dossier /etc . . . . .                            | 9         |
| 2.3      | Configurer le serveur NFS . . . . .                           | 11        |
| 2.3.1    | Activer le serveur NFS . . . . .                              | 11        |
| 2.3.2    | Définir les fichiers a exporter . . . . .                     | 12        |
| 2.4      | Configurer le serveur XDMCP . . . . .                         | 12        |
| 2.4.1    | Approche . . . . .  | 12        |
| 2.4.2    | Configuration . . . . .                                       | 12        |
| 2.4.3    | Xservers . . . . .  | 13        |
| 2.4.4    | Autres fichiers . . . . .                                     | 13        |
| 2.4.5    | Relancer XDM . . . . .  | 13        |
| <b>3</b> | <b>Configurer un terminal servant des consoles "telnet"</b>   | <b>14</b> |
| 3.1      | Configurer le serveur . . . . .                               | 14        |
| 3.2      | Compiler un noyau . . . . .                                   | 14        |
| 3.3      | Créer la racine pour le client . . . . .                      | 15        |
| 3.3.1    | Copie des fichiers . . . . .                                  | 15        |
| 3.3.2    | Dossiers spécifiques a la racine du terminal telnet . . . . . | 16        |
| 3.3.3    | Copie du noyau . . . . .                                      | 16        |

|          |   |           |
|----------|---|-----------|
| 3.3.4    | Création du dossier /etc . . . . .        | 16        |
| <b>4</b> | <b>Quelques liens</b>                     | <b>20</b> |
| 4.1      | Documents génériques . . . . .            | 20        |
| 4.1.1    | Application . . . . .                     | 20        |
| 4.1.2    | A propos des boot diskless . . . . .      | 20        |
| 4.1.3    | A propos de NFS . . . . .                 | 20        |
| 4.1.4    | A propos de XDM . . . . .                 | 21        |
| 4.2      | Documents spécifiques à Linux . . . . .   | 21        |
| 4.3      | Documents spécifiques à FreeBSD . . . . . | 21        |

## Qu'est ce qu'un terminal X ?

Un terminal X est un dispositif se chargeant juste d'afficher les applications exécutées sur un serveur (généralement une machine assez puissante). L'intérêt d'un tel fonctionnement est qu'on peut ainsi mettre à jour tout en mettant à niveau uniquement le serveur X. Un vrai terminal X coûtant assez cher, ce document explique comment en faire un à partir d'un simple PC (486) par exemple.

En effet, cela peut être une application aux terminaux X : permettre de réutiliser de vieux PC qui allaient partir à la poubelle pour en faire un terminal X. Unix étant multiutilisateur, un même PC peut supporter plusieurs utilisateurs (via telnet, ssh, ou encore des terminaux X).

On peut par ailleurs mettre 15 terminaux X sans qu'un réseau 10 Mb soit saturé, pourvu que les terminaux X ne servent pas à jouer à Quake. X passe même si ce n'est pas extrêmement rapide par une ligne téléphonique, le débit d'un réseau local est donc largement suffisant.

Donc, si vous voulez reconvertir votre vieux 486, voilà la solution, vous aurez alors toute la puissance du serveur d'applications pour travailler, ce dernier ne servant qu'à l'affichage.

Ce document décrit aussi comment configurer un PC pour afficher des telnet sur un système distant, toujours en bootant diskless. Cela peut servir par exemple si vous avez un PC qui ne dispose que de très peu de RAM, et que le lancement de Xfree s'avère donc impossible. (Il faut 8 Mo ou 16 Mo de RAM pour lancer Xfree.)

## Introduction

Le but de ce document est d'expliquer comment créer facilement un serveur d'application et un terminal X à partir d'un OS (unix) déjà installé, dans mon cas, il s'agit de FreeBSD. Il faut noter que j'utilise 2 Pc pour cela, le serveur : shiva (192.168.0.1) et le terminal X : vishnu (192.168.0.2) .

Pour configurer tout cela, il faut uniquement 2 logiciels en plus de ce dont vous disposez déjà : isc-dhcpd et etherboot. Ils sont dans les ports, et on peut les trouver sur <http://www.freshmeat.net>.

Je n'ai pas particulièrement pris en compte l'aspect sécurité en écrivant ce document, il faudra donc sécurisé cela si le réseau est peu sur (via internet par exemple).

# Chapitre 1

## Configurer le terminal X

### 1.1 Etherboot

#### 1.1.1 Création de la disquette

Nous allons commencer par "configurer" le terminal X, en effet, il nous faut quelques infos avant de poursuivre la configuration.

Pour cela, il vous suffit de vous placer dans le répertoire des sources de etherboot. Sous FreeBSD, placez vous dans `/usr/ports/net/etherboot`, lancez un `make`, puis passez dans `work/etherboot-version`. Vous pouvez alors créer une disquette de boot via le réseau. Pour une carte Realtek 8029, la commande est `gmake bin32/rtl8029.fd0` ou `make bin32/rtl8029.fd0` si `make` est le GNU Make, et remplacez `rtl8029` par le driver qui convient. Cette commande doit être lancée sous `root`, et vous devez avoir introduit une disquette formatée dans le lecteur.

Sous linux, vous pourrez trouver ce logiciel sur <http://www.freshmeat.net>

#### 1.1.2 Liste des drivers

La liste des drivers disponible dans etherboot est :

|              |               |               |            |              |
|--------------|---------------|---------------|------------|--------------|
| 3c503        | 3c905b-fx     | cs89x0        | fa311      | otulip       |
| 3c507        | 3c905b-t4     | davicom9009   | fa312      | pcnetfastiii |
| 3c509        | 3c905b-tpb2   | davicom9102   | id1029     | rl100tx      |
| 3c529        | 3c905b-tpo    | dc21040       | id1030     | rtl8029      |
| 3c590        | 3c905b-tpo100 | dc21041       | intel21145 | rtl8139      |
| 3c595        | 3c905c-tpo    | depca         | kne110tx   | sis7016      |
| 3c595-1      | 3c980         | dfe530tx+     | ktiet32p2  | sis900       |
| 3c595-2      | 3c9805        | dlink-530tx   | lancepci   | sk_g16       |
| 3c900-t4     | 3csoho100-tx  | dlink-530tx-o | lc82c115   | smc1211      |
| 3c900-tpo    | 82559er       | dm9100        | mx98713    | smc9000      |
| 3c900b-combo | 82562em       | dm9102        | mx98715    | tiara        |
| 3c900b-fl    | 82c168        | dp83815       | mx98725    | tulip        |

|              |               |           |            |               |
|--------------|---------------|-----------|------------|---------------|
| 3c900b-tpb2  | amdhomepna    | ds21140   | mxic-98715 | via-rhine     |
| 3c900b-tpo   | an981         | ds21140a  | natsemi    | via-rhine-old |
| 3c905-combo  | ax88140       | ds21142   | ne         | wd            |
| 3c905-t4     | ax88141       | ds21143   | ne2100     | winbond840    |
| 3c905-tpo    | centaur-c     | eeopro    | ni5010     | winbond940    |
| 3c905-tpo100 | centaur-p     | eeopro100 | ni5210     | xircomtulip   |
| 3c905b-combo | compexrl100at | epic100   | ni6510     |               |
| 3c905b-fl    | compexrl2000  | exos205   | nv5000sc   |               |

## 1.2 Récupération de l'adresse MAC

Vous devez ensuite booter le terminal X avec la disquette, il va booter, chercher la carte reseau, afficher quelques informations (qui nous interessent) et commencer à chercher un serveur DHCP.

Vous allez donc avoir un affichage ressemblant à :

```
Searching for adapter..
WD8013EBT base 0x0300, memory 0x000D8000, addr 00:40:01:43:26:66
```

Ce qui nous interesse est l'adresse 00:40:01:43:26:66, notez la, elle servira à configurer le serveur dhcpd.

## 1.3 Compilation du noyau pour le Terminal X

Si vous ne connaissez pas le matériel présent dans le Terminal X, regardez au moins s'il est supporté et s'il demande des options spéciales pour le noyau, ajoutez les.

Les options de configuration du noyau sont la recherche du serveur NFS par DHCP et la racine via NFS.

### 1.3.1 Sous FreeBSD

Cette partie est ensuite spécifique à FreeBSD . Afin de compiler le noyau (installez les sources si besoin), et placez vous dans `/usr/src/sys/i386/conf`, copiez `GENERIC` en `DISKLESS`, éditez ce dernier en fonction de vos besoins, et ajoutez les options suivantes

```
options BOOTP                # Use BOOTP to obtain IP address/hostname
options BOOTP_NFSROOT        # NFS mount root filesystem using BOOTP info
options BOOTP_NFSV3          # Use NFS v3 to NFS mount root
options BOOTP_COMPAT         # Workaround for broken bootp daemons.
options BOOTP_WIRED_TO=ed0   # Use interface ed0 for BOOTP
                             # Adaptez cette dernière ligne a votre carte
                             # réseau !!!
```

Vérifiez aussi que l'option suivante est bien présente : `options NFS\_ROOT \#NFS usable`

Vous pouvez alors lancer la compilation par :

```
# config DISKLESS
# cd ../../compile/DISKLESS
# make depend && make
```

Votre noyau est prêt.

### 1.3.2 Sous linux

Vous prenez un fichier de config qui marche et avec `make menuconfig` vous ajoutez les options :

- Code Maturity -> Placez "Prompt for development..." a yes.
- Networking Options -> IP : kernel level autoconfiguration
- Networking Options -> IP : BOOTP et DHCP a yes
- File Systems -> Network File Systems -> NFS file system support et Root filesystem on NFS (options en dur, et non en modules)

Et compilez par le classique `make dep && make bzImage`, votre kernel se trouvera alors dans `/usr/src/linux/arch/i386/vmlinuz`

## Chapitre 2

# Configurer le serveur

Vous pouvez placer `/diskless` ailleurs, mais alors , veillez à changer l'emplacement dans tous les fichiers y faisant référence.

### 2.1 Configurer le serveur DHCPD

Vous devez avant tout ajouter le nom d'hôte et l'IP à votre DNS ou à votre fichier `/etc/hosts`.

Vous devez installer le serveur `isc-dhcpd` (il est dans les ports sous FreeBSD), activez son lancement en renommant le script dans `/usr/local/etc/rc.d`. Il s'agit ici de la version 2.

La configuration nécessaire se résume au strict minimum : attribuer une IP et un nom au PC portant l'adresse MAC (que vous avez noté plus haut) et lui donner l'adresse (NFS) du kernel à booter.

Le fichier de config est le suivant :

```
default-lease-time 600;
max-lease-time 7200;
authoritative;
deny unknown-clients;

option domain-name "india.ath.cx";
option domain-name-servers 192.168.0.1;
option routers 192.168.0.1;

subnet 192.168.0.0 netmask 255.255.255.0 {
    use-host-decl-names on;
    option subnet-mask 255.255.255.0;
    option broadcast-address 192.168.0.255;

    host vishnu {
```

```

    hardware ethernet 52:54:05:DF:5A:A1;
    fixed-address vishnu.india.ath.cx;
    next-server 192.168.0.1;
    option root-path "192.168.0.1:/diskless";
    filename "/diskless/kernel";
}
}

```

Vous devez remplacer les noms d'hotes, les IP et l'adresse MAC (hardware ethernet) par celles qui correspondent à votre reseau. Vous pouvez ensuite démarrer `isc-dhcpd`. Sous FreeBSD, c'est la commande `/usr/local/etc/rc.d/isc-dhcpd`, assurez vous qu'il est lancé et en cas de besoin consultez vos logs pour repérer une eventuelle faute de frappe (oubli d'un ; par exemple). Il faut noter que `dhcpd` a besoin que FreeBSD soit compilé avec `options bpf` pour fonctionner.

Une fois toute cette phase effectuée, booter votre terminal X a l'aide de la disquette et normalement, il devrait se bloquer à la procédure du chargement du noyau après avoir obtenu son IP et l'adresse du noyau qui sera chargé par NFS.

## 2.2 Créer la racine du système pour le terminal x

Il existe un script pour créer automatiquement ce dossier (même s'il est plus adapté à une station sans disque plutôt qu'à un simple terminal X), vous le trouverez dans `/usr/share/examples/diskless` sur FreeBSD.

Pour ma part, je ne l'ai pas utilisé, car, pour un simple terminal X, c'est plus simple de le faire à la main. La racine se trouve dans `/diskless`, vous pouvez la déplacer, mais vous devrez alors modifier tous les fichiers de config en fonction du nouveau chemin d'accès.

### 2.2.1 Copie des fichiers

Ici, `bin` est relatif à `/diskless` et désigne donc `/diskless/bin`. Vous avez besoin d'une racine minimale. `bin`, `dev`, `sbin` et `modules` étant identiques à ceux sur le serveur, un `cp -R` devrait donc suffire.

Sous linux, il faut bien sur copier aussi les mêmes dossier, en y ajoutant `lib`

```

# mkdir /diskless
# cd /
# cp -R /bin /sbin /modules /dev /diskless

```

### 2.2.2 Dossiers spécifiques a la racine du terminal X

Créez un dossier `root` vide et des dossiers : `var/log`, `var/db` et `var/tmp`; le dossier `tmp` est un lien symbolique vers `var/tmp`. Créez ensuite les points de montage nécessaire (juste `usr` par un simple `mkdir`). Tous ces fichiers doivent appartenir a `root`.

```
# cd /diskless
# mkdir var var/log var/tmp var/db usr root etc
# chmod 1777 var/tmp      # Défini les droits rwxrwxrwt
```

### 2.2.3 Copie du noyau

Enfin, pour permettre au Terminal X de booter, vous devez copier le noyau que vous venez de compiler dans `/diskless`, la commande pour le faire est : `cp /usr/src/sys/compile/DISKLESS/kernel /diskless`

Sous linux, récupérez le noyau que vous avez crée plus haut.

### 2.2.4 Création du dossier /etc

Ce dossier est extrêmement simple, la config du terminal X étant réduite au minimum, il contient 5 fichiers : `hosts`, `rc`, `resolv.conf`, `ttys` et `XF86Config`

#### master.passwd

Il doit aussi contenir les mots de passes (`passwd` et `shadow` sous linux, laissez juste les comptes de base). Sous FreeBSD, c'est le fichier `master.passwd` qui a ce rôle, copiez le dans le nouveau dossier `etc`, et générez les fichier `spwd.db` et `passwd` grâce a la commande `pwd_mkdb`

```
# Les commandes à exécuter sont les suivantes
cp /etc/master.passwd /diskless/etc; emacs /diskless/etc/master.passwd
# Supprimez alors les comptes utilisateurs qui ne sont pas lié au système
cd /diskless/etc
pwd_mkdb -d /diskless/etc
pwd_mkdb -p -d /diskless/etc
```

#### hosts

Voila donc le contenu de ses fichiers, le fichier `hosts` contient :

```
127.0.0.1      localhost
192.168.0.1   shiva        # Si vous n'avez pas de DNS qui se charge de
192.168.0.2   vishnu      # résolution (seulement pour shiva et vishnu)
```

**resolv.conf**

Le fichier resolv.conf contient l'adresse des DNS (si vous en avez) :

```
domain india.ath.cx
nameserver 192.168.0.1
```

**ttys**

Le fichier ttys est réduit a son strict minimum :

```
console none                                unknown off secure
ttyv0  "/usr/libexec/getty Pc"              cons25  on  secure
# Virtual terminals
ttyv1  "/usr/libexec/getty Pc"              cons25  off  secure
ttyv2  "/usr/libexec/getty Pc"              cons25  off  secure
ttyv3  "/usr/libexec/getty Pc"              cons25  off  secure
ttyv4  "/usr/libexec/getty Pc"              cons25  off  secure
ttyv5  "/usr/libexec/getty Pc"              cons25  off  secure
ttyv6  "/usr/libexec/getty Pc"              cons25  off  secure
ttyv7  "/usr/libexec/getty Pc"              cons25  off  secure
# Pseudo terminals
ttyp0  none                                  network
ttyp1  none                                  network
ttyp2  none                                  network
[...]
ttySv  none                                  network
```

Le fichier ttys est remplacé par inittab sous linux, vous devrez aussi le réduire a un minimum (voir la page de man correspondante) et faire en sorte qu'il finisse par exemple sur le niveau 3 et lance le script rc ci-dessous.

**rc**

Le fichier qui s'occupe de la procédure de boot est le script rc (qui doit être exécutable), il contient :

```
#!/bin/sh
#
PATH=/bin:/:/sbin:/usr/bin:/usr/local/bin:/usr/X11R6/bin
export PATH

# configurer les interfaces reseaux # Corriger l'IP si besoin
# Ces commandes peuvent varier sous linux, voir les HOWTO
/sbin/ifconfig lo0 127.0.0.1
/sbin/ifconfig ed0 192.168.0.2 netmask 0xffffffff00
```

```
# monter les système de fichiers via NFS # Modifier les noms au besoin
/sbin/mount shiva:/diskless /
/sbin/mount shiva:/usr /usr

# Chargement de la map fr pour le clavier
/usr/sbin/kbdcontrol -l /usr/share/syscons/keymaps/fr.iso.acc.kbd
#loadkeys french #pour linux

# Lancer Xfree en mode -query pour contacter XDMCP sur shiva
# Modifier les noms au besoin
/usr/X11R6/bin/X -query shiva -xf86config /etc/XF86Config.vishnu >/dev/nu
#
# Redémarrer lorsque l'on quitte X
    /sbin/reboot # Désactivé dans un premier temps , permet de voir les
                # erreurs et d'éviter les boots en boucle

# Voilà, j'espère que ce script marche , il faudra que je vérifie complet
```

### XF86Config

Et enfin, le fichier de config de Xfree Vous le créez avec `xf86config` et vous répondez non aux demandes d'enregistrement du fichier, jusqu'à ce qu'il vous propose de spécifier un emplacement, et alors, vous pourrez choisir `/diskless/etc/XF86Config.vishnu`

Pour que le terminal X puisse accéder à ces fichiers, il faut configurer un serveur NFS.

## 2.3 Configurer le serveur NFS

### 2.3.1 Activer le serveur NFS

Il suffit pour cela de rajouter les lignes suivantes au `rc.conf` de votre serveur d'application.

```
portmap_enable="YES"
nfs_server_enable="YES"
nfs_server_flags="-u -t -n 4"
mountd_flags="-r"
```

Vous pouvez alors soit le démarrer à la main (cf Handbook) ou alors redémarrer, ou encore passer en `init 1`. Je ne connais pas la procédure pour un serveur sous linux, donc, si vous savez comment faire, écrivez moi.

### 2.3.2 Définir les fichiers a exporter

Cela ce résume à créer un fichier `/etc/eports`, ou `vishnu` est le nom du terminal X :

```
/usr -ro vishnu
/diskless -maproot=0:0 vishnu
```

Vous devez alors imposer la relecture du fichier a `mountd` par `killall -1 mountd`, les erreurs sont enregistrées dans les logs. On ne peut pas exporter `/usr` et `/usr/diskless` par exemple. Il faut que les dossiers ne soient pas imbriqués, comme : `/usr/X11R6` et `/usr/local`.

## 2.4 Configurer le serveur XDMCP

### 2.4.1 Approche

A la fin du fichier `rc` , le Terminal X va lancer `Xfree` en mode `query` sur `shiva`. `Xfree` va donc tenter d'accéder a un serveur XDMCP sur `Shiva`. Le serveur XDMCP authentifiera alors l'utilisateur (par une fenetre XDM), et lancera ensuite le `WindowManager` défini dans `/.xsession` .

Il faut bien sur que le démon XDM soit lancé sur votre machine (soit dans un script `rc`), soit dans `ttys` ou `inittab` suivant l'OS. Dans ce dernier cas, il faut faire attention à le lancer avec l'option `-nodaemon`, ce qui donne une ligne de ce type sous `FreeBSD`

```
ttv8    "/usr/X11R6/bin/xdm -nodaemon"  xterm  on  secure
```

### 2.4.2 Configuration

Les fichiers de configurations se trouvent dans `/usr/X11R6/lib/X11/xdm`

#### **xdm-config**

C'est le fichier de config principal de XDM, on va ajouter 3 lignes :

```
!
DisplayManager.vishnu_0.setup: /usr/X11R6/lib/X11/xdm/Xsetup_0
DisplayManager.vishnu_0.startup: /usr/X11R6/lib/X11/xdm/GiveConsole
DisplayManager.vishnu_0.reset: /usr/X11R6/lib/X11/xdm/TakeConsole
```

Il suffit de remplacer `vishnu` par le nom de votre Terminal X.

Et commenter la ligne : `DisplayManager.requestPort: 0` afin de dire a XDM d'écouter les requêtes réseau

### Xaccess

Xaccess est le fichier listant les clients autorisés à se connecter.

On va juste ajouter une simple ligne

```
vishnu # Autoriser Vishnu à se connecter
```

Vishnu est aussi à changer en fonction de votre nom d'hôte.

### 2.4.3 Xservers

Si vous ne souhaitez pas voir de serveur X avec une fenêtre de logon XDM apparaitre sur votre PC tout en faisant tourner XDM pour accepter les requêtes, commentez la ligne :`0 local /usr/X11R6/bin/X`. Il n'y a pas besoin d'ajouter une ligne pour votre Terminal X, puisque celui-ci est actif sachant qu'il est lancé avec l'option `-query`

### 2.4.4 Autres fichiers

- Xresources : pour changer et améliorer le look de XDM
- Xsession : pour définir des variables pour tous les utilisateurs avant de lancer le WindowManager

### 2.4.5 Relancer XDM

Si xdm tourne déjà, un `killall -1 xdm` le relancera, mais vous serez délogué de force de votre session X en cours. Si vous venez de l'ajouter a `inittab` ou `ttys`, il suffit de relancer `init : kill -1 1`

## Chapitre 3

# Configurer un terminal servant des consoles "telnet"

Cette partie vise plutôt à récupérer des vieux PC ayant 4 Mo de RAM par exemple. Elle décrit donc comment mettre en place du Swap par NFS si vous souhaitez mettre en place cela pour votre Terminal X.

Pour le swap, on crée un fichier vide d'une certaine taille avec `dd` (`dd if=/dev/zero of=swap/swap1 bs=1024k count=16` pour un fichier de swap de 16 Mo) dans un dossier exporté par NFS, et on utilise la commande ensuite la commande `/usr/sbin/vnconfig -e /dev/vn0b /swap/swap1 swap` pour l'activer (à placer dans `etc/rc`).

### 3.1 Configurer le serveur

On configurera DHCPd et NFS de la même manière que pour un serveur dans le cas d'un terminal X.

### 3.2 Compiler un noyau

On cherche ici à configurer le noyau de manière à économiser la RAM, on peut donc utiliser un noyau épuré. Voici, ici le fichier DISKLESS que j'utilise (il ne permet pas de lancer X :

```
machine          i386
cpu              I386_CPU
ident            GANESH
maxusers         8

options          MATH_EMULATE          #Support for x87 emulation
options          INET                  #InterNETworking
options          NFS                    #Network Filesystem
```

```

options      NFS_ROOT          #NFS usable as root device, NFS r
options      COMPAT_43         #Compatible with BSD 4.3 [KEEP TH
options      _KPOSIX_PRIORITY_SCHEDULING
options      KBD_INSTALL_CDEV   # install a CDEV entry in /dev

options      BOOTP             # Use BOOTP to obtain IP address/hostname
options      BOOTP_NFSROOT     # NFS mount root filesystem using BOOTP i
options      BOOTP_WIRED_TO=ed0 # Use interface ed0 for BOOTP
                                   # Remplacer ed0 si besoin

device       isa
device       eisa

# atkbd0 controls the keyboard
device       atkbd0 at isa? port IO_KBD
device       atkbd0  at atkbd? irq 1 flags 0x1

device       vga0    at isa?
device       sc0     at isa? flags 0x100

# Floating point support - do not disable.
device       npx0    at nexus? port IO_NPX irq 13

# Support réseau # NE 2000 Isa/PCI ici.
device       miibus  # MII bus support
device       ed0     at isa? port 0x300 irq 3

pseudo-device vn          # Pour Le SWAP via NFS
pseudo-device loop       # Network loopback
pseudo-device ether      # Ethernet support

```

Recompiler donc votre noyau, et garder le pour la suite. Vous pouvez le "stripper" grace a la commande strip.

### 3.3 Créer la racine pour le client

On placera la racine dans un dossier (/diskless par exemple, et on l'exportera via NFS)

#### 3.3.1 Copie des fichiers

Ici, bin est relatif à /diskless et désigne donc /diskless/bin. Vous avez besoin d'une racine minimale. bin, dev, sbin et modules étant iden-

tiques à ceux sur le serveur, un `cp -R` devrait donc suffire.

Sous linux, il faut bien sûr copier aussi les mêmes dossier, en y ajoutant `lib`

```
# mkdir /diskless
# cd /
# cp -R /bin /sbin /modules /dev /diskless
```

### 3.3.2 Dossiers spécifiques a la racine du terminal telnet

Créez un dossier `root` vide et des dossiers : `var/log`, `var/db` et `var/tmp` ; le dossier `tmp` est un lien symbolique vers `var/tmp`. Créez ensuite les points de montage nécessaire (juste `usr` par un simple `mkdir`). Tous ces fichiers doivent appartenir a `root`.

```
# cd /diskless
# mkdir var var/log var/tmp var/db usr root etc
# chmod 1777 var/tmp      # Défini les droits rwxrwxrwt
```

### 3.3.3 Copie du noyau

Enfin, pour permettre au Terminal X de booter, vous devez copier le noyau que vous venez de compiler dans `/diskless`, la commande pour le faire est : `cp /usr/src/sys/compile/DISKLESS/kernel /diskless`

Sous linux, récupérez le noyau que vous avez crée plus haut.

### 3.3.4 Création du dossier /etc

Ce dossier est extrêmement simple, la config du terminal étant réduite au minimum, il contient une dizaine fichiers : `hosts`, `rc`, `resolv.conf`, `ttys`, `gettytab`, `master.passwd` et les fichiers qui en découlent. Copiez aussi le fichier `services` présent dans le `/etc` de votre système.

#### master.passwd

Il doit aussi contenir les mots de passes (`passwd` et `shadow` sous linux, laissez juste les comptes de base). Sous FreeBSD, c'est le fichier `master.passwd` qui a ce rôle, copiez le dans le nouveau dossier `etc`, et générez les fichiers `spwd.db` et `passwd` grâce a la commande `pwd_mkdb`. On doit ajouter un compte telnet au fichier `master.passwd`. La ligne correspondant à ce compte est `telnet:*:666:65534::0:0:Telnet autologin:/:/bin/telnet-shiva`

```
# Les commandes à exécuter sont les suivantes
cp /etc/master.passwd /diskless/etc; emacs /diskless/etc/master.passwd
# Supprimez alors les comptes utilisateurs qui ne sont pas lié au système
```

```
# Et ajouter la ligne: telnet:*:666:65534::0:0:Telnet autologin://:bin/t

cd /diskless/etc
pwd_mkdb -d /diskless/etc
pwd_mkdb -p -d /diskless/etc
```

### hosts

Voilà donc le contenu de ses fichiers, le fichier hosts contient :

```
127.0.0.1      localhost
192.168.0.1   shiva        # Si vous n'avez pas de DNS qui se charge de
192.168.0.2   vishnu       # résolution (seulement pour shiva et vishnu)
```

### resolv.conf

Le fichier resolv.conf contient l'adresse des DNS (si vous en avez) :

```
domain india.ath.cx
nameserver 192.168.0.1
```

### ttys

Le fichier ttys est réduit a son strict minimum :

```
console none                               unknown off secure
ttyv0  "/usr/libexec/getty Pc"              cons25  on  secure
# Virtual terminals
ttyv1  "/usr/libexec/getty telnet"          cons25  on  secure
ttyv2  "/usr/libexec/getty telnet"          cons25  on  secure
ttyv3  "/usr/libexec/getty telnet"          cons25  on  secure
# Vous pouvez en activer plus grace au on/off
ttyv4  "/usr/libexec/getty telnet"          cons25  off  secure
ttyv5  "/usr/libexec/getty telnet"          cons25  off  secure
ttyv6  "/usr/libexec/getty telnet"          cons25  off  secure
ttyv7  "/usr/libexec/getty telnet"          cons25  off  secure
# Pseudo terminals
ttyp0  none                                  network
ttyp1  none                                  network
ttyp2  none                                  network
[...]
ttySv  none                                  network
```

Le fichier ttys est remplacé par inittab sous linux, vous devrez aussi le réduire a un minimum (voir la page de man correspondante) et faire en sorte qu'il finisse par exemple sur le niveau 3 et lance le script rc ci-dessous. Il affichera ensuite, sur les consoles, des telnet redirigés en entrée et en sortie : telnet shiva </dev/tty2 >/dev/tty2 .

**gettytab**

Le programme getty fait appel au fichier `etc/gettytab`, que vous allez créer en recopiant le fichier présent dans `/etc`, et en y ajoutant une entrée telnet, qui lancera automatiquement un login sous le compte telnet :

```
[...]
P|Pc|Pc console:\
      :ht:np:sp#115200:

telnet|tto:\
      :al=telnet:ht:np:sp#115200:

#
[...]
```

**Le script telnet-shiva**

Le compte telnet dispose d'un script shell en tant que shell, il faut donc créer le script `bin/telnet-shiva`, et rendre ce dernier exécutable avec `chmod x bin/telnet-shiva`.

```
#!/bin/sh
# bin/telnet-shiva
clear
# Lance telnet sur shiva en mode 8bit (pour les accents)
exec /usr/bin/telnet -8 shiva
```

**rc**

Le fichier qui s'occupe de la procédure de boot est le script `rc` (qui doit être exécutable), il contient :

```
#!/bin/sh
#
PATH=/bin:/:/sbin:/usr/bin:/usr/local/bin:/usr/sbin:/usr/local/sbin
export PATH

# configurer les interfaces reseaux # Corriger l'IP si besoin
# Ces commandes peuvent varier sous linux, voir les HOWTO
/sbin/ifconfig lo0 127.0.0.1
/sbin/ifconfig ed0 192.168.0.2 netmask 0xffffffff00

# monter les système de fichiers via NFS # Modifier les noms au besoin
/sbin/mount shiva:/diskless /
```

```
/sbin/mount shiva:/usr /usr

# Chargement de la map fr pour le clavier
/usr/sbin/kbdcontrol -l /usr/share/syscons/keymaps/fr.iso.acc.kbd
#loadkeys french #pour linux

# On met en place le Swap over NFS
# On a créer un fichier vide de 16 Mo avec dd dans /diskless/swap/swap1:
# #dd if=/dev/zero of=swap/swap1 bs=1024k count=16
/usr/sbin/vnconfig -e /dev/vn0b /swap/swap1 swap

# Il faut charger les polices pour les consoles:

for i in 0 1 2 3 4 5 6 7 8 9 a b
do
vidcontrol < /dev/ttyv$i -f 8x14 iso-8x14
vidcontrol < /dev/ttyv$i -f 8x8 iso-8x8
vidcontrol < /dev/ttyv$i -f 8x16 iso-8x16
done

# On passe le relai a init et au fichier ttys
exit 0
```

Voilà, normalement, il ne vous reste plus qu'a booter votre terminal telnet et tout devrait bien se passer.

# Chapitre 4

## Quelques liens

Quelques liens qui vous permettrons d'en savoir plus et d'approfondir le système.

### 4.1 Documents génériques

Ces documents sont disponibles sur <http://www.freenix.org/linux/>.

#### 4.1.1 Application

- Abulédu Un projet basé sur l'utilisation de terminaux X pour l'informatisation des écoles. Il dispose d'une administration par le Web.
- Solupcorp - Xterminal Une solution pour la mise en place de terminaux X tournant sous Linux. (En anglais)
- Linux Terminal Server Project (En anglais)

#### 4.1.2 A propos des boot diskless

- Diskless-HOWTO Présentation rapide des solution de boot en Diskless (Cd, reseau, Eeprom) et les raisons pouvant conduire au choix d'installer une station diskless ou un terminal X. (En anglais)
- Diskless-MiniHowto (En français)
- DHCPd-MiniHowto (En français)

#### 4.1.3 A propos de NFS

- Diskless-root-NFS-HOWTO (en anglais)
- NFS-HOWTO (en anglais)
- Diskless-root-NFS-other (en anglais)
- NFS-Root-Client-MiniHowto (en français)
- NFS-Root-MiniHowto (en français)

#### **4.1.4 A propos de XDM**

- XDMCP-HOWTO (en anglais)
- X-Terminal-MiniHowto (en français)
- XDM-Xterm-MiniHowto (en français)

### **4.2 Documents spécifiques à Linux**

- Net-Howto (en français)
- Ethernet-Howto (en français)
- Net-Howto (en anglais)

### **4.3 Documents spécifiques à FreeBSD**

- Handbook-Diskless (en anglais)
- Handbook-NFS (en anglais)
- Le manuel de FreeBSD (en anglais)

# Fin

Ca y est, normalement, tout marche, autrement, relisez en vitesse le document et écrivez moi, j'ai peut-être oublié quelque chose. Et amusez vous bien en affichant les applis qui tournent sur votre nouvel Athlon XP1600+ sur votre vieux 486 qui allait partir à la casse.

Si cela ne marche pas, vérifiez que vous avez bien relancer tous les serveurs (pour qu'ils relisent leur config), consultez les logs , et assurez vous qu'il y a bien la correspondance nom d'hôte / IP pour le serveur et pour le client dans les deux fichiers `etc/hosts` ou dans le DNS.

Si vous relevez des fautes, vous pouvez toujours m'écrire à l'adresse `nlsn@free.fr`. Si vous avez des informations interessantes a ajouter, vous pouvez aussi me les envoyer.

Je tiens à remercier les personnes du forum `fr.comp.os.bsd` qui m'ont indiqué des pistes pour configurer cela. Et aussi tous les développeurs des logiciels libres (Linux, FreeBSD, XFree) pour l'excellente qualité de ces OS et logiciels.

Si vous pouvez m'envoyez des recettes de cuisine, je vous en serait reconnaissant ;-)