(*sebastien.blondeel@lifl.fr*)

Mini Howto Linux pour systèmes sans lecteur de disque(ttes)

v0.0.3 12 Sep 1996

Ce document décrit la manière de configurer un système sous Linux sans disque dur ni lecteur de disquettes. Ses droits appartiennent à Robert Nemkin, et il est placé sous les termes de la Publique Générale GNU. L'auteur remercie Bela Kis <bkis@cartan.math.klte.hu> pour avoir traduit ce document en anglais. Sébastien Blondeel, <Sebastien.Blondeel@lifl.fr> a traduit ce document en français.

Contents

1 Modifications

- v0.0.3 12 Sep 1996: Quelques erreurs mineures ont été corrigées

2 Comment configurer un système sous Linux sans lecteur de disque(ttes)

Ce document décrit la manière de configurer un système sous Linux sans disque dur ni lecteur de disquette. Il peut parfois s'avérer nécessaire de faire tourner Linux sur des ordinateurs personnels ("PC") qui ne possèdent ni disque dur ni lecteur de disquettes. Si on dispose d'un réseau, d'un autre système sous Unix avec bootp et tftp, d'un serveur NFS et d'un brûleur d'EPROM, alors il est possible de configurer et de faire tourner un système sous Linux sans disque dur ni disquette.

3 Consulter également les documents suivants:

- NFS-root Mini Howto
- Linux NET-2/3-HOWTO par Terry Dawson, `94004531@postoffice.csu.edu.au`
- `/usr/src/linux/README` pour la configuration et la compilation de nouveaux noyaux

4 Matériel

Tout ce qui est décrit ici a été testé avec la configuration suivante:

- Sun-OS 4.1.3 comme serveur d'amorçage

- Slackware 2.3 + Linux 1.2.8 + la carte ethernet wd 8013
- Un réseau ethernet en état de fonctionnement

5 Idées fondamentales

L'idée de base est la suivante: le PC va obtenir son adresse IP du serveur d'amorçage par le protocole bootp, en utilisant 0.0.0.0 comme adresse IP initiale et en obtenant son noyau par le protocole tftp.[1]

Pour cela, suivez les étapes ci-dessous.

5.1 Configurer le PC

Obtenez le paquetage nfsboot (ce paquetage est disponible sur votre site miroir de Linux préféré dans le répertoire `/pub/Linux/system/Linux-boot`). Il contient une image d'amorçage pour l'EPROM de la carte wd8013 qui peut être brûlée telle quelle.

Il y a d'autres manières de préparer le PC:

- si votre machine contient un petit disque ou un lecteur de disquette, vous pouvez utiliser le petit programme sous DOS, ou
- l'image binaire pour disquette qui se trouve dans le même paquetage.

Si vous choisissez la deuxième option, il faut utiliser la commande dd pour écrire l'image sur la disquette.

Ces images contiennent un client bootp et un client tftp. Vous devez également préparer un noyau pour linux, comportant l'option NFS-root (amorçage par NFS).

- Si vous utilisez le dernier noyau stable, linux-1.2.13, alors il faut corriger le noyau avec le fichier de correction contenu dans le paquetage nfsboot[2]
- Si vous utilisez le dernier noyau en date, instable, de la série linux-1.3.x, il vous faut configurer l'option NFS-root.

Vous pouvez ou non choisir de configurer le support pour périphérique en mode bloc (disque dur ou disquette), mais vous devez configurer le support pour tcp/ip, pour la carte ethernet wd, et pour le système de fichiers NFS. Puis recompilez le noyau de manière habituelle.

5.2 Configurer un bootpd sur le serveur

On peut le trouver dans le paquetage `bootpd-2.4.tar.gz` (qui se trouve sur votre site miroir de Linux préféré dans le répertoire `/pub/Linux/system/Network/boot.net`). Chargez le paquetage, compilez-le et installez-le. Si votre autre système sous Linux se trouve être une distribution Slackware, vous pouvez passer à l'étape suivante puisque les distributions standard comportent un bootpd. On peut démarrer le démon, soit en tapant la commande

```
          bootpd -s
```

[1]Un amorçage à travers des segments (via un routeur) n'est pas un problème simple, aussi faut-il mettre à la fois le serveur et la machine sans disque sur le même segment de réseau, ou configurer une adresse UDP d'aide dans votre routeur pointant vers l'adresse du serveur. Référez-vous au manuel de votre routeur pour de plus amples informations sur le sujet.

[2]Consulter patch(1)

soit en utilisant inetd. Dans ce dernier cas, il vous faut éditer:

- /etc/inetd.conf pour ôter le signe dièse de mise en commentaire au début des lignes suivantes:

```
# tftp   dgram   udp     wait    root    /usr/sbin/in.tftpd     tftpd /export
# bootps dgram   udp     wait    root    /usr/sbin/in.bootpd    bootpd
```

- insérer ou ôtez le signe de commentaire pour les deux lignes suivantes dans /etc/services:

```
bootps          67/tcp          # serveur BOOTP
tftp            69/udp          # serveur TFTP
```

- redémarrez inetd en tapant

```
        kill -HUP <numéro d'identification du processus de inetd>.
```

5.3 Configurer le bootpd sur le serveur.

Tout d'abord, bootpd possède un fichier de configuration qui s'appelle bootptab et qui se trouve habituellement dans /etc. Vous devez le modifier en indiquant les adresses IP de votre passerelle, de votre serveur dns, et les adresses ethernet de votre ou vos machines sans disques. Voici un fichier /etc/bootptab d'exemple:

```
global.prof:\
:sm=255.255.255.0:\
:ds=192.168.1.5:\
:gw=192.168.1.19:\
:ht=ethernet:\
:bf=linux:
machine1:hd=/export/root/machine1:tc=global.prof:ha=0000c0863d7a:ip=192.168.1.140:
machine2:hd=/export/root/machine2:tc=global.prof:ha=0800110244e1:ip=192.168.1.141:
machine3:hd=/export/root/machine3:tc=global.prof:ha=0800110244de:ip=192.168.1.142:
```

global.prof est un patron général pour les entrées d'hôtes, où

- le champ sm contient le masque pour le sous-réseau
- le champ ds contient l'adresse du serveur de nom de domaine (DNS)
- le champ gw contient l'adresse de la passerelle par défaut
- le champ ht contient le type de carte réseau
- le champ bf contient le nom du fichier d'amorçage

Après cela, chaque machine doit posséder sa propre entrée sur une ligne:

- le premier champ contient le nom de l'hôte
- le champ hd contient le répertoire du fichier d'amorçage
- on peut inclure le patron global avec le champ tc
- le champ ha contient l'adresse matérielle de la carte ethernet
- le champ ip contient l'adresse IP qui a été attribuée

5.4 Comprendre tftp

TFTP (`Trivial File Transfer Protocol`, ou protocole de transfert de fichiers banal) est un protocole de transfert de fichiers, comme ftp, mais il est beaucoup plus facile à programmer dans des mémoires de type EPROM. On peut utiliser TFTP de deux manières:

- tftp simple: cela signifie que le client peut accéder à la totalité de votre système de fichiers. C'est plus simple, mais cela constitue un gros trou de sécurité (n'importe qui peut obtenir votre fichier de mots de passe par tftp).
- tftp sécurisé: le serveur tftp utilise un appel système chroot.2 pour modifier son propre répertoire racine. Tout ce qui n'est pas dans cette racine sera absolument inaccessible. Comme le répertoire chroot devient le nouveau répertoire racine, le champ hd listé dans le fichier bootptab doit prendre cette situation en compte. Par exemple: lorsqu'on utilise tftp non sécurisé, le champ hd contient le chemin complet menant au répertoire d'amorçage: `/export/root/machine1`. Lorsqu'on utilise tftp sécurisé avec /export comme répertoire racine, alors /export devient / et le champ hd doit être `/root/machine1`.

Comme pratiquement toute installation Unix comporte un serveur tftp, vous n'aurez probablement pas besoin d'installer la votre.

5.5 Configurer un Linux minimal sur le serveur distant.

Il vous faut pour cela, par exemple, les paquetages a, ap, n et x de la distribution Slackware. Il est possible d'installer plus de choses; ce pendant les paquetages ci-dessus suffiront à l'installation d'un terminal X sans disque. Pour l'installation, il vous faut un système sous Linux en état de marche. Trouvez un peu d'espace disque sur la machine distante et exportez-le en lecture et en écriture. Montez le répertoire exporté quelque part (par exemple sur /mnt) sur le système de fichiers du système sous Linux. Démarrez Linux et modifiez l'option de racine dans la configuration; remplacez / par /mnt. Puis configurez les paquetages ci-dessus de manière habituelle. Si vous ne souhaitez faire tourner qu'un seul Linux sans disque, il ne vous faut rien modifier d'autre. D'un autre côté, si vous pensez utiliser plus d'une machine sans disque, la configuration décrite ci-dessus ne fonctionnera pas parce que certains fichiers et répertoires doivent être privés pour chaque machine. On peut contourner ce problème on déplaçant le répertoire /usr (il ne contient aucune donnée personnelle) et ensuite de créer un sous-répertoire pour chaque machine sans disque. Par exemple, si /export/linux/machine1 est monté sur /mnt alors la structure des répertoires après la configuration initiale ressemblera à:

```
/export/linux/machine1/bin
/export/linux/machine1/sbin
/export/linux/machine1/lib
/export/linux/machine1/etc
/export/linux/machine1/var
/export/linux/machine1/usr
```

Après les modifications vous obtiendrez

```
/export/linux/machine1/bin
/export/linux/machine1/sbin
/export/linux/machine1/lib
/export/linux/machine1/etc
/export/linux/machine1/var
/export/linux/usr
```

Maintenant créez les sous-répertoires pour les autres machines. Supposons pour l'instant que vos machines sans disque s'appellent machine1, machine2, machine3, etc.; vous pouvez alors utiliser le script bash qui suit pour configurer les autres répertoires:

```
cd /export/linux
for x in machine2 machine3; do
        mkdir $x; cd $x
        (cd ../machine1; tar cf - *) | tar xvf -
done
```

Puis exportez les répertoires qui suivent:

- /export/linux/usr en lecture seule pour tout le monde.
- /export/linux/machine1 uniquement sur machine1, en lecture/écriture et avec les droits de root.
- /export/linux/machine2 idem, sur machine2.
- /export/linux/machine3 idem, sur machine3.

comme suit[3]:

```
# Ce fichier est /etc/export
# pour des terminaux sous le système Linux distants
# Écrit par Buci
# N'écrivez cette ligne qu'une fois
/export/root/usr             -access=linuxnet
# N'écrivez ces lignes qu'une fois pour chaque hôte
/export/root/machine1       rw=machine1,root=machine1
/export/root/machine2       rw=machine2,root=machine2
/export/root/machine3       rw=machine3,root=machine3
```

N'oubliez pas de lancer exportfs -a.

5.6 Configurer le serveur tftp

C'est maintenant le moment de configurer le serveur tftp. Si vous n'avez pas besoin de tftp sécurisé alors tout est très simple puisque vos clients peuvent être amorcés depuis le répertoire /export.

Si vous utilisez un tftp sécurisé vous pouvez soit mettre en place une structure de répertoire /export/linux complète sous /tftpboot (en n'utilisant qu'un seul véritable noyau et des liens symboliques pour les autres machines), ou laisser le répertoire /export jouer le rôle du répertoire d'amorçage pour le tftpd sécurisé. Ou encore, si vous disposez d'un répertoire tftpboot séparé, de façon similaire, vous n'aurez besoin que d'un seul noyau dans la structure de répertoires d'origine, et de liens pour les autres. Vous pouvez obtenir ce résultat en tapant ce qui suit:

```
mkdir -p /tftpboot/export/linux/machine1
cd /tftpboot/export/linux/machine1
cp /export/linux/machine1/<nom du noyau>.
```

Puis tapez ce qui suit:

```
mkdir -p /tftpboot/export/linux/machine2
cd ../machine2
ln -s ../machine2/<nom du noyau>
```

[3] le format de cet exemple est conforme à la syntaxe des exportations de fichiers pour SunOS 4.1.3

5.7 Derniers réglages

Enfin, il vous faut insérer

```
        /sbin/mount nfs_server:/export/linux/usr /usr
```

à la première ligne de

```
        /export/linux/<machinex>/etc/rc.d/rc.S
```

où <machinex> signifie machine1, machine2, etc.

6 Mémoire et espace disque requis; vitesse

. Je n'ai testé ceci que pour la distribution Slackware 2.3; pour d'autres distributions ou versions les nombres qui suivent peuvent varier:

- Espace disque: 28Mo + 6.5Mo/machine
- RAM: J'utilise X avec 8Mo. Comme il ne faut que 8Mo de système de pagination sur mémoire de masse, on peut les mettre en place, je pense – de façon séparée pour chacune des machines – dans /tmp. N'oubliez pas de lancer mkswap.
- Vitesse: Je n'ai pas eu de problèmes sur un 486 DX2/66 avec 8 Mégaoctets.

7 Erreurs possibles

- J'ai découvert une erreur étrange: dans le sous-répertoire /dev, SunOS a corrompu les entrées de périphériques de telle sorte que j'ai dû relancer MAKEDEV en montant le sous-répertoire sur un système sous Linux avec disque. (La raison de cela provient des différences entre le NFS de linux et le NFS de SunOS: tous deux utilisent 32bits pour les numéros de périphériques Mineur et Majeur, mais linux utilise des champs de 16bits pour ces deux numéros, alors que SunOS utilise un champ de 14bits pour le numéro de périphérique Majeur, et un champ de 18bits pour le numéro de périphérique Mineur.)
- Quand on amorce un système sous Linux sans disque, la table de routage au serveur tftp ne contient qu'un seul routage, et vous devez configurer des tables de routage correctes. Vous avez pour cela deux possibilités:
  - configurer le rc.S de chacune des machines à la main
  - utiliser un paquetage de client bootp et rédiger un script de configuration généralisé

8 Erreurs et développements possibles de ce document

- Citer correctement les documents liés à tout ceci.
- SunOS est fondé sur BSD. Il faut inclure une configuration de serveur fondée sur SVR4 (c'est-à-dire sur Solaris).
- Même si Linux ressemble beaucoup à SunOS en tant que serveur bootp/tftp, il peut être utile de fournir un exemple de serveur fondé sur Linux.

- Mettre à jour ce document pour le paquetage etherboot en cours.
- Montrer les différences entre le noyau version 1.2.13 corrigé pour la racine NFS et le dernier noyau 1.3.x, qui contient la correction de racine par NFS.
- Besoin d'essayer d'autres cartes ethernet que la wd8013
- Inclure des renseignements de configuration pour bootpc, un client bootp pour Linux qui sert à configurer des tables de routage correctes.
- Fautes de frappe et autres: notifiez-les, s'il vous plaît, à `buci@math.klte.hu` ou à `Sebastien.Blondeel@lifl.fr` pour la traduction française. Merci.