Linux NET-2/NET-3 MODE D'EMPLOI

Terry Dawson, <94004531@postoffice.csu.edu.au>
Traduction française par Frédéric Lepied, fred@sugix.frmug.fr.net et Jacques Chion, jimmy@cwoux.remcomp.com
v3.4, 21 Nov 1995

Ce document décrit comment obtenir, installer et configurer les ensembles logiciels Linux NET-2 et NET-3. Quelques questions et leurs réponses sont également incluses. Ce document n'est pas un cours sur tcp/ip, bien que, lorsque c'est possible, des informations de ce type soient incluses. Des renvois à d'autres documentations enseignant les principes de tcp/ip sont listés.

Table des matières

1 Introduction

Ce document est une réécriture complète de l'ancienne NET-FAQ et fait suite au document NET-2-HOWTO version 1.0+, en ce qui concerne les nouvelles couches réseau tcp-ip appelées NET-2, et NET-3 pour les noyaux Linux 1.0 et suivants.

1.1 Changements depuis la version précédente

```
Ajouts_:
        References a Caldera Netware Client sous forme de questions.
        Informations relatives au developpement de X25.-Merci a Ian Brace.
Corrections_:
        Mise a jour du nouveau site pour les outils net-tools.
        Mise a jour de sliplogin a la version 2.0.
        Mise a jour de la section des fichiers rc pour mise en accord avec la convention Debian.
        Incluse une configuration PPP plus reactive aux appels entrants.
        Suppression de nombreuses virgules.
        Nouvelle notice de droits d'auteur.
```

1.2 Bref historique du développement de la couche réseau dans Linux

Ross Biro <biro@yggdrasil.com> a écrit le code réseau originel du noyau Linux. Il a utilisé les pilotes ethernet écrits par Donald Becker <becker@cesdis1.gsfc.nasa.gov>, un pilote *slip* écrit par Laurence Culhane <loz@holmes.demon.co.uk> et un pilote D-Link écrit par Bj0rn Ekwall <bj0rn@blox.se>.

La suite du développement du code réseau a été reprise par Fred van Kempen <waltje@hacktic.nl>, qui prit le code de Ross et qui produisit la version appelée *NET-2*. NET-2 a évolué plusieurs fois jusqu'à la version NET-2d. Ensuite Alan Cox <iialan@iifeak.swan.ac.uk> a pris le code de Fred dans le but de produire une version stable et fonctionnelle. Cette version fut appelée *NET-2D(debugged)* et incorporée aux versions standards du noyau quelque temps avant que la version 1.0 ne sorte.

Le support PPP a été ajouté par Michael Callahan, <callahan@maths.ox.ac.uk> et Al Longyear, <longyear@netcom.com>. Au début en tant que patches du noyau puis dans les versions suivantes en tant qu'option.

La dernière version du code, NET-3, existe depuis la version 1.1.5. C'est en fait essentiellement la même version que la précédente avec beaucoup de corrections et d'améliorations.

Alan a ajouté des fonctionnalités comme les modules IPX et AX.25. Florian La Roche, <flla@stud.uni-sb.de> a produit une version mise à jour de la distribution des applications réseau. NIIBE Yukata a amélioré le pilote PLIP.

Jonathan Naylor a repris le développement sur le code AX.25 et y a ajouté de nombreuses choses dont le support NetRom. De nombreuses personnes ont contribué à éliminer les bogues, à apporter des applications et à écrire des pilotes.

2 Mise en garde

Le code réseau de Linux est une nouvelle implémentation des couches TCP/IP. Il a été entièrement repensé et ne résulte pas du portage du code réseau de quelque noyau que ce soit. Le code réseau Linux est l'un des codes réseaux les plus récents et des plus innovateurs. Il y a beaucoup de développeurs y travaillant, pour y apporter de nouveaux services, et en raisant de cela il grossit rapidement. Il est en constante évolution et il peut rester des bogues et subsister des problèmes, de là sont publiés de nombreuses corrections et patches. Si vous êtes gênés par un problème, il vous suffit de coller à la version du code réseau mise à jour avec les versions standards du noyau et des jeux d'utilitaires. Ces versions standards de noyau possèdent un chiffre **pair** comme second numéro, 1.2.7 est donc par exemple une version standard. Les versions avec un nombre impair comme second numéro sont au stade **alpha** et vous risquez de rencontrer des problèmes ou des bogues, étant donné que ce sont des versions d'**essai**. Une petite équipe se consacre à la programmation du code réseau, elle est aidée dans sa tâche par un groupe d'un millier de programmeurs testant le code, collectant et rapportant les bogues et problèmes. Un problème que vous rencontrez peut avoir déjà été découvert, et être en cours de correction, alors soyez patient ; mais si vous le pouvez n'hésitez pas à nous aider.

Nous ne savons pas et nous ne pouvons pas connaître tout ce qu'il y a à savoir à propos du logiciel réseau de Linux. Merci de comprendre que ce document contient certainement des erreurs. Pour une information plus détaillée et précise, veuillez lire tous les fichiers README inclus dans les nombreux morceaux de code, mentionnés dans ce document. Autant que faire se peut, nous essaierons de maintenir ce document sans erreur et à jour. Les versions de logiciel citées sont les versions courantes à l'heure où nous écrivons.

NOTE : Bien que son nom apparaisse dans *Berkeley Software Distribution NET-2 release*, le code réseau Linux n'a en fait rien à voir. Merci de ne pas les confondre.

3 Déjà des questions ?

La seule question stupide est celle qui n'a pas été posée.

Si vous avez des questions générales à propos de la configuration, et si vous n'avez pas réussi à trouver de réponse après avoir lu les différents HOWTO et FAQ, alors vous serez mieux servi en postant vos questions dans le newsgroup *comp.os.linux.help*. Ou encore si vous pensez que votre question concerne tout particulièrement le code réseau Linux, vous pouvez la poster à la mailing list NET. **Merci d'y inclure toutes informations pertinentes**, il n'y a rien de plus énervant qu'un rapport sur un bogue ou un problème, impossible à élucider faute d'information sur le contexte.

Les numéros de version et de révision du code, une description détaillée du problème, ainsi que les circonstances qui l'ont fait apparaître, sont essentiels. Les messages de trace et de debug quand ils sont disponibles, doivent être considérés comme obligatoires.

Si vous avez des questions se rapportant à la configuration, ou un problème que vous avez eu avec **n'importe quelle** distribution linux, quel que soit votre fournisseur, merci de vous adresser à la personne qui a créé la distribution en premier lieu, plutôt que de porter votre problème à l'attention des développeurs du code réseau. Nous insistons sur ce point car quelques distributions utilisent des structures non-standards de répertoires et proposent des versions de code et d'utilitaires en version de test ou non-standards. Les développeurs du code NET-2 ne peuvent pas offrir plus de support pour le code réseau que ce qui est décrit dans ce document ou dans les instructions des distributions Alpha/Beta.

Pour rejoindre le canal **linux-net** du serveur de courrier électronique dédié à Linux envoyez un courrier à :

```
Majordomo@vger.rutgers.edu
```

avec la ligne :

```
suscribe linux-net
```

au début du message (pas dans la ligne de sujet).

Gardez à l'esprit que le canal **linux-net** est uniquement réservé aux discussions sur le développement du code réseau. Il existe aussi une liste PPP. Pour la rejoindre, effectuez la même procédure en remplaçant `linux-net` par `linux-ppp` dans le corps du message.

Il existe aussi une liste pour discuter des programmes relatifs aux radio amateurs. Pour la rejoindre, effectuez la même procédure en remplaçant `linux-net` par `linux-hams` dans le corps du message.

4 Documentation utile

Si vous recherchez des informations sur TCP/IP que ce document ne couvre pas, vous pouvez regarder dans les documents qui suivent.

`Olaf Kirch` a écrit un document important faisant partie du *Linux Documentation Project* appelé *Linux Network Administration Guide*. C'est un excellent document. Il couvre tous les aspects de TCP/IP sous Linux, depuis la mise en marche jusqu'à l'utilisation, en incluant NFS, UUCP, mail, News, nameserver etc.

Le livre d'Olaf complète cette HOWTO. Il couvre l'installation et la configuration du code réseau, c'est à dire 'Comment mettre votre machine sur le réseau'. Si vous êtes nouveau dans le monde réseau UNIX, je vous recommande fortement d'en obtenir une copie et de commencer par la lire. Il répondra à beaucoup de questions que vous vous posez qui sont hors de propos dans ce document.

La version courante est disponible à :

sunsite.unc.edu

```
/pub/Linux/docs/linux-doc-project/network-guide/*
```

Il existe plusieurs formats disponibles de ce document dans le répertoire. La plupart des formats sont supportés: ascii, Postscript, DVI, Latex et groff.

Le copyright © du *Linux Network Administrators Guide* appartient à Olaf Kirch.

Il y a maintenant un certain nombre de sociétés publiant de la documentation sur Linux, aussi si vous voulez éviter de retrouver et d'imprimer vous-mêmes le document, vous n'aurez aucun mal à trouver l'ouvrage d'Olaf sous forme reliée dans les bonnes librairies.

Vous devriez aussi lire les autres HOWTO qui ont un lien avec le réseau sous Linux :

(ftp://sunsite.unc.edu/pub/Linux/docs/HOWTO/Ethernet-HOWTO.html)
que vous devez lire si vous voulez utiliser une carte ethernet sous Linux. Elle comprend beaucoup de détails sur la façon de sélectionner, installer et configurer une carte ethernet sous Linux.
(http://sunsite.unc.edu/mdw/HOWTO/PPP-HOWTO.html) si vous voulez utiliser PPP.
(http://sunsite.unc.edu/mdw/HOWTO/Serial-HOWTO.html) si vous avez l'intention d'utiliser SLIP ou PPP en mode serveur.
(http://sunsite.unc.edu/mdw/HOWTO/NIS-HOWTO.html) si vous êtes intéressés par le Network Information Service de Sun.
(http://sunsite.unc.edu/mdw/HOWTO/HAM-HOWTO.html) si vous êtes intéressés par la configuration et l'utilisation de logiciels radio-amateurs.
(http://sunsite.unc.edu/mdw/HOWTO/Mail-HOWTO.html) et (http://sunsite.unc.edu/mdw/HOWTO/News-HOWTO.html) pour des informations spécifiques sur le courrier et les conférences électroniques sur votre système.

(http://sunsite.unc.edu/mdw/HOWTO/UUCP-HOWTO.html) si vous voulez être relié au réseau par UUCP.
(http://sunsite.unc.edu/mdw/HOWTO/Firewall-HOWTO.html) si vous voulez construire une passerelle sous Linux pour votre réseau.

Si vous recherchez des informations générales de base sur les réseaux, je vous recommande les documents suivants:

Introduction à tcp-ip

version texte: (ftp://athos.rutgers.edu/runet/tcp-ip.intro.doc) version postscript: (ftp://athos.rutgers.edu/runet/tcp-ip.intro.ps)

Administration tcp-ip

version texte: (ftp://athos.rutgers.edu/runet/tcp-ip-admin.doc) version postscript: (ftp://athos.rutgers.edu/runet/tcp-ip-admin.ps)

Si vous recherchez des informations détaillées sur les couches TCP/IP, je vous recommande fortement :

```
"Internetworking with TCP/IP"
by Douglas E. Comer

ISBN 0-13-474321-0
Prentice Hall publications.
```

Si vous voulez apprendre comment écrire des applications pour réseau dans un environnement compatible Unix, je vous recommande alors chaudement:

```
"Unix Network Programming"
by W.Richard Stevens

ISBN 0-13-949876-1
Prentice Hall publications.
```

4.1 Nouvelles versions de ce document

La dernière version de ce document peut être récupérée par ftp anonyme sur:

sunsite.unc.edu

```
/pub/Linux/docs/HOWTO/NET-2-HOWTO
```

ou:

```
/pub/Linux/docs/HOWTO/other-formats/NET-2-HOWTO.{tex,ps,dvi}
```

Ou à partir du World Wide Web au `(http://sunsite.unc.edu/mdw/linux.html)`, à la page : `(http://sunsite.unc.edu/mdw/HOWTO/NET-2-HOWTO.html)` ou directement par moi, <`94004531@postoffice.csu.edu.au`>. Elle sera aussi postée régulièrement dans les groupes de conférence : `comp.os.linux.networking`, `comp.os.linux.answers`, et `news.answers`.

Vous pouvez trouver les FAQ de `news.answers`, celle-ci incluse, archivées à `rtfm.mit.edu:/pub/usenet`.

4.2 Retour d'informations.

Envoyez-moi tous vos commentaires, mises à jour ou suggestions à <`94004531@postoffice.csu.edu.au`>. Plus tôt j'aurai le retour, plus tôt je pourrai effectuer les mises à jour et les corrections nécessaires. Si vous voyez des problèmes, envoyez-moi un courrier plutôt que de poster dans une conférence que je pourrais manquer. Vous pourrez aussi me contacter sur IRC , avec *terryd* comme nom sur le canal IRC `#linpeople`.

5 Quelques définitions de termes utilisés dans ce document

Vous verrez souvent les termes `client` et `serveur` utilisés dans ce document. Ce sont plutôt des termes utilisés de manière spécifique, mais ici j'ai généralisé leur définition dans le sens suivant:

client

La machine ou le programme qui initie une action ou une connexion dans le but d'utiliser un service ou des données.

serveur

La machine ou le programme qui accepte des connexions entrantes de plusieurs machines éloignées et leur fournit un service ou des données.

Ces définitions ne sont pas très exactes mais elles permettent de distinguer les extrémités de systèmes avec appariement, tels que *SLIP* et *PPP* qui n'ont en fait ni clients ni serveurs.

D'autres termes que vous verrez:

Adresse IP

C'est un nombre qui identifie de manière unique un hôte TCP/IP sur le réseau. L'adresse comprend 4 octets et est représentée habituellement sous la forme "notation décimale avec points", ou chaque octet est représenté sous forme décimale avec des points entre eux.

Adresse matérielle

C'est un nombre qui identifie de manière unique un hôte dans un réseau physique au niveau de la couche moyens d'accès. Exemples: *Adresses Ethernet* et *Adresses AX25*.

Datagram

Un datagramme est un ensemble discret de données et d'en-têtes contenant des adresses, et qui est l'unité de base de transmission sur un réseau IP.On l'appelle aussi 'paquet'.

MTU

Le 'Maximum Transmission Unit' (*MTU*) est un paramètre qui détermine le plus grand datagramme transmissible par une interface IP sans qu'il soit nécessaire de le segmenter en unités plus petites. Le MTU doit être plus grand que le datagramme le plus grand que vous voulez transmettre sans fragmentation. A noter qe cela vous permet de vous protéger des fragmentations au niveau local, certains liens sur le chemin pouvant avoir un MTU plus petit, fragmentant alors le datagramme. Les valeurs classiques sont de 1500 octets pour une interface ethernet et 576 octets pour une interface SLIP.

MSS

Le Maximum Segment Size *MSS* est la plus grande quantité de données qui peut être transmise en une seule fois. Si vous voulez éviter une fragmentation au niveau local MSS doit être égal à MTU moins l'en-tête IP.

Fenêtre (window)

La fenêtre *window* est la plus grande quantité de données que le point de réception peut recevoir à un moment donné.

route

La *route* est le chemin que vos datagrammes suivent sur le réseau pour atteindre leur destination.

ARP

C'est l'acronyme de *Address Resolution Protocol* et c'est la manière dont une machine associe une adresse IP avec une adresse matérielle.

6 Fonctionnalités NET-3 supportées

Le code du réseau (NET) est une implémentation complète de TCP/IP pour Linux.

6.1 Support général

Les versions récentes de NET-3 supportent :

Cartes ethernet

La plupart des cartes ethernet sont supportées, y compris des adaptateurs de poche portables et PCI.

SLIP (Serial Line IP) et PPP

pour faire du TCP/IP sur des lignes séries comme le téléphone avec un modem ou bien un simple câble entre deux machines.

Van Jacobsen Header Compression

qui compresse les en-têtes TCP/IP pour augmenter les performances de SLIP sur les lignes à bas débits.

PLIP (Parallel Lines IP)

qui permet de faire communiquer deux machines en les reliant par leurs ports parallèles.

Equilibrage de charge EQL

qui permet d'utiliser deux (ou plus) liens pour connecter votre machine à une autre , ou bien à Internet (si votre fournisseur le permet) pour doubler effectivement la bande passante. Les nouveaux noyaux le supportent.

NFS (Networked File System)

qui permet de monter des systèmes de fichiers situés sur d'autres machines via un réseau.

IPX (Novell)

qui vous permet d'écrire des applications IPX locales , ou d'utiliser Linux comme routeur IPX.

Network Information System de Sun (NIS)

Une implémentation NIS a été portée sur Linux qui devrait vous encourager à l'utiliser.

ARCNet

Un pilote ARCNet a été écrit et inclus dans des noyaux récents. Il ne devrait pas être aussi rapide qu'ethernet mais les cartes sont beaucoup moins chères.

Token Ring d'IBM

qui permet à Linux d'être installé sur un réseau Token Ring. Un pilote expérimental Token Ring est inclus dans les derniers noyaux.

Appletalk

ou bien Ethertalk? Quoi qu'il en soit, je pense que cela vous permettra de partager fichiers et imprimantes avec votre Macintosh. Voir 'modules expérimentaux et en développement' ci-dessous.

Support carte réseau Wavelan Wireless

qui vous permet d'utiliser votre machine Linux de manière autonome ou à quelque distance du réseau. Un support pour Wavelan Wireless est maintenant inclus dans les noyaux récents.

ISDN

Il existe un support expérimental pour quelques cartes ISDN. Voir 'modules expérimentaux et en développement' ci-dessous.

ATM

Il existe une équipe de programmeurs travaillant sur un support ATM pour Linux.

Protection firewall IP

pour vous aider àconfigurer votre machine Linux en tant que passerelle sûre.

Comptabilité IP

vous permet de savoir qui utilise, et combien de fois, votre réseau.

Percement de tunnels IP (tunnelling)

pour l'expérimentation IP itinérante.

Le code réseau NET-3 ne supporte pas encore:

SPX/NCP (Novell Netware)

qui permet à Linux de servir et de monter des systèmes Novell ou d'utiliser des imprimantes Novell. Le travail est en cours mais en raison du caractère breveté du produit, cela prendra du temps.

FDDI

Je ne connais pas actuellement de support Linux pour les cartes FDDI.

Chemins de transfert de données (streams) System V

Il y a une équipe qui travaille dessus. Voir détails plus loin.

6.2 Cartes ethernet supportées

La version 1.2.0 du noyau linux supporte les cartes ethernet suivantes:

- WD80*3 et proches compatibles.
- SMC Ultra.
- AMD LANCE et PCNet (AT1500 et NE2100) et proches compatibles.
- 3c501 (dépassée et très lente).
- 3c503 et proches compatibles.
- 3c505.
- 3c507.
- 3c509/3c579.
- Cabletron E21xx.
- DEPCA et proches compatibles.
- Etherworks 3.
- ARCNet.

- AT1700 (pas les clones).
- EtherExpress.
- NI5210 et proches compatibles.
- NI6510.
- WaveLAN.
- HP PCLAN+ (27247B et 27252A).
- HP PCLAN (27245 et autres séries 27xxx).
- NE2000/NE1000 et proches compatibles.
- SK_G16.
- Ansel Communications EISA 3200.
- Apricot Xen-II sur carte ethernet.
- DE425, DE434, DE435.
- Zenith Z-Note.
- Adaptateur de poche AT-LAN-TEC/RealTek.
- Adaptateur de poche D-Link DE600.
- Adaptateur de poche D-Link DE620.

Les dernières versions des noyaux permettent de supporter une grande variété de cartes. Si vous avez l'intention d'utiliser une carte ethernet avec Linux je vous conseille de consulter `(http://sunsite.unc.edu/mdw/HOWTO/Ethernet-HOWTO.html)` qui contient beaucoup d'informations sur les cartes ethernet supportées, y compris laquelle choisir si vous devez en acheter une spécialement pour Linux.

Comme mentionné précédemment, Linux supporte d'autres moyens de se connecter en réseau si vous n'avez pas de carte ethernet ou de connexion directe. Beaucoup d'universités et de sociétés proposent des accès au réseau à partir du téléphone. En général, les formes d'accès proposées sont des accès SLIP ou PPP. Tout ce dont vous avez besoin, c'est d'un modem. Celui que vous possédez peut-être déjà peut très bien convenir. Il suffit de configurer Linux de manière adéquate. Il y a des chapitres traitant de ces solutions dans la suite du document.

6.3 Support pour radio-amateurs.

Linux supporte maintenant des systèmes spécifiques pour radio-amateurs. Les derniers noyaux alpha sont maintenant distribués avec des supports standards pour:

Support AX.25

Alan Cox et Jonathan Naylor ont travaillé sur un support socket AX25 pour le noyau.

Support NetRom

Jonathan Naylor a développé un support NetRom pour le noyau. Il est toujours en phase expérimentale mais il progresse bien.

Carte Ottawa PI

Un pilote évolué pour la carte OttaWa PI a été développé par Dave Perry du groupe packet-radio d'Ottawa.

Pilote générique pour carte SCC

Un pilote générique pour cartes SCC est maintenant inclus dans les noyaux alpha, développé par Joerg Reuter.

Pilote de lien réseau pour noyau

en vue de relier les programmes utilisateurs au noyau sans problèmes.

D'autres détails sur les supports radio-amateurs peuvent trouvés dans (`http://sunsite.unc.edu/mdw/HOWTO/HAM-HOWTO.html`).

7 Assembler les logiciels NET-2/NET-3.

Avant de configurer le logiciel réseau, vous devez récupérer toutes les parties le composant. Vous aurez besoin de la version en cours du noyau (1.0 ou plus récente), les bibliothèques système correctes, les fichiers et programmes de configuration de TCP/IP (par exemple /sbin/ifconfig, /etc/hosts...) et bien sûr les applications réseau (comme telnet, ftp, rlogin...).

Si vous avez obtenu Linux à partir d'une distribution, vous possédez probablement tout ce dont vous avez besoin. Vérifiez et soyez certains de ce que vous faites. Par exemple, certaines distributions de Linux viennent avec toutes les fichiers de configuration réseau, les fichiers binaires, les bibliothèques et le noyau installé, dès lors il n' y a pas de raison d'assembler les fichiers qui vont suivre.

NOTE : il peut y avoir des différences entre les fichiers et les répertoires décrits dans cette HOWTO et ceux de votre distribution.

Si vous **AVEZ** les logiciels réseau, vous pouvez passer directement à la section 'Configurer le noyau'. Si vous **N'AVEZ PAS** les logiciels réseau suivez les indications ci-dessous.

7.1 Les sources du noyau

Les versions 1.2.* du noyau Linux sont les versions de *référence*. Toutes les versions 1.3.* sont des versions en développement. Si vous ne voulez pas modifier le noyau vous-même, vous devez utiliser uniquement les versions 1.2.*, elles feront la majeure partie de ce que vous voulez. En ce qui concerne le code réseau, je vous recommande vivement de prendre une profonde respiration et de regarder les versions les plus récentes, parce qu'il peut y avoir des modifications importantes de la partie réseau du noyau. Je sais que vous l'entendez partout mais quand vous installez une nouvelle version du noyau assurez-vous d'avoir une sauvegarde au cas où quelque chose fonctionnerait vraiment de travers.

La version courante du noyau se trouve à :

ftp.funet.fi

```
/pub/OS/Linux/PEOPLE/Linus/v1.2/linux-1.2.13.tar.gz
```

C'est un fichier compressé ; vous aurez besoin de *gzip* pour le décompresser.

Pour l'installer, faire :

```
# cd /usr/src
# mv linux linux.old
# gzip -dc linux-1.2.13.tar.gz | tar xvf -
```

Vous trouverez aussi des fichiers appelés `patch-1.2.1.gz...` dans le même répertoire. Ce sont des patches. Si votre version du noyau Linux est 1.2.1 cela signifie que vous avez un noyau Linux version 1.2.0 avec le patch 1 déjà appliqué. S'il y a des patches avec des numéros de version supérieurs au vôtre, vous devez récupérer **tous** les patches intermédiaires et les appliquer dans l'ordre avec les commandes suivantes :

```
# cd /usr/src
# for patchfile in .../patch*
> do
> gzip -dc $patchfile | patch -p0>>patch.errs
> done
```

...

Vérifiez le fichier de sortie (patch.errs) et chercher la chaîne *fail*. Si vous ne la trouvez pas, tout s'est bien passé. Si vous la trouvez, au moins un des patches ne s'est pas appliqué correctement. Alors vous repartez d'une archive de noyau sûre et vous appliquez les patches un à un jusqu'à ce que vous trouviez le patch incriminé. Si vous ne vous en sortez pas, renseignez-vous.

7.2 Les bibliothèques

Vous aurez besoin au moins de la version 4.4.2 de la *libc*, parce qu'il y a des problèmes avec les versions précédentes qui perturbent les masques de sous-réseaux.

La version courante (libc-4.6.27) se trouve à :

sunsite.unc.edu

```
/pub/Linux/GCC/
```

Vous aurez aussi besoin des fichiers suivants :

- image-4.6.27.tar.gz
- inc-4.6.27.tar.gz
- extra-4.6.27.tar.gz

- release.libc-4.6.27

Vous **DEVEZ** impérativement lire **release.libc-4.6.27** avant d'installer les bibliothèques.

Pour utiliser la version 4.5.26 vous devrez avoir au moins GCC version 2.6.2 et un noyau Linux version 1.1.52 ou des versions supérieures.

7.3 Les outils de configuration du réseau

Vous aurez besoin d'utilitaires pour configurer le support réseau.

La version courante NET-2 est disponible à :

ftp.linux.org.uk

```
/pub/linux/Networking/PROGRAMS/NetTools/net-tools-1.2.0.tar.gz
```

La partie réseau du noyau évoluant encore, vous devrez choisir la version des outils réseau qui correspond à celle de votre noyau.

Les noms des fichiers reflètent la version du noyau avec lesquels ils fonctionnent. Choisissez celui dont la version est inférieure ou égale à la version du noyau que vous voulez utiliser.

Pour fabriquer et installer les outils, faire :

```
# cd /usr/src
# mkdir net-tools
# cd net-tools
# gzip -dc net-tools-1.2.0.tar.gz | tar xvf -
# make
```

Le script `Configure.sh` tournera automatiquement. Si tout est ok, faire :

```
# make install
```

Si vous utilisez une version du noyau inférieure à 1.1.26 vous devez chercher à :

ftp.linux.org.uk

```
/pub/linux/Networking/PROGRAMS/Others/net032
```

Dans ce répertoire, vous trouverez trois versions des outils réseau. La table suivante vous indique quelle version de net-032 choisir, selon votre version du noyau :

```
net-0.32d-net3.tar.gz     1.1.12+
net-0.32b.tar.gz          1.1.4+
net-0.32.old.tar.gz       pre 1.1.4 kernels
```

Ces archives contiennent l'essentiel des programmes de configuration réseau comme *ifconfig*, *route*, *netstat* etc. Ils seront décrits par la suite.

7.4 Les applications réseau

Vous voudrez sûrement des applications réseaux. Des programmes comme *telnet*, *ftp*, *finger* et leurs démons par exemple. `Florian La Roche, <flla@stud.uni-sb.de>` a fabriqué une distribution pratiquement complète des applications réseau sous forme binaire et source. Ces applications TCP/IP et des exemples de fichiers de configuration se trouvent à :

ftp.funet.fi

```
/pub/OS/Linux/PEOPLE/Linus/net-source/base/NetKit-A-0.08.bin.tar.gz
/pub/OS/Linux/PEOPLE/Linus/net-source/base/NetKit-B-0.06.bin.tar.gz
```

S'il existe des versions plus récentes, utilisez-les. Consultez le fichier `README` en **premier** pour être sûr d'avoir les conditions nécessaires. Florian avait l'habitude d'avoir une distribution binaire des applications réseaux (fichier B) disponible, mais elle n'existe plus, aussi vous devez construire vos fichiers vous-même. Vous pouvez utiliser la procédure suivante:

```
# cd /usr/src
# gzip -dc NetKit-B-0.06.tar.gz | tar -xpvlf -
# cd NetKit-B-0.06
```

Puis, lisez le fichier `README`. Vous éditez le fichier `Makefile` et définissez `HAVE_SHADOW_PASSWORDS` de manière appropriée. Je n'utilise pas de mots de passe shadow, aussi j'ai placé un `#` au début de la ligne. Le reste n'a pas besoin d'être modifié, donc il reste à faire:

```
# make
# make install
```

NOTE IMPORTANTE : Florian a construit et préparé ces fichiers tar pour votre confort. Il a essayé de les rendre les plus complets possible et a inclus une distribution des binaires dans les versions `net-tools-n.n.nn`. Malheureusement Florian n'a pas choisi la même structure de répertoires qu'Alan avait choisie pour ses scripts d'installation des outils réseau. Cela signifie que vous devrez être particulièrement prudents. quand vous procéderez à l'installation. Florian va la modifier plus tard pour qu'il n'y ait plus de différences, mais en attendant, je vous suggère de faire ceci à la place de la commande précédente :

```
-  Desarchiver les binaires dans un endroit sur :
# cd /usr/src
# mkdir NetKit
# cd NetKit
# gzip -dc NetKit-A-0.07.bin.tar.gz | tar xpvlf -
# gzip -dc NetKit-B-0.06.bin.tar.gz | tar xpvlf -

-  Suppression des outils reseau decrits precedemment :
# rm ./bin/hostname ./sbin/route ./sbin/ifconfig ./sbin/netstat
# rm ./usr/sbin/arp ./usr/sbin/rarp ./usr/sbin/slattach

-  Copier les fichiers de Florian dans leur destination initiale :
# cp -vrpd . /
```

7.5 Pilotes ou paquetages additionnels

Si vous voulez installer des paquetages en cours de développement, ou du code en version Alpha/Beta, comme le support AX.25, vous aurez besoin des archives appropriées à ces logiciels. Consultez la suite du document pour plus de détails.

8 Configurer le noyau

Avant de pouvoir utiliser ou configurer les outils réseau, vous devez vous assurer que votre noyau contient les parties nécessaires. La meilleure façon de s'en assurer est de compiler le vôtre en sélectionnant les options que vous voulez.

En supposant que vous ayez obtenu et désarchivé un noyau puis appliqué les patches dont vous avez besoin pour supporter des options non-standards, tout ce que vous avez à faire est d'éditer `/usr/src/linux/drivers/net/CONFIG`. Ce fichier contient beaucoup de commentaires pour vous guider. En général vous n'aurez pas besoin de le modifier beaucoup ; il contient des valeurs par défaut quasi-universelles. Dans mon cas je n'ai pas besoin de l'éditer du tout. Ce fichier est nécessaire si votre carte est inhabituelle ou si elle n'est pas détectée par le pilote ethernet. Ceci permet de coder en dur certains éléments de matériel ethernet. Par exemple, si votre carte ethernet est proche, mais n'est pas un clone exact d'une WD-8013, vous pourrez configurer les adresses de la mémoire partagée pour être sûr que le pilote détecte bien la carte et qu'il sait la gérer correctement. Référez-vous à `(/http:/sunsite.unc.edu/mdw/HOWTO/Ethernet-HOWTO)` pour plus d'informations sur ce fichier et ses effets sur les cartes ethernet. Ce fichier contient aussi des paramètres pour PLIP, mais les valeurs par défaut devraient être bonnes sauf si vous avez une machine particulièrement lente.

Quand vous serez satisfait de votre fichier CONFIG, vous pourrez construire le noyau. La première étape est d'éditer le fichier Makefile au début pour être sûr que le noyau correspondra à votre configuration VGA ; puis vous pourrez lancer le programme de configuration du noyau :

```
# cd /usr/src/linux
# make config
```

On vous posera une série de questions. Il y a quatre sections traitant du code réseau. Ce sont les sections `General setup`, `Networking options`, `Network device support` et `Filesystems`. La plus difficile à configurer est la partie `Network device support`. Cette partie vous permet de choisir les pilotes des périphériques que vous voudrez configurer. Dans l'ensemble, vous pouvez choisir les options par défaut pour les autres parties sans risque. Voici un exemple du déroulement des opérations :

```
*
* General setup
*
 ...
 ...
Networking support (CONFIG_NET) [y] y
 ...
 ...
```

Dans la section `General setup` vous dites simplement si vous désirez le support réseau ou non. Naturellement vous répondez yes.

```
*
* Networking options
*
TCP/IP networking (CONFIG_INET) [y]
IP forwarding/gatewaying (CONFIG_IP_FORWARD) [n]
IP multicasting (CONFIG_IP_MULTICAST) [n]
IP firewalling (CONFIG_IP_FIREWALL) [n]
IP accounting (CONFIG_IP_ACCT) [n]
*
* (it is safe to leave these untouched)
*
PC/TCP compatibility mode (CONFIG_INET_PCTCP) [n]
Reverse ARP (CONFIG_INET_RARP) [n]
Assume subnets are local (CONFIG_INET_SNARL) [y]
Disable NAGLE algorithm (normally enabled) (CONFIG_TCP_NAGLE_OFF) [n]
The IPX protocol (CONFIG_IPX) [n]
*
```

La seconde partie de `Networking options` vous permet de sélectionner ou d'inhiber des options spéciales. Vous pouvez sans problème accepter les valeurs par défaut sauf si vous savez vraiment pourquoi vous voulez les changer. Elles sont décrites plus loin si vous êtes intéressés.

```
*
* Network device support
*
Network device support? (CONFIG_NETDEVICES) [y]
Dummy net driver support (CONFIG_DUMMY) [n]
SLIP (serial line) support (CONFIG_SLIP) [y]
 CSLIP compressed headers (CONFIG_SLIP_COMPRESSED) [y]
 16 channels instead of 4 (SL_SLIP_LOTS) [n]
PPP (point-to-point) support (CONFIG_PPP) [y]
PLIP (parallel port) support (CONFIG_PLIP) [n]
Do you want to be offered ALPHA test drivers (CONFIG_NET_ALPHA) [n]
Western Digital/SMC cards (CONFIG_NET_VENDOR_SMC) [y]
WD80*3 support (CONFIG_WD80x3) [y]
SMC Ultra support (CONFIG_ULTRA) [n]
AMD LANCE and PCnet (AT1500 and NE2100) support (CONFIG_LANCE) [n]
3COM cards (CONFIG_NET_VENDOR_3COM) [n]
Other ISA cards (CONFIG_NET_ISA) [n]
EISA, VLB, PCI and on board controllers (CONFIG_NET_EISA) [n]
Pocket and portable adaptors (CONFIG_NET_POCKET) [n]
*
```

Cette section est la plus importante. C'est l'endroit où vous sélectionnez les périphériques que vous voulez utiliser. Vous pouvez voir que j'ai choisi le support SLIP avec les en-têtes compressés, PPP, le pilote WD80*3 et rien d'autre. Répondez simplement `y` à tout ce qui vous intéresse et `n` au reste.

```
*
* Filesystems
*
 ...
 ...
/proc filesystem support (CONFIG_PROC_FS) [y]
NFS filesystem support (CONFIG_NFS_FS) [y]
 ...
 ...
```

Si vous voulez utiliser un client NFS, vous devrez choisir le système de fichier NFS. Vous devez aussi inclure le système de fichier `/proc` parce qu'un certain nombre d'utilitaires l'utilisent.

Une fois que vous aurez terminé, il ne vous reste plus qu'à compiler le noyau :

```
# make dep
# make
```

N'oubliez pas de faire un `make zlilo` pour que votre nouveau noyau fonctionne correctement.

8.1 Que font toutes ces drôles d'options réseau?

Les noyaux récents offrent un grand nombre d'options quand vous faites `make config`. En général vous n'avez pas besoin de les changer, mais ce peut être utile dans certains cas.

TCP/IP networking

Cette option est évidente, elle choisit si vous optez pour l'ensemble tcp/ip dans votre noyau. Il y a des chances pour que, si vous lisez ceci, vous répondiez oui.

Dummy networking device

Ceci a été ajouté pour permettre aux utilisateurs de SLIP et PPP de configurer une adresse sur leur machine linux indépendante de la liaison série lorsqu'elle est établie. C'est un moyen commode pour donner à votre machine deux adresses.

IP forwarding/gatewaying

Ceci détermine ce que fera le noyau quand il recevra un datagramme ayant une adresse de destination qui n'appartient pas à l'un de ses propres périphériques. Vous **devez** choisir cette option si vous voulez que votre noyau agisse en tant que routeur IP. La plupart des serveurs SLIP et PPP ont besoin de cette option.

IP multicasting

C'est un support au niveau test alpha pour le multilcasting IP, permettant d'utiliser des services tels que 'Internet Talk Radio' et la video en direct. Vous avez besoin de programmes additionnels pour utiliser cette fonctionnalité, ce n'est que le support noyau.

IP firewalling

Cette option permet de vous fournir des options de sécurité faciles à utiliser sur votre machine linux. Vous pouvez activer/désactiver la possibilité d'accès à vos ports tcp/ip dans des domaines d'adresses que vous aurez choisies. Ceci nécessite également des programmes additionnels pour fonctionner.

IP accounting

Cette option est pour ceux qui veulent utiliser leur machine pour fournir des connexions Internet aux autres à titre onéreux. Cela permet de compter et d'enregistrer par port et par adresse les octets qui entrent et qui sortent. Avec l'aide d'un logiciel adéquat ceci vous permet de répartir les frais d'utilisation pour chaque personne utilisant vos capacités réseau.

PC/TCP compatibility mode

Cette option fournit une solution pour un bogue qui engendre des problèmes quand on utilise des programmes PC/TCP pour dialoguer avec votre machine. Il y a un bogue dans PC/TCP qui provoque des difficultés et cette option permet d'éviter une catastrophe. Normalement vous la laissez désactivée, mais si vous avez des utilisateurs sur votre réseau qui tournent avec PC/TCP vous activez cette option pour éviter les problèmes.

Reverse ARP

Cette option vous permet de configurer le protocole RARP dans votre noyau. Cette option a été ajoutée pour permettre le lancement des systèmes SUN 3. Ce n'est cependant pas très utile.

Assume subnet are local

Cette option permet de choisir si vous considérez que tout votre sous-réseau est directement connecté à votre machine linux, ou bien si il peut être relié ou subdivisé sur une couche inférieure. En pratique, il y aura peu de différences si vous laissez l'option par défaut.

Disable NAGLE algorithm

C'est un option de temps qui détermine le moment où un datagramme doit être envoyé. Le réglage par défaut donne le meilleur rendement dans la plupart des cas et doit être laissé tel quel, sa désactivation risquant de dégrader le rendement. On peut changer cette option si on utilise un programme avec option socket , mais il vaut mieux la garder.

IPX protocol

Cette option permet d'inclure le support protocole IPX dans votre noyau. Le protocole IPX est un protocole inter-réseau similaire au protocole IP. Ce protocole est un de ceux utilisés dans l'ensemble Novell.

Amateur Radio AX.25 Level 2

Cette option permet de choisir si vous compilez avec l'ensemble radio-amateur AX25. Si vous le choisissez, alors une nouvelle classe de sockets réseau est disponible pour la programmation. Le protocole AX.25 est utilisé principalement par les radio-amateurs pour le packet-radio.

9 Configurer les pilotes réseau

Si tout c'est bien passé jusqu'ici, vous devez avoir un noyau Linux supportant les périphériques réseau que vous voulez utiliser et les outils réseau prêts à être configurés. *Maintenant place aux réjouissances !* Vous allez avoir besoin de configurer chaque pilote que vous voulez utiliser. Cette configuration consiste en général pour chaque pilote à indiquer son adresse IP et le réseau auquel il sera connecté.

Dans les versions précédentes de ce document, je vous avais présenté une version pratiquement complète des divers fichiers de configuration et j'avais inclus les commentaires nécessaires pour modifier ou détruire les lignes correspondant à vos besoins. À partir de cette version et pour les suivantes, je vais suivre une approche légèrement différente qui j'espère vous ménera à l'ensemble complet des fichiers de configuration en partant de zéro pour que vous puissiez comprendre exactement le pourquoi du comment. Je vais décrire chacun des fichiers et leurs fonctions au fur et à mesure.

9.1 Configuration des fichiers spéciaux dans /dev

Vous n'avez pas besoin de configurer de fichier spécial dans le répertoire `/dev`. Linux n'a pas besoin et n'utilise pas ces fichiers comme d'autres systèmes d'exploitation semblent en avoir besoin. Les pilotes sont fabriqués dynamiquement en mémoire dans le noyau et comme ils n'interagissent qu'avec d'autres routines du noyau, il n'y a pas de raison pour qu'ils soient visibles directement. Le noyau offre tous les points d'entrées dont vous aurez besoin pour les utiliser effectivement.

9.2 Quelles sont les informations nécessaires avant de commencer?

Avant de configurer le logiciel réseau, vous aurez besoin de quelques informations concernant votre connexion réseau. Votre fournisseur réseau ou votre administrateur réseau devrait être capable de vous fournir la plupart d'entre elles.

9.2.1 L'adresse IP

C'est l'unique adresse de machine, en notation décimale, que votre machine utilisera. Par exemple `128.253.153.54`. Elle vous sera fournie par votre administrateur réseau.

Si vous utilisez uniquement une connexion SLIP ou PLIP vous n'avez pas besoin de cette information. Vous pouvez aller directement à la partie traitant du pilote slip.

Si vous utilisez uniquement le pilote loopback, c'est-à-dire pas de périphérique ethernet, pas de connexion slip ou plip, alors vous n'avez pas besoin d'adresse ip car le pilote loopback utilise toujours l'adresse `127.0.0.1`.

9.2.2 Le masque réseau ('netmask')

Pour des raisons de performance, il est préférable de limiter le nombre d'hôtes sur tout segment particulier d'un réseau. C'est pourquoi les administrateurs réseau divisent habituellement leurs réseaux en plusieurs réseaux plus petits, appelés *subnets*. Chaque sous-réseau possède des adresses réseau qui lui sont assignées. Le *masque réseau* est un champ de bits qui, appliqué sur une adresse de votre réseau, permet de savoir à quel sous-réseau elle appartient. C'est très important pour le routage, et si vous trouvez par exemple que vous pouvez parler avec des gens en dehors de votre réseau mais pas avec certaines personnes de votre propre réseau, alors il est presque certain que vous avez donné un mauvais masque de sous-réseau.

Vos administrateurs réseau ont choisi le masque réseau quand le réseau a été créé et ils doivent donc être en mesure de vous procurer le masque à utiliser. La plupart des réseaux sont des sous sous-réseaux de classe C qui utilise le masque `255.255.255.0`. D'autres réseaux plus grands utilisent des masques de classe B (`255.255.0.0`). Le code NET-2/NET-3 choisit automatiquement un masque par défaut quand vous assignez une adresse à un pilote. La valeur par défaut suppose que votre réseau n'a **pas** été divisé en sous-réseaux.

Le logiciel NET-2/NET-3 choisira les masques suivants par défaut :

```
Pour des adresses avec comme premier octet_:
1-127         255.0.0.0         (Class A)
128-191       255.255.0.0       (Class B)
192+          255.255.255.0     (Class C)
```

Si l'un de ceux-ci ne fonctionne pas, essayez en un autre. Si ça ne fonctionne pas demandez de l'aide à votre administrateur réseau ou au gourou réseau local (on en trouve à la pelle).

Vous n'avez pas besoin de vous soucier du masque réseau pour l'interface loopback, ni si vous ne devez utiliser que SLIP/plip.

9.2.3 Adresse réseau

C'est votre adresse IP masquée (ET bit à bit) avec votre masque réseau. Par exemple :

```
Votre masque est :            255.255.255.0
et votre adresse IP est :     128.253.154.32    &&
                              ---------------
l'adresse de votre reseau :   128.253.154.0     =
```

9.2.4 Adresse de diffusion

'Un cri est un chuchotement que tout le monde entend qu'il le veuille ou non'

C'est normalement votre adresse réseau combinée par un OU logique avec le complément de votre masque réseau. Pour un réseau de classe C, avec un masque réseau `255.255.255.0`, votre *adresse de diffusion* sera votre adresse réseau (calculée précédemment), combiné par un OU logique avec `0.0.0.255`, le complément de votre masque réseau.

L'exemple doit ressembler à :

```
Si votre masque reseau est :  255.255.255.0      !
le complement est :             0.  0.  0.255    =
Si votre adresse reseau est : 128.253.154.0      ||
                              ----------------
Votre adresse de diffusion :  128.253.154.255    =
```

Pour des raisons historiques certains réseaux utilisent l'adresse du réseau comme adresse de diffusion. Si vous avez des doutes, contactez l'administrateur réseau.

Si vous avez accès à un *sniffer*, ou à un logiciel capable de visualiser le trafic réseau, vous devriez être capable de déterminer les adresses du réseau et de diffusion en regardant le trafic des autres sur le réseau local. Gardez un oeil ouvert sur (ou en filtrant tout sauf) les trames ethernet destinées à l'adresse de diffusion ethernet : `ff:ff:ff:ff:ff:ff`. Si l'une d'elles a comme source IP l'adresse de votre routeur local, et que l'identificateur de protocole n'est pas ARP, alors regardez l'adresse de destination IP car cette trame pourrait très bien être un RIP routant une diffusion de votre routeur et dans ce cas l'adresse IP de destination sera votre adresse de diffusion.

Encore une fois, si vous n'êtes pas sûr, vérifiez avec votre administrateur. Il préférera vous aider plutôt que de vous laisser connecter avec une machine mal configurée.

9.2.5 Adresse du routeur ('Gateway')

'Il doit bien y avoir un chemin hors d'ici.'

C'est l'adresse de la machine qui relie votre réseau au reste de l'internet. C'est votre 'porte' sur le monde extérieur. Quelques conventions existent pour choisir les adresses des routeurs que votre réseau doit suivre : l'adresse du routeur est la plus petite ou la plus grande du réseau. La pratique la plus courante est la première ; le routeur a une adresse pratiquement identique à la vôtre excepté un `.1` comme dernier octet. Autrement dit, si votre adresse est `128.253.154.32`, alors celle de votre routeur peut être `128.253.154.1`. Le routeur peut en fait être n'importe quelle adresse valide sur votre réseau et fonctionnera correctement, l'adresse n'y change rien. Il peut même y avoir plusieurs routeurs sur votre réseau. Vous devrez probablement demander à votre administrateur réseau de vous indiquer l'adresse de votre routeur.

Si vous utilisez seulement le loopback, vous n'avez pas besoin d'adresse de routeur. Si vous utilisez PPP, vous n'en avez pas besoin non plus. PPP déterminera automatiquement l'adresse correcte pour vous. Si vous utilisez SLIP, alors l'adresse du routeur sera votre adresse SLIP.

9.2.6 L'adresse du serveur de noms

La plupart des machines sur le réseau ont accès à un serveur de noms pour traduire les adresses lisibles par des humains en des adresses lisibles par des ordinateurs et vice-versa. Votre administrateur vous fournira l'adresse du serveur de noms le plus proche. Vous pouvez en fait démarrer un serveur de noms sur votre propre machine en lançant *named*, dans ce cas l'adresse du routeur sera `127.0.0.1`, l'adresse *loopback*. Cela dit, il n'est pas du tout nécessaire de lancer *named* (voir la section 'named' pour plus d'information).

Si vous utilisez seulement le loopback vous n'avez pas besoin de serveur de noms car vous n'allez communiquer qu'avec votre propre machine.

9.2.7 NOTE pour les utilisateurs de SLIP/PLIP/PPP

Vous pouvez avoir ou ne pas avoir besoin des informations précédentes. Cela dépend de la manière dont votre connexion est établie et des capacités de la machine à laquelle vous êtes reliés. Vous trouverez plus de détails dans la section qui décrit SLIP/PLIP et PPP.

9.3 /etc/rc.d/rc.inet1,2 ou /etc/rc.net ou /etc/init.d/network

Les commandes de configuration du réseau peuvent être tapées manuellement mais vous voudrez probablement que l'interface réseau se configure automatiquement à chaque fois que vous démarrez votre machine.

Les fichiers `'rc'` ont été créés spécialement dans ce but. Pour les non-spécialistes d'UNIX : les fichiers `'rc'` sont lancés au démarrage par le programme *init* et démarrent tout le système de base comme *syslog*, *update* et *cron*. Ils sont analogues au fichier MS-DOS *autoexec.bat*. *rc* signifie *'runtime commands'*. Par convention ces fichiers se trouvent dans le répertoire */etc*. Le standard du système de fichier Linux ne décrit pas où vous devez mettre vos fichiers *rc*. Il laisse le choix de suivre les conventions BSD (/etc/rc.*) ou System-V (/etc/rc.d/rc*). Alan, Fred et moi, utilisons tous la convention System-V, c'est donc ce qui va être décrit dans la suite. Cela signifie que les fichiers se situent dans */etc/rc.d* et se nomment *rc.inet1* et *rc.inet2*. Le premier fichier *rc* appelé au démarrage est le fichier `/etc/rc` qui à son tour appelle les autres comme `rc.inet1`, lui même appelant `rc.inet2`. La localisation de ces fichiers n'a aucune importance tant que *init* peut les trouver.

Dans certaines distributions le fichier *rc* pour le réseau est appelé *rc.net* et se trouve dans le sous-répertoire */etc*. Le fichier *rc.net* est simplement les fichiers *rc.inet1* et *rc.inet2* combinés en un seul fichier. Quelques équivalences:

```
Conventionnellement          Slackware            Debian et autres nouveaux
===================         ================     =========================
/etc/rc.d/rc.inet1          /etc/rc.net           /etc/init.d/network

/etc/rc.d/rc.inet2          /etc/rc.net           /etc/init.d/netbase
                                                  /etc/init.d/netstd_init
                                                  /etc/init.d/netstd_nfs
                                                  /etc/init.d/netstd_misc
```

L'endroit où les commandes sont déclenchées n'a aucune importance tant que les interfaces sont configurées avant de démarrer les démons et applications réseau.

Dans la suite j'utiliserai les fichiers *rc.inet1* et *rc.inet2*. Je les conserverai dans le répertoire `/etc/rc.d`. Donc si vous utilisez une distribution qui utilise *rc.net* ou si vous voulez les placer ailleurs vous devrez faire les ajustements nécessaires.

Nous allons construire ensemble ces fichiers.

9.3.1 rc.inet1

Le fichier *rc.inet1* configure les interfaces TCP/IP de base en utilisant deux programmes : */sbin/ifconfig* et */sbin/route*.

ifconfig

/sbin/ifconfig met en place les paramètres de votre interface comme votre adresse IP, votre masque réseau, votre adresse de diffusion... Vous pouvez utiliser *ifconfig* sans paramètre pour visualiser la configuration de toutes vos interfaces réseau. Consultez la page du manuel d'*ifconfig* pour plus de détails.

route

/sbin/route est utilisé pour créer, modifier et détruire des entrées dans une table (la table de routage) que le code réseau consulte quand il a un paquet à transmettre. La table de routage liste l'adresse de destination et l'interface par laquelle cette adresse peut être atteinte. Vous pouvez utiliser la commande *route* sans paramètre pour visualiser le contenu de la table de routage. Pour plus de détails consultez les pages du manuel sur *route*.

9.3.2 rc.inet2

Le fichier *rc.inet2* lance tous les démons réseau comme *inetd*, *portmapper* et les autres. Ceci sera traité dans la section 'rc.inet2', pour le moment nous allons nous concentrer sur *rc.inet1*. Je mentionne ce fichier ici pour que vous compreniez que si vous utilisez une autre configuration, par exemple un fichier *rc.net* unique, sa seconde partie correspond à *rc.inet2*. Il est important de se rappeler que vous devez démarrer les applications et les démons réseau **après** avoir configuré vos interfaces réseau.

9.4 Configurer l'interface Loopback (obligatoire)

L'interface loopback ne correspond pas à une interface physique. C'est un artifice logiciel qui se comporte comme une interface physique. Sa fonction est de vous permettre de vous connecter à vous même et ainsi de tester le logiciel réseau sans être relié à un réseau physique. C'est très utile si par exemple vous développez un logiciel réseau et vous n'avez qu'une connexion SLIP. Vous pouvez écrire et tester votre logiciel localement et quand vous êtes prêt à le tester sur vrai réseau, vous établissez votre liaison SLIP et vous le testez. En procédant ainsi vous ne gênez pas les autres utilisateurs si votre programme fonctionne de travers.

Par convention, l'interface loopback correspond toujours à l'adresse IP `127.0.0.1`, vous utiliserez donc cette adresse lors de la configuration.

Sous Linux l'interface loopback est appelé *'lo'*. Nous allons maintenant créer notre première entrée dans le fichier *rc.inet1* :

```
#!/bin/sh
#
# rc.inet1   --  configure les interfaces reseau.
#
# Attachement de l'interface loopback.
/sbin/ifconfig lo 127.0.0.1
#
# Ajout d'une route vers l'interface loopback.
/sbin/route add 127.0.0.1
# Fin loopback
#
```

Vous avez utilisé le programme *ifconfig* pour donner son adresse IP à l'interface loopback et le programme *route* pour créer une entrée dans la table de routage pour être sûr que toutes les données destinées à l'adresse `127.0.0.1` seront bien envoyées à l'interface loopback.

Il y a deux points importants à noter ici.

Premièrement, le masque réseau et l'adresse de diffusion ont pris des valeurs par défaut pour l'interface loopback. Pour connaître leurs valeurs, lançons la commande *ifconfig* sans argument.

```
# ifconfig
lo        Link encap Local Loopback
          inet addr 127.0.0.1  Bcast 127.255.255.255  Mask 255.0.0.0
          UP BROADCAST LOOPBACK RUNNING  MTU 2000  Metric 1
          RX packets 0 errors 0 dropped 0 overrun 0
          TX packets 30 errors 0 dropped 0 overrun 0
#
```

Deuxièmement, il n'est pas évident de comprendre comment la commande *route* choisit l'interface loopback comme destination pour l'adresse `127.0.0.1`. Le programme *route* est assez intelligent pour savoir que l'adresse `127.0.0.1` appartient au réseau géré par l'interface loopback. Il arrive à trouver cela en consultant l'adresse IP et le masque réseau. Vous pouvez visualiser le contenu de la table de routage avec la commande *route* sans argument :

```
# route
Kernel routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
127.0.0.0       *               255.0.0.0       U     0      0       30 lo
#
```

Note : vous pouvez utiliser l'argument `-n` si votre résolution de noms n'est pas configurée correctement. L'argument `-n` indique à route de n'afficher que les adresses sous forme **n**umérique et de ne pas s'occuper de rechercher le nom correspondant.

9.5 Configurer une interface ethernet (optionnel)

Vous ne serez intéressé par cette section que si vous voulez configurer une carte ethernet. Sinon passez à la section suivante.

La configuration d'une interface ethernet est légèrement plus compliquée que celle de l'interface loopback. Cette fois vous devrez probablement spécifier explicitement le masque réseau et l'adresse de diffusion sauf si les valeurs par défaut vous conviennent et elles vous conviendront probablement.

Pour cela vous aurez besoin de l'adresse IP qui vous a été assignée, le masque réseau utilisé sur votre réseau et de l'adresse de diffusion en vigueur sur votre réseau.

La première interface ethernet sous *Linux* est appelée *'eth0'*, la deuxième *'eth1'* et ainsi de suite. Vous allez maintenant **ajouter** une section à votre fichier *rc.inet1*. Le fragment de code suivant devrait faire l'affaire si vous remplacez les adresses spécifiées par les vôtres :

```
#
# Attachement d'une interface ethernet
```

```
#
#  configure l'adresse IP, le masque reseau et
#  l'adresse de diffusion.
/sbin/ifconfig eth0 IPA.IPA.IPA.IPA
/sbin/ifconfig eth0 netmask NMK.NMK.NMK.NMK
/sbin/ifconfig eth0 broadcast BCA.BCA.BCA.BCA
#
# ajout d'une route reseau pour pointer dessus
/sbin/route add -net NWA.NWA.NWA.NWA device eth0
#
# Fin ethernet
#
```

Où :

IPA.IPA.IPA.IPA

représente votre adresse IP.

NMK.NMK.NMK.NMK

représente votre masque réseau.

BCA.BCA.BCA.BCA

représente votre adresse de diffusion.

NWA.NWA.NWA.NWA

représente l'adresse de votre réseau.

Notez l'utilisation de l'argument `-net` à la commande *route*. Ceci signifie à *route* d'ajouter une route vers un *réseau* et non vers un *hôte*. Il existe autre méthode pour atteindre les mêmes objectifs : vous pouvez supprimer la ligne avec `-net` si vous avez l'adresse de votre réseau dans le fichier */etc/networks*. Cette méthode sera traitée plus tard dans la section '/etc/networks'.

9.6 Configurer une interface SLIP (optionnel)

SLIP (Serial Line Internet Protocol) vous permet d'utiliser TCP/IP sur une ligne série, que ce soit une ligne téléphonique avec un modem numéroteur ou une ligne spécialisée. Bien sûr pour utiliser SLIP, vous devez avoir accès à un *serveur SLIP*. Beaucoup d'universités et de sociétés proposent des accès SLIP à travers le monde.

SLIP utilise le port série de votre machine pour transporter des données IP. Pour réaliser ceci, il doit prendre le contrôle du port série. Les interfaces SLIP s'appellent *sl0*, *sl1* etc. Comment ces noms sont associés à vos ports série ? Il y a deux programmes prévus pour réaliser ces liens, ils s'appellent *dip* et *slattach*.

9.6.1 dip

dip (Dialup IP) est un programme intelligent qui est capable de configurer la vitesse de votre port, de commander à votre modem de numéroter le numéro de votre correspondant, de se connecter automatiquement au serveur distant et d'extraire des informations comme votre adresse IP et réaliser les *ioctl* nécessaires pour passer votre port série en mode SLIP. *dip* possède un langage de commandes très puissant que vous pouvez exploiter pour réaliser des procédures de connexions automatiques.

dip était proposé en standard dans l'archive `net-tools`, mais depuis que le développement de *dip* est maintenant autonome vous devez récupérer les sources séparément. Il y a de nombreuses autres versions de *dip* qui offrent un ensemble de nouvelles fonctionnalités. Depuis que `dip-uri` est si populaire, les exemples décrits dans ce document sont basés sur les versions habituelles. Vous pouvez les trouver sur:

sunsite.unc.edu

```
/pub/Linux/system/Network/serial/dip337j-uri.tgz
```

Pour l'installer, faites:

```
#
# cd /usr/src
# gzip -dc dip337j-uri.tgz | tar xvf -
# cd dip.3.3.7j

<editez Makefile>

# make install
#
```

Le fichier `Makefile` suppose l'existence d'un groupe nommé *uucp*, mais vous pourrez changer cela en *dip* ou *SLIP* suivant votre configuration.

9.6.2 slattach

Par contre *slattach* est un programme très simple, très facile à utiliser, mais qui n'a pas la sophistication de *dip*. Il n'a pas de possiblité de faire des scripts, tout ce qu'il fait est de configurer votre interface série en tant que dispositif SLIP. Il suppose que vous avez toutes les informations nécessaires et que votre liaison série est établie avant de lui faire appel. *slattach* est idéal pour ceux qui possède un lien permanent avec leur serveur, comme un câble ou une ligne spécialisée.

9.6.3 Quand et lequel utiliser ?

Vous devez utiliser dip quand votre lien avec la machine qui vous sert de serveur est un modem ou tout autre lien temporaire. Vous devez utiliser *slattach* quand vous avez un lien permanent, une ligne spécialisée, voire un simple câble, entre votre machine et le serveur, et qu'il n'y a aucune action spéciale à réaliser pour que le lien fonctionne. Pour plus d'information consulter la section 'Connexion SLIP permanente'.

La configuration de SLIP est à peu près identique à celle d'une interface ethernet (commencez par lire la section 'Configuration d'une interface ethernet'). Il existe quand même des différences importantes.

Premièrement, les liens SLIP sont différents des réseaux ethernet dans le sens où il n'y a jamais que deux hôtes sur le réseau, un à chaque bout. Deuxièmement, l'interface ethernet est disponible immédiatement dès que vous êtes relié alors qu'avec SLIP, cela dépend du type de lien que vous avez, vous pouvez avoir besoin d'initialiser votre connexion d'une manière spéciale.

Si vous utilisez *dip*, vous ne le ferez pas normalement au démarrage de votre machine mais plus tard, quand vous en aurez besoin. Il est possible d'automatiser cette procédure. Si vous utilisez *slattach* vous voudrez probablement ajouter la section adéquate au fichier *rc.inet1*. Ceci sera décrit bientôt.

Il y a deux catégories de serveurs SLIP : les serveurs à adresses IP dynamiques et les serveurs à adresses IP statiques. Pratiquement tous les serveurs SLIP vous demanderont un nom d'utilisateur et un mot de passe à la connexion. *dip* peut gérer cette connexion automatiquement.

9.6.4 Serveur SLIP statique et DIP

Avec un serveur SLIP statique, vous avez une adresse IP qui vous est personnelle. A chaque fois que vous vous connecterez au serveur vous devrez configurer votre interface SLIP avec cette adresse. Le serveur SLIP statique répondra à votre appel, peut-être en vous demandant un nom d'utilisateur et un mot de passe, puis il aiguillera tous les paquets de données qui vous sont destinés vers votre liaison. Si vous avez un serveur SLIP statique, vous pouvez mettre votre nom de machine et votre adresse IP dans le fichier `/etc/hosts`. Vous pouvez aussi configurer d'autres fichiers comme : `rc.inet2`, `host.conf`, `resolv.conf`, `/etc/HOSTNAME`, et `rc.local`. Vous n'avez pas besoin d'ajouter des commandes spéciales à votre fichier `rc.inet1` car *dip* fait tout le sale boulot pour vous. Vous devrez donner à *dip* les informations nécessaires et il vous paramètrera l'interface après avoir piloté votre modem et s'être connecté à votre serveur SLIP.

Si c'est la manière dont fonctionne votre serveur SLIP vous pouvez aller directement à la section 'Utiliser Dip'.

9.6.5 Serveur SLIP dynamique et DIP

Un serveur SLIP *dynamique* vous alloue une adresse IP prise au hasard dans un ensemble à chaque fois que vous vous connectez. Vous n'avez donc aucune garantie pour avoir une adresse IP identique à chaque fois et cette adresse peut très bien être utilisée par quelqu'un d'autre dès que vous vous déconnectez. L'administrateur réseau qui a configuré le serveur SLIP aura choisi un ensemble d'adresses et quand le serveur recevra un nouvel appel, il choisira une adresse inutilisée, conduira le processus de connexion, imprimera un message de bienvenue qui contiendra l'adresse IP allouée et utilisera ensuite cette adresse pendant toute la durée de l'appel.

La configuration pour ce type de serveur est identique à celle d'un serveur statique excepté que vous devez ajouter une étape pour récupérer l'adresse IP que le serveur vous a allouée et configurer l'interface avec.

Là aussi, *dip* fait le sale boulot, et les nouvelles versions sont suffisamment intelligentes pour non seulement vous connecter mais aussi pour récupérer automatiquement l'adresse IP dans le message de bienvenue et l'enregistrer pour que vous puissiez configurer votre interface avec elle. Si c'est la manière dont votre serveur

SLIP travaille, alors vous pouvez aller à la section 'Utiliser dip' pour apprendre comment configurer *dip* convenablement.

9.6.6 Utiliser DIP

Comme expliqué précédemment, *dip* est un programme puissant qui peut simplifier et automatiser votre procédure de connexion au serveur SLIP, en vous connectant, en démarrant la connexion, et en configurant l'interface SLIP avec les commandes *ifconfig* et *route* appropriées.

Vous devrez essentiellement écrire un 'script de dialogue', qui consiste en une liste de commandes *dip* expliquant comment exécuter toutes les actions dont vous avez besoin. Pour un exemple vous pouvez consulter le fichier `sample.dip` qui est fourni avec *dip*. *dip* est un programme puissant avec beaucoup d'options. Vous pouvez consulter la page de *man*, le fichier README et les fichiers d'exemples fournis avec *dip*.

Le script `sample.dip` suppose que vous utilisez un serveur statique, donc que vous connaissez votre adresse IP à l'avance. Pour les serveurs SLIP dynamiques, les versions récentes de *dip* possèdent une commande que vous pouvez utiliser pour configurer votre interface SLIP avec l'adresse IP que le serveur vous a allouée. L'exemple suivant est une version modifiée de `sample.dip` fournie avec *dip337j-uri.tgz* et est probablement un bon point de départ. Vous pouvez le sauvegarder dans le fichier `/etc/dipscript` et éditez le pour l'adapter à votre configuration:

```
#
# sample.dip    Programme support de connexion IP.
#
#               Ce fichier montre (devrait montrer) comment utiliser DIP
#       Ce fichier doit fonctionner avec des serveurs dynamiques Annex,si vous
#       utilisez un serveur a adresse statique, alors utilisez le
#       fichier sample.dip qui fait partie du paquetage dip337-uri.tgz.
#
#
# Version:      @(#)sample.dip  1.40    07/20/93
#
# Auteur:       Fred N. van Kempen, <waltje@uWalt.NL.Mugnet.ORG>
#

main:
# Ci-dessous, mettre l'adresse et le nom de l'autre machine.
# Ma machine appelee s'appelle 'xs4all.hacktic.nl' (== 193.78.33.42)
get $remote xs4all.hacktic.nl
# Regler le masque de reseau sur sl0 a 255.255.255.0
netmask 255.255.255.0
# Reglez le port serie desire et sa vitesse.
port cua02
speed 38400

# Faire un reset sur le modem et le terminal.
# Ceci semble creer des ennuis a certains!
reset
```

```
# Note! Valeurs "standards" predefinies de "errlevel":
#  0 - OK
#  1 - CONNECT
#  2 - ERROR
#
# Vous pouvez les changer en faisant 'grep "addchat()" *.c'

# Preparation de la numerotation.
send ATQ0V1E1X4\r
wait OK 2
if $errlvl != 0 goto modem_trouble
dial 555-1234567
if $errlvl != 1 goto modem_trouble

# Nous sommes connectes. Debut de session sur le systeme.
login:
sleep 2
wait ogin: 20
if $errlvl != 0 goto login_trouble
send MYLOGIN\n
wait ord: 20
if $errlvl != 0 goto password_error
send MYPASSWD\n
loggedin:

# Nous sommes maintenant enregistres.
wait SOMEPROMPT 30
if $errlvl != 0 goto prompt_error

# Commander le serveur en mode SLIP
send SLIP\n
wait SLIP 30
if $errlvl != 0 goto prompt_error

# Obtenir et regler votre adresse IP a partir du serveur.
#   Ici nous supposons qu'apre avoir command\'{e} le serveur SLIP
#   \`{a} se mettre en mode SLIP, qu'il donne votre adresse IP
get $locip remote 30
if $errlvl != 0 goto prompt_error

# Reglage des parametres.
get $mtu 296
# S'assurer que "route add -net default xs4all.hacktic.nl" sera fait
default

# Dire bonjour et a l'assaut!
done:
```

```
print CONNECTED $locip ---> $rmtip
mode CSLIP
goto exit

prompt_error:
print TIME-OUT waiting for sliplogin to fire up...
goto error

login_trouble:
print Trouble waiting for the Login: prompt...
goto error

password:error:
print Trouble waiting for the Password: prompt...
goto error

modem_trouble:
print Trouble occurred with the modem...
error:
print CONNECT FAILED to $remote
quit

exit:
exit
```

L'exemple précédent suppose que vous appelez un serveur SLIP *dynamique*. Si vous appelez un serveur *statique*, utilisez le fichier `sample.dip` fourni avec *dip337j-uri.tgz*.

La commande *dip get $local* cherche dans le texte envoyé par le serveur une chaîne de caractères qui ressemble à une adresse IP, en fait des nombres séparés par un point. Cette modification a été créée spécialement pour gérer automatiquement les adresses IP allouées par les serveur SLIP dynamiques.

L'exemple précédent assigne l'interface SLIP comme étant la route par défaut. Si ce n'est pas ce que vous voulez, vous pourriez avoir une connexion ethernet qui sera votre route par défaut, supprimez la commande *default* du script.

Quand le script est terminé, si vous faites la commande *ifconfig*, vous verrez que vous avez une interface *sl0*. C'est votre interface SLIP. Si vous avez besoin, vous pouvez la modifier manuellement après la fin de *dip* en utilisant les commandes *ifconfig* et *route*.

dip permet de choisir parmi plusieurs protocoles avec la commande `mode`, l'exemple le plus courant est *cslip* (SLIP avec compression). Pour l'utiliser, les deux extrémités du lien doivent être d'accord a priori, vous devez donc vous assurer que vous avez le même protocole que votre serveur.

L'exemple précédent est assez robuste et devrait se débrouiller avec la plupart des erreurs. Consultez la page de man de *dip* pour plus d'informations. Vous pouvez aussi par exemple écrire des scripts qui rappellent s'ils n'ont pas obtenu la connexion ou bien qui tentent de se connecter à d'autres serveurs si vous avez accès à plusieurs.

9.6.7 Connexion SLIP permanente sur une liaison spécialisée et slattach

Si vous avez un câble entre deux machines, ou si vous êtes assez fortuné pour avoir une ligne dédiée, ou toute autre connexion série permanente entre votre machine et une autre, vous n'avez pas besoin d'utiliser *dip* pour initialiser votre liaison série. *slattach* est un utilitaire très simple d'emploi qui possède juste les fonctionnalités nécessaires pour configurer votre connexion.

Comme votre liaison doit être permanente, vous pouvez ajouter les commandes pour la créer dans le fichier `rc.inet1`. En résumé tout ce que vous avez à faire est de configurer votre interface série avec la bonne vitesse et de la basculer dans le mode SLIP. *slattach* vous permet de le faire en une commande. **Ajoutez** la commande suivante à votre fichier `rc.inet1` :

```
#
# Attache une connexion SLIP statique avec ligne specialisee
#
#  configuration de /dev/cua0 pour une vitesse de 19.2kbps et cslip
/sbin/slattach -p cslip -s 19200 /dev/cua0 &
/sbin/ifconfig sl0 IPA.IPA.IPA.IPA pointopoint IPR.IPR.IPR.IPR up
#
# Fin SLIP statique.
```

où :

IPA.IPA.IPA.IPA
représente votre adresse IP.

IPR.IPR.IPR.IPR
représente l'adresse IP de la machine distante.

slattach associe la première interface SLIP libre à la liaison série spécifiée. *slattach* débute avec *sl0*. Donc la première commande *slattach* associe l'interface SLIP *sl0* à la liaison série spécifiée, puis *sl1* la fois suivante, etc.

slattach vous permet de configurer un protocole spécifique avec l'option `-p`. Dans votre cas vous utiliserez *slip* ou *cslip* suivant que vous utilisez la compression ou non. Les deux extrémités doivent être d'accord sur le fait d'utiliser la compression ou non.

9.7 Configurer une interface PLIP (optionnel)

plip (Parallel Line IP) est comme SLIP, dans le sens où il vous permet d'établir une connexion réseau *point à point* entre deux machines, sauf qu'il a été conçu pour utiliser le port parallèle de votre machine au lieu du port série. Comme il est possible de transférer plus d'un bit à la fois sur le port parallèle, on peut obtenir des vitesses plus élevées avec l'interface *plip* qu'avec une interface série standard. De plus le plus simple des ports parallèles, le port d'imprimante, peut être utilisé sans avoir besoin d'acheter de coûteux UART 16550AFN pour vos ports série.

Veuillez noter que certains portables utilisent des puces qui ne fonctionnent pas avec PLIP parce qu'elles ne permettent pas certaines combinaisons de signaux nécessaires à PLIP, et que les imprimantes n'utilisent pas.

L'interface *plip* de Linux est compatible avec le *Crywyr Packet Driver PLIP*, ce qui signifie que vous pouvez relier votre machine Linux à une machine DOS via *plip*, avec n'importe quel sorte de logiciel TCP/IP. Vous avez deux manières d'utiliser le pilote PLIP. Vous pouvez soit compiler le pilote dans votre noyau, soit utiliser le progiciel modules pour charger le module PLIP de manière dynamique. Je recommande de le compiler dans votre noyau: c'est le plus simple et vous avez à tout instant le pilote sous la main. Au moment de compiler le noyau, vous devez vérifier un seul fichier pour configurer *plip*. Ce fichier c'est `/usr/src/linux/driver/net/CONFIG`, il contient les timings pour *plip* en millisecondes. Les valeurs par défaut devraient être correctes dans la plupart des cas. Vous aurez peut-être besoin de les augmenter si vous avez un ordinateur particulièrement lent, et dans ce cas les temps à augmenter sont sur l'**autre** ordinateur. Le pilote suppose les valeurs suivantes par défaut:

```
dispositif   adresse e/s     IRQ
----------   -----------     -------
plip0        0x3BC           5
plip1        0x378           7
plip2        0x278           2 (9)
```

Si votre port parallèlle ne s'accorde pas avec l'une de ces combinaisons vous pouvez changer l'IRQ d'un port en utilisant la commande *ifconfig* avec 'irq' comme paramètre. Vérifiez bien que votre ROM BIOS supporte l'option de pouvoir valider des IRQ sur vos ports paralèlles.

Pour configurer une interface *plip*, vous devez **ajouter** ces lignes au fichier `rc.inet1`:

```
#
# Attache une interface PLIP
#
#  configure le premier port parallele comme interface PLIP
/sbin/ifconfig plip0 IPA.IPA.IPA.IPA pointopoint IPR.IPR.IPR.IPR up
#
# Fin plip
```

où :

IPA.IPA.IPA.IPA

représente votre adresse IP.

IPR.IPR.IPR.IPR

représente l'adresse IP de la machine distante.

Le paramètre *pointopoint* a la même signification que pour SLIP, dans le sens qu'il spécifie l'adresse de la machine à l'autre bout de la liaison.

Dans presque tous les cas vous pouvez traiter l'interface *plip* comme l'interface *slip*, sauf que ni *dip* ni *slattach* ne devront ni ne pourront être utilisés.

9.7.1 Diagramme de câblage PLIP

plip a été conçu pour être utilisé avec les mêmes câbles que ceux utilisés habituellement sous MS-DOS pour les transferts de PC à PC.

Le diagramme de cablage (extrait de `/usr/src/linux/drivers/net/plip.c`) ressemble à :

```
Nom broche  Relier broche - broche
----------  ------------------------------
GROUND      25 - 25
D0->ERROR   2 - 15
ERROR->D0   15 - 2
D1->SLCT    3 - 13
SLCT->D1    13 - 3
D2->PAPOUT  4 - 12
PAPOUT->D2  12 - 4
D3->ACK     5 - 10
ACK->D3     10 - 5
D4->BUSY    6 - 11
BUSY->D4    11 - 6
D5          7*
D6          8*
D7          9*
STROBE      1*
FEED        14*
INIT        16*
SLCTIN      17*
```

Notes : Ne pas relier les broches marquées avec une '*'. Les masses supplémentaires sont les broches 18, 19, 20, 21, 22, 23 et 24.

Si le câble que vous utilisez possède un blindage métallique, il doit être relié à la partie métallique de la prise DB-25 à **une extrémité seulement**.

Attention: un câble mal branché peut détruire votre carte contrôleur. Soyez très soigneux et vérifiez chaque liaison pour vous éviter tout travail inutile ou des gros ennuis. Bien que l'on puisse utilser PLIP avec des câbles sur des longues distances, évitez le si possible. Les spécifications des câbles permettent d'avoir des longueurs d'un mètre à peu près. Faites attention quand vous utilisez de grandeurs longueurs car des sources de champs magnétiques intenses comme les éclairs, les lignes de puissance, les émetteurs radio peuvent donner des interférences et endommager votre carte contrôleur. Si vous voulez vraiment connecter deux ordinateurs sur une grande distance, il vaut mieux regarder plutôt vers une paire de cartes ethernet et un câble coaxial.

10 Routage (obligatoire)

Après avoir configuré toutes vos interfaces réseau vous vous demandez comment votre machine va aiguiller vos paquets IP. Si vous avez une seule interface réseau, votre choix est simple, tous les paquets pour toutes les machines excepté la vôtre doivent être dirigés vers cette interface. Si vous avez plus d'une interface, le choix sera plus compliqué. Vous pouvez par exemple avoir à la fois une interface ethernet et une liaison SLIP avec votre machine à la maison. Dans ce cas vous devez diriger les paquets pour votre machine vers l'interface SLIP et tous les autres vers l'interface ethernet. Le routage est vraiment un mécanisme très simple, mais ne vous inquiétez pas si vous le trouvez un peu difficile au premier abord; tout le monde passe par là.

Vous pouvez visualiser le contenu de votre table de routage en utilisant la commande *route* sans option.

Il y a quatre mécanismes utilisés fréquemment pour configurer le réseau sous unix. Je vais les décrire un par un rapidement.

10.1 Routes Statiques/Manuelles

Le routage statique, comme son nom l'indique, est un routage 'codé en dur'. Autrement dit, il ne changera pas si il y a des problèmes sur votre réseau ou si une autre route devient utilisable. Les routes statiques sont souvent utilisées quand vous avez un réseau très simple sans autres routes possibles vers la machine de destination, ou si vous voulez utiliser une route particulière quels que soient les changements du réseau.

Sous Linux, il y a une utilisation spéciale des routes manuelles qui sert à ajouter une route SLIP ou PLIP vers une machine que vous avez configurée : le paramètre *ifconfig pointopoint*. Donc si vous avez un lien slip/plip, et avez utilisé le paramètre *pointopoint* en spécifiant l'adresse de la machine distante, vous devez ajouter une route statique vers cette adresse pour que le logiciel de routage ip sache comment aiguiller les paquets vers cette adresse. La commande *route* que vous utiliseriez pour un lien SLIP/PLIP avec une ligne spécialisée serait :

```
#/sbin/route add IPR.IPR.IPR.IPR
```

où : `IPR.IPR.IPR.IPR` représente l'adresse IP de la machine distante.

10.2 Route par défaut

Le mécanisme de la *route par défaut* est probablement le mécanisme le plus commun et le plus utilisé par les utilisateurs de stations de travail et les machines sur la plupart des réseaux. La route *par défaut* est une route statique spéciale qui correspond à toutes les adresses de destination ; ainsi, quand un paquet ne trouve pas de destination, il est envoyé à la route par *défaut*.

Si votre configuration est uniquement composée d'une interface ethernet ou d'une seule interface SLIP, vous devrez diriger celle-ci vers votre route par défaut. Dans le cas d'une interface ethernet, le noyau Linux sait où envoyer les paquets pour n'importe quelle machine de votre réseau. Il y arrive grâce à l'adresse réseau et masque réseau décrits précédemment. Cela signifie que seuls les paquets ayant une destination extérieure à votre réseau posent problème. Pour que cela fonctionne vous devez faire pointer votre route par défaut sur l'adresse de votre *routeur* de façon à faire sortir les paquets de votre réseau local. Si vous possédez une connexion SLIP, votre *serveur SLIP* fera office de *routeur*, donc votre route par défaut sera votre *serveur SLIP*.

Pour configurer votre route par défaut, **ajoutez** les lignes suivantes à votre fichier `rc.inet1` **après** la configuration de toutes vos interfaces réseau :

```
#
# Ajouter une route par defaut.
#
/sbin/route add default gw RGA.RGA.RGA.RGA
#
```

où : `RGA.RGA.RGA.RGA` représente votre l'adresse de votre routeur.

10.3 Procuration ARP

Cette méthode est laide et risquée. Elle doit être employée avec d'extrêmes précautions, mais certains d'entre vous voudront l'utiliser quand même.

Ceux qui auront le plus besoin du mécanisme de *procuration ARP* seront ceux qui veulent utiliser leurs machines comme serveur SLIP. Par contre pour ceux qui veulent utiliser PPP, le démon PPP vous simplifiera et vous automatisera la tâche, ce qui la rendra beaucoup plus sûre d'emploi.

Normalement quand une machine de votre réseau ethernet veut parler avec vous, elle connaît votre adresse IP, mais ne connaît pas votre adresse physique (ethernet) vers laquelle envoyer les paquets. Le mécanisme ARP a été conçu spécialement pour mettre en correspondance les adresses réseau et les adresses physiques.

Si vous voulez utiliser votre machine comme serveur pour d'autres machines, elle devra savoir répondre aux requêtes ARP à leur place car elles ne seront pas reliées physiquement au réseau ethernet. Disons que vous avez choisi un ensemble d'adresses IP sur votre réseau local qui serviront aux clients SLIP. Par exemple ces adresses seraient : `128.253.154.120-124`, et votre carte ethernet a comme adresse physique `00:00:C0:AD:37:1C`. (Vous trouverez l'adresse physique en exécutant la commande *ifconfig* sans paramètre). Pour indiquer à votre serveur Linux de répondre aux requêtes ARP par procuration pour les adresses désirées **ajouter** les commandes suivantes à votre fichier `rc.inet1` :

```
#
# Procuration ARP pour les adresses SLIP allouees.
#
/sbin/arp -s 128.263.154.120 00:00:C0:AD:37:1C pub
/sbin/arp -s 128.263.154.121 00:00:C0:AD:37:1C pub
/sbin/arp -s 128.263.154.122 00:00:C0:AD:37:1C pub
/sbin/arp -s 128.263.154.123 00:00:C0:AD:37:1C pub
/sbin/arp -s 128.263.154.124 00:00:C0:AD:37:1C pub
#
# Fin des procurations ARP.
```

L'argument `pub` signifie *'publier'*. C'est cet argument qui indique à votre machine de répondre aux requêtes pour ces adresses bien qu'elles ne soient pas destinées à votre machine. Quand elle répondra, elle renverra bien sûr sa propre adresse physique.

Naturellement vous devez vous assurer que votre serveur Linux possède bien ces adresses IP reliées à l'interface SLIP.

Si vous utilisez PPP, vous n'aurez pas besoin de vous tracasser avec la table ARP, *pppd* se chargera de gérer ces entrées pour vous si vous utilisez le paramètre *proxyarp*, et aussi longtemps que l'adresse IP de la machine distante restera valable. Vous devrez mettre le masque de votre réseau sur la ligne de commande de *pppd*.

10.4 gated - le démon de routage

gated pourrait être utilisé à la place de la *procuration ARP* et ce serait certainement plus propre, mais son utilisation principale est de faire de votre machine Linux un *routeur IP*. *gated* supporte plusieurs protocoles de routage, le plus commun, utilisé pour les petits réseaux, s'appellent *rip*. *rip* signifie *'Routing Information Protocol'* (Protocole d'Information de Routage). Si vous utilisez *gated*, configuré pour *rip*, votre machine Linux diffusera périodiquement une copie de sa *table de routage* sur votre réseau dans un format spécial. Avec cette méthode, toutes les autres machines de votre réseau sauront quelles sont les adresses accessibles à partir de votre machine.

gated devra être lancé par `rc.inet2`. Cela sera traité dans la section suivante. Vous avez peut-être un démon appelé *routed* qui tourne. *gated* est supérieur à *routed* parce qu'il est plus souple et plus fonctionnel. Vous devez utiliser *gated* et non *routed*.

10.4.1 Obtenir *gated*

Gated est disponible à :

sunsite.unc.edu

`/pub/Linux/system/Network/daemons/gated11_bin.tgz`

10.4.2 Installer *gated*

La distribution binaire de *gated* comprend trois programmes et deux exemples de fichiers de configuration.

Les programmes sont :

gated

le démon *gated*.

gdc

l'interface utilisateur de *gated*. *gdc* est utilisé pour contrôler le démon *gated* ; l'arrêter et le relancer, obtenir son état...

ripquery

est un outil de diagnostic pour demander les routes connues du routeur en utilisant des requêtes RIP de type 'query' ou 'poll'.

Les fichiers de configurations sont :

gated.conf

c'est l'actuel fichier de configuration du démon *gated*. Il vous permet de spécifier comment *gated* réagit quand il fonctionne. Vous pouvez autoriser ou inhiber un protocole de routage et contrôler le comportement des protocoles de routage actifs.

gated.version

un fichier texte qui donne la version du démon *gated*.

La distribution binaire de *gated* n'installe pas les fichiers aux endroits habituels. Heureusement ils sont peu nombreux.

Pour installer les binaires faire :

```
# cd /tmp
# gzip -dc .../gated.linux.bin.tgz | tar xvf -
# install -m 500 bin/gated /usr/sbin
# install -m 444 bin/gated.conf bin/gated.version /etc
# install -m 555 bin/ripquery bin/gdc /sbin
# rm -rf /tmp/bin
```

Je garde les démons réseau dans `/usr/sbin`; si les vôtres sont ailleurs vous devrez évidement les mettre où vous avez l'habitude. Les exemples de fichiers de configuration de *gated* incluent des configurations pour émuler l'ancien démon *routed*. Il fonctionnera sans doute pour vous dans la plupart des cas et il ressemble à cela:

```
#
#  Cette configuration emule routed.Elle execute RIP et n'envoie
#  des mises a jour s'il y a plus d'une interface en activite
#  et le forwarding IP est active dans le noyau.
#
#       NOTEZ que RIP *ne tournera pas* si la somme de controle UDP
#       n'est pas activee dans le noyau.
#
rip yes ;
traceoptions all;
#
```

Si vous avez des routes statiques à ajouter, vous pouvez le faire dans un paragraphe `static` ajouté à votre `/etc/gated.conf` comme suit:

```
#
static {
37.0.0.0 mask 255.0.0.0 gateway 44.136.8.97 ;
host 44.136.8.100 gateway 44.136.8.97 ;
} ;
#
```

L'exemple précédent crée une route statique dans le réseau Classe A `37.0.0.0` via la passerelle `44.136.8.97` et une route statique vers un hôte ayant l'adresse `44.136.8.100` via la passerelle `44.136.8.97`. Si vous faites cela vous n'avez pas besoin d'ajouter des routes en utilisant la commande *route* , *gated* ajoutera et gérera les routes à votre place.

Pour installer les pages de *man*, faire :

```
# cd /tmp
# gzip -dc .../gated.linux.man.tgz | tar xvf -
# install -m 444 man/*.8 /usr/man/man8
# install -m 444 man/*.5 /usr/man/man5
# rm -rf /tmp/man
```

Les fichiers *man* contiennent des informations précises et détaillées sur la configuration de *gated*. Pour trouver des informations pour configurer *gated*, regarder la page de *man* de `gated-config`.

11 Configurer les démons réseau

Comme mentionné précédement, il y a d'autres fichiers dont vous aurez besoin pour terminer votre installation réseau. Ces fichiers concernent des niveaux de configuration plus élévés des logiciels réseau. Chaque fichier important est décrit dans les sous-sections suivantes, mais en même temps vous trouverez qu'il y en a d'autres à configurer au fur et à mesure que vous deviendrez plus familiers avec les outils.

11.1 /etc/rc.d/rc.inet2 (la seconde moitié de rc.net)

Si vous avez suivi ce document pas à pas jusqu'ici vous devez avoir construit un fichier *rc* configurant vos interfaces réseau et mis en place le routage nécessaire à votre propre configuration réseau. Nous allons maintenant attaquer les couches plus hautes des logiciels réseau.

Vous devriez à ce stade lire le livre d'Olaf Kirch *Network Administrators Guide* : il devrait être considéré comme le document de référence de cette étape du processus de configuration. Il vous aidera à choisir ce qu'il faut inclure dans ce fichier et surtout ce qu'il **ne faut pas** inclure. Faites attention que plus on a de services réseau disponibles, plus on a de chances d'avoir des carences de sécurité ; n'utilisez que ce dont vous avez besoin.

Vous devez en connaître un peu plus sur certains *démons* importants (processus système tournant en tâche de fond). Les pages de *man* vous en diront plus, mais ce sont :

11.1.1 inetd

inetd est un programme qui tourne en tâche de fond et qui gère les requêtes de connexion Internet. Son but est d'éviter de laisser des serveurs tourner quand personne n'y est connecté. Quand il voit une requête pour un service particulier, comme par exemple *telnet*, ou *ftp*, il va lire le fichier `/etc/services`, trouver quel serveur doit traiter la requête, le lancer et lui passer la connexion. Imaginez que c'est le serveur maître de votre serveur Internet. Il possède aussi quelques services qu'il traite lui même. Ce sont les services *echo*, *discard* et *generate* utilisés à des fins de tests réseau. *inetd* ne gère pas **tous** les serveurs et services que vous devriez mettre en route, mais il le fait pour la plupart. Normalement des services tels que ceux basés sur `udp` ou des services qui gèrent eux-mêmes leur propre multiplexage de connexion, comme les serveurs World Wide Web ou muds doivent tourner indépendamment de *inetd*. En général la documentation les accompagnant vous ira s'il faut utiliser *inetd* ou non.

11.1.2 syslogd

syslogd est un démon qui gère les messages de trace des autres démons. Il aiguille ces messages suivant les règles établies par le fichier `/etc/syslogd.conf`. Par exemple pour certains types de messages, vous voudrez les envoyer sur la console et aussi dans un fichier d'archivage, alors que pour d'autres types vous voudrez les archiver uniquement dans un fichier. *syslogd* vous permet de décider quels sont les messages que vous voulez tracer et où ils doivent aller.

11.2 Un exemple de fichier rc.inet2

Le fichier ci-dessous est un exemple de fichier rc.inet2 que Fred a construit. Il lance un grand nombre de serveurs, vous devrez donc réduire ce nombre pour satisfaire vos besoins. Pour cela effacez ou commentez les paragraphes (entre *if* et *fi*) dont vous n'avez pas besoin. Chaque paragraphe teste si le programme existe, envoie un message que vous verrez au démarrage de votre machine puis exécute le programme en tâche de fond avec les arguments voulus. Pour plus d'information sur chacun de ces démons, consultez le *Network Administrators Guide* ou les pages de *man* correspondantes.

```
#! /bin/sh
#
# rc.inet2      Ce script lance le systeme INET complet.
#               Ce script monte aussi tous les disques NFS
#               importants (comme /usr), il faut donc faire
#               attention a ce que les binaires soient presents
#               maintenant...
#
# Version:      @(#)/etc/rc.d/rc.inet2  2.18    05/27/93
#
# Author:       Fred N. van Kempen, <waltje@uwalt.nl.mugnet.org>
#

# Constantes.
NET="/usr/sbin"
IN_SERV="lpd"
LPSPOOL="/var/spool/lpd"

# A ce point, nous sommes prets a parler au reste du monde...
echo -e "\nMounting remote file systems ..."
/bin/mount -t nfs -v            # This may be our /usr runtime!!!
```

```
echo -e "\nStarting Network daemons ..."
# Lance le demon SYSLOG. Ceci doit etre le premier serveur.
# C'est une OBLIGATION, donc laissez-le.
echo -n "INET: "
if [ -f ${NET}/syslogd ]
then
echo -n "syslogd "
${NET}/syslogd
fi

# Lance le SUN RPC Portmapper.
if [ -f ${NET}/rpc.portmap ]
then
echo -n "portmap "
${NET}/rpc.portmap
fi

# Lance le super serveur INET
# C'est une OBLIGATION, donc laissez-le.
if [ -f ${NET}/inetd ]
then
echo -n "inetd "
${NET}/inetd
else
echo "no INETD found.  INET cancelled!"
exit 1
fi

# Lance le serveur de nom NAMED/BIND.
# NOTE: probablement vous n'avez pas besoin de le lancer.
#if [ ! -f ${NET}/named ]
#then
#       echo -n "named "
#       ${NET}/named
#fi

# Lance le serveur ROUTEd.
# NOTE: routed est maintenant obsolete. vous devez utiliser gated
#if [ -f ${NET}/routed ]
#then
#       echo -n "routed "
#       ${NET}/routed -q #-g -s
#fi

# Lance le serveur GATEd.
if [ -f ${NET}/gated ]
then
echo -n "gated "
```

```
${NET}/gated
fi

# Lance le serveur RWHO.
if [ -f ${NET}/rwhod ]
then
echo -n "rwhod "
${NET}/rwhod -t -s
fi

# Lance le serveur U-MAIL SMTP.
if [ -f XXX/usr/lib/umail/umail ]
then
echo -n "umail "
/usr/lib/umail/umail -d7 -bd {$<$}/dev/null >/dev/null 2>&1 &
fi

# Lance les divers serveurs INET.
for server in ${IN_SERV}
do
if [ -f ${NET}/${server} ]
then
                echo -n "${server} "
                ${NET}/${server}
fi
done

# Lance les divers serveurs SUN RPC.
if [ -f ${NET}/rpc.portmap ]
then
if [ -f ${NET}/rpc.ugidd ]
then
                echo -n "ugidd "
                ${NET}/rpc.ugidd -d
fi
if [ -f ${NET}/rpc.mountd ]
then
                echo -n "mountd "
                ${NET}/rpc.mountd
fi
if [ -f ${NET}/rpc.nfsd ]
then
                echo -n "nfsd "
                ${NET}/rpc.nfsd
 fi

# Reveille le(s) demon(s) PC-NFS.
if [ -f ${NET}/rpc.pcnfsd ]
```

```
then
                echo -n "pcnfsd "
                ${NET}/rpc.pcnfsd ${LPSPOOL}
fi
if [ -f ${NET}/rpc.bwnfsd ]
then
                echo -n "bwnfsd "
                ${NET}/rpc.bwnfsd ${LPSPOOL}
fi

fi
echo network daemons started.
# Termine !
```

11.3 Autres fichiers de configuration réseau nécessaires.

Il y a d'autres fichiers de configuration que vous devez configurer si vous voulez que des personnes se connectent et utilisent votre machine en tant que hôtes. Si vous avez installer linux à partir d'une distribution, vous avez probablement copies de ces fichiers, dès lors vérifiez les pour voir s'ils sont corrects et si ce n'est pas le cas vous utiliserez les exemples suivants.

11.3.1 Un fichier /etc/inetd.conf en exemple.

Votre fichier `/etc/rc.d/rc.inet2` a déjà lancé *inetd*, *syslogd* et les divers serveurs *rpc* pour vous. Vous devez maintenant configurer les démons réseau qui seront gérés par *inetd*. *inetd* utilise un fichier de configuration appelé `/etc/inetd.conf`.

Ci-dessous un exemple simple de configuration:

```
#
# Les services internes.
#
# Auteurs:      Original taken from BSD UNIX 4.3/TAHOE.
#               Fred N. van Kempen, <waltje@uwalt.nl.mugnet.org>
#
echo    stream tcp nowait root  internal
echo    dgram  udp wait   root  internal
discard stream tcp nowait root  internal
discard dgram  udp wait   root  internal
daytime stream tcp nowait root  internal
daytime dgram  udp wait   root  internal
chargen stream tcp nowait root  internal
chargen dgram  udp wait   root  internal
#
# Services standards.
#
ftp     stream tcp nowait root  /usr/sbin/tcpd in.ftpd ftpd
```

```
telnet  stream tcp nowait root  /usr/sbin/tcpd in.telnetd
#
# Shell, login, exec et talk sont des protocolese BSD.
#
shell   stream tcp nowait root  /usr/sbin/tcpd in.rshd
login   stream tcp nowait root  /usr/sbin/tcpd in.rlogind
exec    stream tcp nowait root  /usr/sbin/tcpd in.rexecd
talk    dgram  udp wait   root  /usr/sbin/tcpd in.talkd
ntalk   dgram  udp wait   root  /usr/sbin/tcpd in.talkd
#
# Services d'informations et d'etats.
#
finger  stream tcp nowait root  /usr/sbin/tcpd in.fingerd
systat  stream tcp nowait guest /usr/sbin/tcpd /usr/bin/ps -auwwx
netstat stream tcp nowait guest /usr/sbin/tcpd /bin/netstat
#
# Fin de inetd.conf.
```

La page manuel de *inetd* décrit la signification de chaque ligne, mais plus simplement chaque entrée décrit quel programme est exécuté quand une connexion est reçue sur la socket listé en première entrée. Ces entrées qui possèdent `incoming` où le nom du programme et les arguments doivent être sont des services qui sont fournis de manière interne par le programme *inetd* .

La conversion entre le nom du service dans la première colonne et le numéro effectif de la socket auquel il se réfère est effectuée par le fichier `/etc/services`.

11.3.2 Un exemple de fichier /etc/services.

Le fichier `/etc/services` est simplement un tableau de noms de services Internet avec le numéro et le protocole utilisé. Ce tableau est utilisé par certains programmes comme *inetd*, *telnet* and *tcpdump*. Il simplifie un peu la vie en permettant de se référer à un service par son nom plutôt que par un chiffre.

Voici un exemple de fichier `/etc/services`:

```
#
# /etc/services - base de donnees incluant nom du service,
#                 numero de la socket et protocole.
#
# Auteur:
#     Fred N. van Kempen, <waltje@uwalt.nl.mugnet.org>
#
tcpmux     1/tcp
echo       7/tcp
echo       7/udp
discard    9/tcp   sink null
discard    9/udp   sink null
systat     11/tcp  users
daytime    13/tcp
```

```
daytime    13/udp
netstat    15/tcp
chargen    19/tcp  ttytst source
chargen    19/udp  ttytst source
ftp-data   20/tcp
ftp        21/tcp
telnet     23/tcp
smtp       25/tcp  mail
time       37/tcp  timserver
time       37/udp  timserver
name       42/udp  nameserver
whois      43/tcp  nicname    # usually to sri-nic
domain     53/tcp
domain     53/udp
finger     79/tcp
link       87/tcp  ttylink
hostnames  101/tcp hostname   # usually to sri-nic
sunrpc     111/tcp
sunrpc     111/tcp portmapper # RPC 4.0 portmapper TCP
sunrpc     111/udp
sunrpc     111/udp portmapper # RPC 4.0 portmapper UDP
auth       113/tcp authentication
nntp       119/tcp usenet     # Network News Transfer
ntp        123/tcp            # Network Time Protocol
ntp        123/udp            # Network Time Protocol
snmp       161/udp
snmp-trap  162/udp
exec       512/tcp            # BSD rexecd(8)
biff       512/udp comsat
login      513/tcp            # BSD rlogind(8)
who        513/udp whod       # BSD rwhod(8)
shell      514/tcp cmd        # BSD rshd(8)
syslog     514/udp            # BSD syslogd(8)
printer    515/tcp spooler    # BSD lpd(8)
talk       517/udp            # BSD talkd(8)
ntalk      518/udp            # SunOS talkd(8)
route      520/udp routed     # 521/udp too
timed      525/udp timeserver
mount      635/udp            # NFS Mount Service
pcnfs      640/udp            # PC-NFS DOS Authentication
bwnfs      650/udp            # BW-NFS DOS Authentication
listen     1025/tcp listener  # RFS remote_file_sharing
ingreslock 1524/tcp           # ingres lock server
nfs        2049/udp           # NFS File Service
irc        6667/tcp           # Internet Relay Chat
# End of services.
```

L'entrée *telnet* nous dit que le service *telnet* utilise la socket numéro 23 et le protocole *tcp*. L'entrée *domain* nous dit que le service de nom de domaine (DNS) utilise la socket 52 et à la fois les protocoles *tcp* et *udp*. Vous devez avoir une entrée adéquate dans `/etc/services` pour chaque entrée dans `/etc/inetd.conf`.

11.3.3 Un exemple de fichier /etc/protocols.

Le fichier `/etc/protocols` est un tableau de noms de protocoles avec son numéro correspondant. Puisque le nombre de protocoles en usage est faible ce fichier est très simple.

```
#
# /etc/protocols - base de donnees de protocoles.
#
# Auteur:
#   Fred N. van Kempen, <waltje@uwalt.nl.mugnet.org>
#
ip   0   IP   # internet protocol
icmp 1   ICMP # internet control message protocol
igmp 2   IGMP # internet group multicast protocol
ggp  3   GGP  # gateway-gateway protocol
tcp  6   TCP  # transmission control protocol
pup  12  PUP  # PARC universal packet protocol
udp  17  UDP  # user datagram protocol
idp  22  IDP
raw  255 RAW
#
# Fin
```

11.4 Résolution des noms

La résolution des noms est le processus qui consiste à convertir une adresse en notation pointée (comme `tsx-11.mit.edu`) en une adresse IP que les couches réseaux peuvent comprendre. Il y a deux façons de faire, une simple et une plus complexe.

11.4.1 /etc/hosts

`/etc/hosts` contient une liste des adresses IP et les noms des machines correspondants. De cette manière vous pouvez faire référence à une machine soit par son nom, soit par son adresse IP. En utilisant un serveur de noms (voir section 'named'), vous automatisez le même mécanisme de traduction nom->adresse IP. (Utiliser *named* vous permet d'utiliser votre propre serveur de noms sur votre machine Linux). Ce fichier contient au moins une entrée pour `127.0.0.1` avec comme nom `localhost`. Si vous n'utilisez pas uniquement l'interface loopback, vous aurez besoin d'ajouter une entrée pour votre adresse IP, avec votre nom complet (comme `loomer.vpizza.com`). Vous pouvez aussi ajouter des entrées pour vos passerelles et adresses réseau.

Par exemple, si `loomer.vpizza.com` a comme adresse IP `128.253.154.32`, le fichier `/etc/hosts` contiendra :

```
# /etc/hosts
```

```
# Liste des noms de machines et de leurs adresses IP.
127.0.0.1              localhost
128.253.154.32         loomer.vpizza.com loomer
# Fin de hosts
```

Bien sûr, vous aurez besoin d'éditer ce fichier pour qu'il corresponde à vos besoins. Si vous utilisez seulement le loopback, la seule ligne nécessaire doit être pour `127.0.0.1`, avec à la fois `localhost` et votre nom de machine.

Notez que pour la seconde ligne, il y a deux noms pour `128.253.154.32`: `loomer.vpizza.com` et juste `loomer`. Le premier nom est le nom complet de la machine, appelé « Nom de Domaine Complètement Qualifié », et le second est simplement un alias du premier nom.

11.4.2 named - en ai-je besoin ?

'Je T'implore...'

named est le démon serveur de nom pour la plupart des systèmes d'exploitation de style UNIX. Il permet à votre machine de répondre aux requêtes de mise en correspondance de nom, non seulement pour elle-même, mais aussi pour les autres machines sur le réseau. Autrement dit, si une autre machine veut trouver l'adresse de '`goober.norelco.com`', et que vous avez cette machine dans votre base de données *named*, alors vous pouvez répondre à cette requête et dire aux autres machines quelle est l'adresse de '`goobers`'.

Avec certaines vieilles implémentations de TCP/IP sous Linux, vous aurez besoin d'utiliser *named* pour créer des alias de noms de machine (même pour la vôtre). Le problème est que *named* est relativement difficile à paramétrer proprement et à maintenir. Pour résoudre ce problème, un programme appelé `hostcvt.build` a été créé pour traduire le fichier `/etc/hosts` dans l'ensemble des fichiers qui constituent les fichiers des bases de données de *named*. De toute façon même avec ce problème résolu, *named* utilise du CPU et crée du trafic réseau.

La morale est la suivante : **vous n'avez pas besoin d'utiliser named** sur votre système Linux. Les instructions de la distribution SLS vous indiqueront sûrement de lancer `hostcvt.build` pour initialiser *named*. C'est simplement inutile sauf si vous voulez utiliser votre machine Linux comme serveur de noms pour d'autres machines, et dans ce cas vous devrez en apprendre plus sur *named* de toute façon. Quand il cherche une correspondance pour un nom, votre machine Linux cherche d'abord dans le fichier `/etc/hosts`, puis demande au serveur de nom sur le réseau.

Les seules raisons que vous puissiez avoir d'utiliser *named* sont :

- Vous installez un réseau de machines et vous avez besoin d'un serveur de noms pour une des machines et vous n'avez pas d'autre serveur de noms sur le réseau.
- Votre administrateur réseau veut que votre système Linux serve de serveur de noms pour quelque raison que ce soit.
- Vous avez une liaison SLIP lente et vous voulez mettre au point un système de cache de noms pour que votre machine n'ait pas à communiquer sur la ligne série à chaque recherche de nom. Si vous aller être relié à un petit nombre de machines sur le réseau, et que vous connaissez leurs adresses, vous

pouvez les mettre directement dans le fichier `hosts` sans avoir besoin d'un serveur de noms du tout. En général, la mise en correspondance de noms n'est pas si lente et fonctionne très bien sur une liaison SLIP de toute façon.

– Vous voulez utiliser un serveur de noms pour le plaisir.

En général, **vous n'avez PAS besoin d'utiliser named**; cela signifie que vous devez commenter les lignes correspondantes dans le fichier `rc.inet2`, et vous ne devez pas lancer `hostcvt.build`. Si vous voulez des alias de noms de machines, par exemple, si vous voulez vous référez à `loomer.vpizza.com` avec `loomer`, vous pouvez simplement ajouter un alias au fichier `/etc/hosts` à la place. Il n'y a pas de raison d'utiliser *named* sauf si vous avez un besoin spécifique de le faire. Si vous avez accès à un serveur de noms, (votre administrateur réseau vous procurera son adresse), la plupart des réseau en ont un, ne vous ennuyez pas avec *named*.

Si vous utilisez seulement le loopback, vous pouvez lancer *named* et donner `127.0.0.1` comme adresse du serveur de noms, mais comme ce sera la seule machine à laquelle vous pourrez parler vous ne l'appelerez jamais.

11.4.3 /etc/networks

Le fichier `/etc/networks` liste les adresses de votre propre réseau et des autres. Il est utilisé par la commande *route*, et vous permet d'appeler un réseau par son nom si vous le voulez.

Pour chaque route vers un réseau que vous désirez ajouter par la commande *route*, vous devez avoir l'entrée correspondante dans le fichier `/etc/networks` sauf si vous donnez l'option `-net` sur la ligne de commande de *route*.

Son format est similaire au fichier `/etc/hosts`. Ci-dessous un exemple:

```
#
# /etc/networks: liste tous les reseaux pour lesquels vous voulez ajouter
#                une route
#
default          0.0.0.0         # route par defaut - recommande
loopnet          127.0.0.0       # reseau loopback - recommande
mynet            128.253.154.0   # Exemple de reseau CHANGER avec le VOTRE
#
# Fin de networks
```

11.4.4 /etc/host.conf

Le système possède quelques fonctions faisant partie de la librairie appelée *resolver*. Le fichier `/etc/host.conf` indique au système comment rechercher les correspondances des noms de machines. Il doit comprendre au moins les deux lignes suivantes :

```
order hosts,bind
multi on
```

Ces deux lignes indiquent à la librairie *resolver* de chercher d'abord dans le fichier `/etc/hosts` puis de demander au serveur de noms (s'il est présent). La ligne *multi* indique que vous pouvez avoir plusieurs adresses IP par nom de machine donnée dans `/etc/hosts`.

Ce fichier provient de l'implémentation de la librairie *resolv+* pour Linux. Vous trouverez plus de documentation dans la page de man de *resolv+(8)* si vous l'avez. Sinon vous pouvez l'obtenir à :

sunsite.doc.ic.ac.uk

```
/computing/comms/tcpip/nameserver/resolv+/resolv+2.1.1.tar.Z
```

Cette archive contient la page de man *resolv+.8* pour la librairie resolver.

11.4.5 /etc/resolv.conf

Ce fichier configure le système de résolution de noms. Il contient deux types d'entrées : l'adresse de votre serveur de noms (s'il y en a un) et le nom de votre domaine, si vous en possédez un. Si vous faites fonctionner votre propre serveur de noms (*named* tournant sur votre machine Linux), l'adresse de votre serveur de noms est `127.0.0.1`, l'adresse de loopback.

Votre nom de domaine est votre nom de domaine complètement qualifié sans votre nom de machine. Par exemple si votre nom complet est `loomer.vpizza.com`, alors votre nom de domaine est `vpizza.com`, sans le nom de machine `loomer`.

Par exemple, si votre machine est `goober.norelco.com`, et votre serveur de noms est à l'adresse `128.253.154.5`, alors votre fichier `/etc/resolv.conf` ressemblera à :

```
domain norelco.com
nameserver 128.253.154.5
```

Vous pouvez indiquer plus d'un serveur de noms. Chacun d'eux doit avoir une entrée *nameserver* dans le fichier `resolv.conf`.

Rappelez-vous que vous n'avez pas besoin de serveur de noms si vous n'utilisez que l'interface loopback.

11.4.6 /etc/HOSTNAME

Il ne vous reste plus qu'à configurer le nom de votre machine. Pour permettre à des applications comme *sendmail* d'accepter le courrier pour votre machine et à votre machine de s'identifier elle-même auprès des autres machines qui lui sont connectées.

Il y a deux programmes qui sont utilisés pour configurer ce type d'information et qui sont généralement mal utilisés. Ce sont *hostname* et *domainname*. Si vous utilisez une version de `net-tools` antérieure à la version `1.1.38` vous pouvez inclure une ligne dans votre fichier `/etc/rc` qui ressemble à :

```
/bin/hostname -S
```

cela signifie que la commande *hostname* va lire le fichier `/etc/HOSTNAME` qui devra contenir le nom de la machine complètement qualifié. *hostname* séparera ensuite le nom de la machine du nom de domaine et enregistrera les deux pour vous.

Par exemple la machine précédente devrait avoir le fichier `/etc/HOSTNAME` suivant :

```
goober.norelco.com
```

Par contre, si vous utilisez la commande *hostname* qui vient avec `net-tools-1.1.38` ou les versions suivantes, vous devrez ajouter la ligne suivante à la fin de votre fichier `/etc/rc.d/rc.inet1` :

```
/bin/hostname goober.norelco.com
```

ou si vous avez mis à jour à partir d'une version précédente :

```
/bin/hostname -F /etc/HOSTNAME
```

et tout devrait fonctionner comme avant.

La commande `/bin/domainname` est utilisée pour définir le nom de domaine **N.I.S.** et **non pas** le nom de domaine D.N.S. Vous n'avez pas à le mettre en place sauf si vous utilisez *NIS*, que l'on décrira plus loin.

11.5 Autres fichiers

Il y a bien sûr d'autres fichiers dans le répertoire `/etc` avec lesquels vous barboterez encore. Vous trouverez plus d'informations dans le livre d'Olaf *Network Administration Guide*. Il débute où cette *HOWTO* finit, plus de détails seront proposés dans les versions futures de ce document.

Une fois tous ces fichiers en place, chaque chose à sa place, vous devriez pouvoir relancer votre machine avec votre nouveau noyau et naviguer sur le réseau où bon vous semblera. Quoi qu'il arrive, je vous recommande vivement de conserver une copie de votre ancien noyau et même d'une 'disquette de secours' au cas où quelque chose ne tournerait pas rond pour pouvoir revenir en arrière et corriger le problème. Vous pouvez utiliser la disquette d'initialisation de H.J. Lu ou le disk1 de la distribution SLS.

12 Configurations avancées

Les configurations précédentes décrivent la façon d'installer une station Linux pour des utilisations normales. Certains d'entre vous ont peut-être des besoins qui requièrent des configurations légèrement plus avancées. Les exemples suivants sont les plus communs. Les détails concernant AX.25, Ottawa PI et les pilotes génériques SCC ont été mis sur (`http://sunsite.unc.edu/mdw/HOWTO/HAM-HOWTO.html`).

12.1 PPP - Protocole Point à Point

Le *Protocole Point à Point* est un protocole transportant d'autres protocoles, TCP/IP par exemple, sur des lignes séries, et que beaucoup de personnes utilisent à la place de SLIP. Il offre des fonctionnalités avancées, des options de corrections d'erreurs et gestion de la sécurité. Il corrige beaucoup de déficiences de SLIP et est utilisable aussi bien pour des liaisons asynchrones que synchrones.

Une caractéristique importante de PPP est l'allocation dynamique d'adresses, probablement exploitée par votre serveur PPP. Cette caractéristique permet à un client PPP de demander son adresse au serveur grâce à une requête spéciale. De cette manière, la configuration est plus propre qu'avec SLIP car la capacité à retrouver son adresse est externe au protocole.

Les auteurs de la version Linux sont `Michael Callahan, <callahan@maths.ox.ac.uk>` et `Al Longyear, <longyear@netcom.com>`. La plupart des informations que vous trouverez ici viennent de la documentation qui accompagne les logiciels PPP. Cette documentation est complète et vous en apprendra beaucoup plus.

Le code PPP de Linux est sorti de la phase de tests ALPHA et est maintenant disponible pour tout le monde. Le code PPP 1.0.0 de Linux est basé sur la version libre de PPP de Paul Mackerras pour les systèmes d'exploitation dérivés de BSD. La version 1.0.0 est basée sur la version 2.1.1 du code libre.

Le code PPP se compose de deux parties. La première est un module du noyau qui gère l'assemblage et le désassemblage des paquets. La seconde est un ensemble de protocoles appelés LCP, IPCP, UPAP et CHAP, qui négocient les options de la liaison, faisant fonctionner la liaison et gérant l'authentification.

12.1.1 Dois-je utiliser PPP à la place de SLIP ?

Vous utiliserez PPP à la place de SLIP pour plusieurs raisons. Les plus courantes sont :

Votre opérateur internet ne supporte que PPP

C'est la plus évidente des raisons pour utiliser PPP au lieu de SLIP.

Vous avez du bruit sur votre ligne série en temps normal

PPP fournit une séquence de vérification pour chaque paquet transmis, SLIP non. Si vous avez une ligne série bruitée, et que vous utilisez SLIP, votre correction d'erreur sera réalisée d'une extrémité à l'autre de la connexion IP, c'est-à-dire entre votre machine et la machine de destination, alors qu'avec PPP la détection d'erreur s'effectue localement sur la liaison entre votre machine et le serveur PPP. Ceci permet une récupération d'erreurs plus rapide.

Vous avez besoin d'une des caractéristiques propres à PPP.

PPP propose plusieurs possibilités que ne permet pas SLIP. Vous pouvez vouloir par exemple transporter non seulement de l'IP, mais aussi du DECNET, ou des paquets AppleTalk au dessus de votre liaison série. PPP vous permet de le faire.

12.1.2 Où obtenir les logiciels PPP

Le logiciel PPP est disponible à :

sunsite.unc.edu

```
/pub/Linux/system/Networking/serial/ppp-2.1.2d.tar.gz
```

Ce fichier contient le source du noyau et les sources et binaires de *pppd*. La version 1.0.0 est prévu pour être utilisée avec les noyaux 1.0.x et 1.1.x. La version 2.1.2 est prévue pour les noyaux 1.2.x, et la 2.2.0 pour les noyaux 1.3.x.

12.1.3 Installer le logiciel PPP

L'installation du logiciel PPP est très simple.

Le pilote du noyau. Un support pour PPP a été construit dans le noyau depuis quelque temps, dès lors vous êtes encouragés à utiliser un noyau récent. La configuration du noyau est très simple, les commandes suivantes devraient fonctionner :

```
# make config (rappelez-vous de repondre 'y' au support PPP)
# make dep
# make (pensez a installer le nouveau noyau apres la recompilation!)
```

Quand vous rebootez avec le nouveau noyau, vous devriez voir à peu près les messages suivants au moment du boot:

```
PPP: version 0.2.7 (4 channels) NEW_TTY_DRIVERS OPTIMIZE_FLAGS
TCP compression code copyright 1989 Regents of the University of California
PPP line discipline registered.
```

Ceci indique que le support PPP a bien été compilé dans votre noyau.

Maintenant, essayez de regarder le contenu de `/proc/net/dev`. Faites attention à ne pas utiliser *more* ou *less* dans le système de fichiers `proc` car ils vérifient la longueur du fichier en premier et c'est une caractéristique des fichiers de `proc` d'avoir une longueur nulle, aussi faites:

```
# cat /dev/proc/net
```

Il devrait ressembler à quelque chose comme :

```
Inter-|   Receive                  |  Transmit
face |packets errs drop fifo frame|packets errs drop fifo colls carrier
    lo:      0    0    0    0    0        0    0    0    0     0    0
  ppp0:      0    0    0    0    0        0    0    0    0     0    0
  ppp1:      0    0    0    0    0        0    0    0    0     0    0
  ppp2:      0    0    0    0    0        0    0    0    0     0    0
  ppp3:      0    0    0    0    0        0    0    0    0     0    0
```

Cela indique que le pilote noyau a été installé correctement.

pppd Pour extraire le logiciel pppd et les utilitaires l'accompagnant, ce qui suit devrait convenir:

```
# cd /usr/src
# gzip -dc ppp-2.1.2d.tar.gz | tar xvf -
```

Si vous voulez recompiler *pppd*, tapez *make* dans le sous-répertoire `pppd`. Il y aura des messages d'avertissement pendant la compilation de `lcp.c`, `upap.c` et `chap.c` mais tout sera OK.

Si vous voulez recompiler *chat*, consultez le fichier `README.linux` dans le répertoire `chat`.

Pour installer, taper *make install* dans les répertoires `chat` et `pppd`. Ces commandes installeront les binaires de *chat* et *pppd* dans `/usr/etc` et la page de man `pppd.8` dans `/usr/man/man8`.

`pppd` a besoin d'être lancé sous `root`. Vous pouvez positionner le bit suid root ou juste le lancer quand vous êtes root. *make install* essaiera de l'installer avec le bit suid root, donc si vous êtes root quand vous l'installez tout devrait fonctionner correctement.

12.1.4 Configurer et utiliser les logiciels PPP

Comme SLIP, vous pouvez configurer les logiciels PPP comme client ou comme serveur. Le programme *chat* joue le même rôle que *dip* dans le sens qu'il automatise la composition du numéro de téléphone et la connexion à la machine distante. A la différence de *dip*, il ne convertit pas la ligne série à la discipline PPP. Ceci est réalisé par le programme *pppd*. *pppd* peut servir de serveur ou de client. Quand vous l'utilisez comme un client, il appelle le programme *chat* pour établir la connexion puis il reprend la main pour faire l'*ioctl* qui change la discipline de la ligne pour celle de *ppp* et ensuite il vous laisse opérer.

Référez-vous aux pages de man de *pppd* et *chat* pour plus d'information. Vous pouvez aussi regarder le fichier README livré avec le logiciel PPP.

Configurer un client PPP utilisant un modem numéroteur. C'est peut-être ce que la plupart d'entre vous voudront faire, donc nous allons l'aborder en premier. Vous utiliserez cette configuration si votre fournisseur réseau propose des accès PPP par modem. Quand vous voulez établir la connexion vous lancez simplement *pppd* sans arguments. Ceci suppose que votre serveur PPP vous assignera une adresse de manière dynamique. Nous verrons plus loin le cas où vous avez une adresse statique. Le programme *pppd* a un certain nombre d'arguments en ligne qui peuvent modifier son comportement. Ces arguments peuvent être placés d'un fichier de configuration de telle manière que les commandes n'apparaissent pas visibles lorsque quelqu'un branché sur le système fait la commande *ps*. Ceci est particulièrement important quand *pppd* fait appel au programme chat: en effet il a besoin du mot de passe pour se connecter sur le serveur et en général vous ne souhaitez pas qu'il soit visible aux autres utilisateurs du système. Le fichier que *pppd* utilise s'appelle `/etc/ppp/options`. Un bon point de départ pour une installation 'typique' PPP peut ressembler à ceci:

```
connect /etc/ppp/ppp-connect
/dev/ttyS1
19200
crtscts
modem
lock
```

```
asyncmap 0
defaultroute
```

Cela signifie que:

1. *pppd* doit utiliser le script `/etc/ppp/ppp-connect` pour numéroter via le modem et se connecter.
2. *pppd* doit utiliser le dispositif `ttyS1` à `19200` bits par seconde, en utilisant le protocole d'échanges de données matériel `crtscts` et doit respecter les signaux du modem (particulièrement la détection de porteuse) pour détecter si le modem est toujours en ligne ou si l'appel est fini.
3. *pppd* doit créer un fichier `lock` pour le dispositif `tty` pour s'assurer qu'un autre processus n'essaie pas de l'utiliser en même temps que lui.
4. Que la ligne est 'bonne pour les 8 bits', c'est à dire que tous les caractères peuvent passer sur la ligne. L'option `asyncmap` détermine quels caractères *pppd* doit échapper et par défaut ceci est valable pour tous les caractères de contrôle. La plupart des serveurs PPP modernes sont aptes pour les 8 bits et mettre `asyncmap` à `0` enjoint *pppd* de n'échapper aucun caractère.
5. *pppd* doit créer une `defaultroute` via le dispositif ppp quand la connexion a réussi.

La prochaine étape consiste à configurer le script `/etc/ppp/ppp-connect`. Vous devriez normalement utiliser le programme *chat* livré, avec *pppd* à l'intérieur de ce script, étant données sa simplicité d'utilisation et sa brièveté. Pour commencer vous devez connaître à quoi ressemble la séquence de connexion de votre serveur PPP. Ce qui suit est un exemple fictif sur lequel je m'appuyerai.

```
CONNECT 14400

Bienvenue sur les serveur PPP XYZ !

login:                    <vous entrez votre nom d'utilisateur>^M
password:                 <vous entrez votre mot de passe>^M

Maintenant PPP:
```

Le programme *chat* attend comme arguments une séquence envoi/attente, signifiant qu'il veut une série de '*ce qu'il veut*', suivi par '*ce qu'il doit envoyer*'. `/etc/ppp/ppp-connect` **ne doit pas** être accessible en lecture. Un échantillon qui colle avec l'exemple fictif ci-dessus ressemble à cela:

```
#!/bin/sh
# Un script pour se connecter au serveur PPP XYZ
NUM=5552857
UID=terryd
PASSWD=secret1
#
/usr/sbin/chat -v "" ATZ OK ATDT$NUM CONNECT ogin: $UID word: \\q$PASSWD PPP:
```

J'ai ajouté au script une séquence d'initialisation avant de numéroter (C'est ATZ. Il est recommandé d'initialiser votre modem avant avant de démarrer).

Notez l'argument `""`, il signifie '*n'attendez rien*' et est utilisé quand vous voulez démarrer une séquence d'envoi.

L'argument `-v` enjoint à *chat* d'être **verbeux**. Dans ce mode il enverra au fichier *syslog* ce qui se passe pendant la connexion pour que vous puissiez voir ce qui s'est passé. Notez le
`q` avant `$PASSWD`. Ceci enjoint à *chat* de ne pas écrire ce qui suit dans *syslog*. De cette manière votre mot de passe ne sera pas enregistré. Il est bon de faire `chmod 600 /etc/ppp/ppp-connect` pour être sûr que les autres utilisateurs ne puissent le lire en vue de voir votre mot de passe. Pour démarrer une session PPP avec la configuration ci-dessus, il suffit de faire : *pppd* et tout se passera automatiquement. Vous devriez observer le début de session dans le journal du système et quand tout est fini vous devriez voir une route `default` dans votre table de routage quand vous utilisez la commande `route -n` pointée via votre nouveau dispositif `ppp0`.

Si vous utilisez une adresse IP statique, vous pouvez l'indiquer en incluant une ligne comme ceci dans votre fichier `/etc/ppp/options`:

```
nnn.nnn.nnn.nnn:
```

`nnn.nnn.nnn.nnn` étant votre adresse IP. Notez que cela ne marchera que si votre fournisseur de services Internet vous l'autorise. Les deux points ':' sont importants.

Il y a beaucoup d'autres options que vous pouvez mettre dans votre fichier `options`. Voir les pages de documentations de *pppd* et *chat* pour plus d'informations.

Configurer un client PPP sur une ligne spécialisée. Configurer un client PPP pour une ligne spécialisée est très rapide. Vous utiliserez encore le programme *pppd*, mais comme vous n'aurez pas besoin d'établir la connexion, les arguments du programme *chat* seront plus simples.

L'exemple que je présente ici suppose que le serveur PPP ne nécessite aucune procédure particulière. J'ai fait ce choix car chaque procédure de connexion est différente. Si vous voulez simplement utiliser une liaison locale, ce script devrait vous suffire.

```
pppd defaultroute noipdefault debug \
       kdebug 2 /dev/cua0 9600
```

Cette commande ouvre le port série, ouvre *ioctl* pour le changer en un dispositif *ppp*, définira la route par défaut sur l'interface *ppp*. Le paramètre *noipdefault* signifie que le programme *pppd* doit demander l'adresse à utiliser pour son interface au serveur. Les messages de debug passeront par *syslog*. L'argument *kdebug 2* génère des messages de debug de niveau 2, ce qui vous donnera plus d'informations sur le déroulement des opérations. Il utilisera le pilote `/dev/cua0` à 9600 bps.

Si votre serveur PPP a besoin d'une procédure de connexion spéciale, vous pouvez facilement utiliser le programme *chat* comme dans l'exemple du serveur appelant pour réaliser cette fonction pour vous.

Référez-vous aux pages de man de *pppd* et *chat* pour plus d'information. Vous pouvez aussi regarder le fichier README qui vient avec le logiciel PPP.

Configurer un serveur PPP. La configuration d'un serveur PPP est similaire à celle d'un serveur SLIP. Vous pouvez créer un compte 'ppp', qui utilisera un script exécutable comme shell de connexion. L'entrée du fichier `/etc/passwd` ressemblera à :

```
ppp:EncPasswd:102:50:PPP client login:/tmp:/etc/ppp/ppplogin
```

et le script `/etc/ppp/ppplogin` ressemblera à :

```
#!/bin/sh
exec /usr/etc/pppd passive :192.1.2.23
```

L'adresse indiquée sera celle qui sera assignée à la machine **appelante**.

Naturellement, si vous voulez que plusieurs utilisateurs puissent se connecter simultanément vous devrez créer autant de comptes et de scripts de connexion que vous voudrez autoriser d'accès PPP car vous ne pouvez utiliser qu'une seule adresse IP par script.

12.1.5 Où obtenir plus d'information sur PPP ou envoyer des rapports d'anomalies?

L'url `(http://sunsite.unc.edu/mdw/HOWTO/PPP-HOWTO.html)` est une excellente référence pour obtenir des informations plus compréhensible que celles que je donne ici. La plupart des discussions sur PPP pour Linux se tiennent sur la liste de diffusion PPP.

Pour joindre le canal **PPP** Linux sur le serveur de listes, envoyez un courrier électronique à :

```
Majordomo@vger.rutgers.edu
```

avec la ligne :

```
subscribe linux-ppp
```

dans le corps du message. La ligne sujet est ignorée.

N'oubliez pas de joindre tout le contexte de votre problème, ainsi que toutes les informations utiles pour votre rapport d'anomalie afin que ceux qui vous assisteront comprennent bien votre problème.

Vous pouvez aussi consulter :

Les RFC 1548, 1331, 1332, 1333 et 1334. Ce sont les documents de référence pour PPP.

W. Richard Stevens décrit aussi PPP dans son livre 'TCP/IP Illustrated Volume 1', (Addison-Wesley, 1994, ISBN 0-201-63346-9).

12.2 Configurer Linux comme serveur SLIP.

Si vous avez une machine qui doit être joignable par téléphone, et proposer des services réseau, vous devrez configurer votre machine en serveur. Si vous voulez utiliser SLIP comme protocole série, vous aurez le choix entre trois solutions pour configurer votre machine Linux comme serveur SLIP. Ma préférence va à la première présentée, *sliplogin*, car elle est la plus simple à configurer et à comprendre, mais je présenterai un résumé de chaque pour vous faire une opinion.

12.2.1 Serveur SLIP utilisant *sliplogin*

sliplogin est un programme que vous pouvez utiliser à la place du programme login normal pour les comptes SLIP. Il convertit la ligne du terminal en ligne SLIP.

L'appelant se connectera avec la procédure standard, en entrant son nom d'utilisateur puis son mot de passe, mais au lieu d'avoir un shell ensuite, *sliplogin* est exécuté et cherche une entrée dans son fichier de configuration (`/etc/slip.hosts`) qui corresponde au nom d'utilisateur. S'il en trouve une, il configure la ligne en mode 8 bits propre et utilise un appel *ioctl* pour convertir la ligne à la discipline SLIP. Quand cette procédure est terminée, *sliplogin* lance un script qui configure l'adresse IP, le masque réseau et les routes appropriées. Ce script est généralement appelé `/etc/slip.login`, mais d'une manière similaire à *getty*, si vous avez des appelants spéciaux qui nécessitent une initialisation spéciale, vous pouvez créer des scripts appelés `/etc/slip.login.loginname` qui seront exécutés à la place de celui par défaut. Il y a trois ou quatre fichiers que vous devez configurer pour que *sliplogin* travaille pour vous. Je vais détailler où et comment obtenir les fichiers et comment est configuré chacun en détail. Ces fichiers sont:

- `/etc/passwd` pour les comptes utilisateurs.
- `/etc/slip.hosts`, qui contient une information unique pour chaque utilisateur entrant.
- `slip.login`, qui gère la configuration de routage qui est nécessaire pour l'utilisateur.
- `slip.tty`, requis uniquement si vous configurez votre serveur pour une *allocation d'adresse dynamique* et contenant une table d'adresses à allouer.
- `/etc/slip.logout`, qui contient les commandes à effectuer quand l'utilisateur raccroche ou se déconnecte.

Où se procurer *sliplogin*. *sliplogin* peut être obtenu à :

sunsite.unc.edu

```
/pub/Linux/system/Network/serial/sliplogin-2.0.tar.gz
```

Le fichier tar contient à la fois les sources, les binaires précompilés et la page de *man*.

Pour être sûrs que seuls les utilisateurs autorisés pourront se servir de *sliplogin*, vous devez ajouter une entrée dans `/etc/group` similaite à celle-ci:

```
..
SLIP::13:radio,fred
..
```

Quand vous installez l'ensemble *sliplogin*, `Makefile` changera le nom de groupe propriétaire en `SLIP`, dès lors seuls ceux qui appartiennent au groupe pourront l'exécuter. L'exemple donné ci-dessus montre que seuls les utilisateurs radio et fred pourront exécuter *sliplogin*. Pour installer les binaires dans le répertoire `/sbin`, et la page de *man* dans la section 8, faire :

```
# cd /usr/src
```

```
# gzip -dc .../sliplogin-2.0.tar.gz | tar xvf -
# <..editez Makefile si vous n'utilisez pas les mots de passe shadow..>
# make install
```

Si vous voulez recompiler les binaires avant de les installer, ajouter la ligne `make clean` avant la ligne `make install`. Si vous voulez installer les binaires ailleurs, vous devrez éditer le fichier `Makefile` pour modifier la règle pour *install*. Lisez les fichiers `README` livrés avec l'ensemble pour plus d'informations.

Configurer `/etc/passwd` pour les hôtes SLIP. Vous aurez besoin de créer des comptes spéciaux pour les correspondants SLIP dans votre fichier `/etc/passwd`. Une convention répandue est de préfixer d'un 'S' le nom de la machine correspondante. Par exemple si la machine s'appelle `radio`, vous créerez une entrée dans `/etc/passwd` qui ressemblera à :

```
Sradio:FvKurok73:1427:1:radio SLIP login:/tmp:/sbin/sliplogin
```

La façon dont le compte est appelé n'a pas d'importance, tant qu'elle a une signification pour vous. Note : l'appelant n'a pas besoin d'un répertoire personnel, comme on ne lui présentera pas de shell, `/tmp` est donc un bon choix. Noter aussi que *sliplogin* est utilisé à la place du shell normal.

Configurer `/etc/slip.hosts`. *sliplogin* scrute le fichier `/etc/slip.hosts` pour trouver les entrées correspondant au nom de connexion. C'est dans ce fichier que vous indiquez l'adresse IP et le masque réseau qui seront assignés à l'appelant. Des entrées pour deux hôtes, une avec une configuration statique pour l'utilisateur hôte `radio` et l'autre avec une configuration dynamique pour l'utilisateur hôte `albert` ressembleraient à :

```
#
Sradio    44.136.8.99   44.136.8.100   0xffffff00   normal
Salbert   44.136.8.99   DYNAMIC        0xffffff00   compressed
#
```

Les entrées du fichier `/etc/slip.hosts` sont :

1. le nom de connexion de l'appelant.
2. l'adresse IP de la machine serveur, c'est-à-dire votre machine.
3. l'adresse IP qui sera assignée à l'appelant. Si ce champ est marqué comme `DYNAMIC` alors une adresse ip sera allouée suivant les informations contenues dans `/etc/slip.tty` décrit plus tard. **Note**: vous devez utiliser au moins la version 1.3 de sliplogin pour que cela marche.
4. le masque réseau assigné à la machine appelante en notation hexadécimale, par exemple 0xffffff00 pour un masque de réseau classe C, ainsi pour un réseau de classe C ce sera `0xffffff00`.
5. paramètres optionnels pour autoriser ou inhiber la compression ou d'autres possibilités.

Note : vous pouvez utiliser les noms ou les adresses IP en notation décimale pointée pour les champs 2 et 3. Si vous utilisez les noms, ils doivent être résolvables, c'est-à-dire que votre machine doit être capable de faire l'association entre le nom et l'adresse IP correspondante, sinon le script échouera au moment de l'appel. Vous pouvez tester la correspondance en utilisant le programme telnet avec comme paramètre le nom à tester. Si vous avez comme réponse 'Trying nnn.nnn.nnn...' alors votre machine a réussi à trouver la correspondance. Par contre si vous avez le message 'Unknown host', alors votre machine n'a pas su résoudre le nom. Si vous avez un problème mettez l'adresse en notation décimale pointée ou corrigez votre configuration de résolution. (Voir le paragraphe `Résolution de nom`).

Les paramètres optionnels les plus courants pour les champs `opt1` et `opt2` sont :

normal

pour établir SLIP sans-compression.

compress

pour établir une compression van Jacobsen des en-têtes (cslip)

Evidemment ces deux options sont mutuellement exclusives, vous pouvez utiliser l'une ou l'autre. Pour plus d'information sur les autres options consulter les pages de *man*.

Configurer le fichier `/etc/slip.login`. Une fois que *sliplogin* a recherché et trouvé dans le fichier `/etc/slip.hosts` une entrée correspondante, il essaiera d'exécuter le programme `/etc/slip.login` pour configurer l'interface SLIP avec son adresse IP et son masque réseau.

L'exemple du fichier `/etc/slip.login` qui vient avec le paquetage *sliplogin* ressemble à :

```
#!/bin/sh -
#
#       @(#)slip.login  5.1 (Berkeley) 7/1/90
#
# generic login file for a SLIP line.  sliplogin invokes this with
# the parameters:
#     $1       $2        $3       $4         $5        $6    $7-n
#   SLIPunit ttyspeed loginname local-addr remote-addr mask opt-args
#
/sbin/ifconfig $1 $4 pointopoint $5 mtu 1500 -trailers up
/sbin/route add $5
arp -s $5 <hw_addr> pub
exit 0
#
```

Vous noterez que ce script utilise simplement les commandes *ifconfig* et *route* pour configurer l'interface SLIP avec son adresse IP, l'adresse IP distante et le masque réseau, et crée une route pour l'adresse distante qui passe par l'interface SLIP. C'est exactement la même chose que si vous utilisiez la commande *slattach*. Notez aussi l'utilisation de *Procuration ARP* pour s'assurer que les autres hôtes sur le même ethernet que la machine sauront comment atteindre un hôte entrant. Le champ `hw_addr` doit être l'adresse matérielle de la carte ethernet sur la machine. Si vous n'avez pas de carte ethernet vous pouvez omettre cette ligne complètement.

Configurer le fichier `/etc/slip.logout`. Quand l'appel est terminé, vous voulez vous assurer que l'interface série est restaurée dans un état correct pour que les appels suivants réussissent. Ceci est réalisé grâce au fichier `/etc/slip.logout`. C'est assez simple, à nouveau, je vous présente l'exemple extrait du paquetage *sliplogin*.

```
#!/bin/sh -
#
#               slip.logout
#
/sbin/ifconfig $1 down
/sbin/route del $5
arp -d $5
exit 0
#
```

Il se contente de passer l'interface en mode 'down' et détruit la route manuelle créée précédemment. Il utilise aussi la commande *arp* pour enlever tout procuration arp en place, encore une fois vous n'avez pas besoin de la commande *arp* si vous n'avez pas de port ethernet.

Configurer le fichier `/etc/slip.tty` Si vous utilisez une allocation d'adresse ip dynamique (vous avez au moins un hôte configuré avec le mot-clé `DYNAMIC` dans le fichier `/etc/slip.hosts`), alors vous devez configurer le fichier `/etc/slip.tty` pour lister quelles adresses sont assignées à quel port. Vous n'avez besoin de ce fichier que si vous voulez que votre serveur alloue de manière dynamique des adresses aux utilisateurs.

Le fichier est un tableau qui liste les dispositifs *tty* qui supporteront les connexions d'entrée SLIP et les adresses ip qui seront assignées aux utilisateurs appelant ces ports. Son format est le suivant:

```
# slip.tty    tty-> adresse IP
# format: /dev/tty?? xxx.xxx.xxx.xxx
#
/dev/ttyS0      192.168.0.100
/dev/ttyS1      192.168.0.101
#
```

Ce que dit cette table est que tout appelant sur le port `/dev/ttyS0` qui a son champ d'adresse dans le fichier `/etc/slip.hosts` réglé sur `DYNAMIC` aura comme adresse allouée `192.168.0.100`.

Par ce biais vous n'avez besoin seulement que d'allouer une adresse par port pour tous les utilisateurs ne demandant pas une adresse fixe pour eux-mêmes. Cela vous permet de conserver le nombre d'adresses dont vous avez besoin au minimum sans gâchis.

12.2.2 Un serveur SLIP utilisant *dip*

Quelques-unes des informations ci-dessous proviennent des pages de man de *dip*, où la façon d'utiliser Linux comme serveur SLIP est décrite rapidement. Prenez garde aussi que ce qui suit est basé sur l'ensemble *dip337j-uri.tgz* et ne s'applique probablement pas à d'autres versions de *dip*. *dip* fonctionne suivant un mode où il place automatiquement une entrée pour l'utilisateur qui la réclame et configure la ligne série en tant que lien SLIP selon les informations qu'il trouve dans le fichier `/etc/diphosts`. Ce mode est activé en invoquant les fichiers *dip* et *diplogin*. Ceci est donc comment utiliser *dip* comme serveur SLIP, en créant des comptes spéciaux où *diplogin* est utilisé comme shell d'entrée. La première chose que vous devez faire est de créer un lien symbolique comme suit:

```
# ln -sf /usr/sbin/dip /usr/sbin/diplogin
```

Vous devez maintenant ajouter des entrées dans vos fichiers `/etc/passwd` et `/etc/diphosts`. Ces entrées sont formatées comme suit:

Pour configurer Linux comme un serveur SLIP, vous avez besoin de créer des comptes utilisateurs, où *dip* (en mode esclave) est utilisé comme shell de connexion. Fred suggère de suivre la convention suivante : tous les comptes SLIP débutent avec la lettre 'S' comme 'Sfredm'. Un exemple d'entrée dans `/etc/passwd` pour un serveur slip ressemble à:

```
Sfredm:ij/SMxiTlGVCo:1004:10:Fred:/tmp:/usr/sbin/diplogin
^^         ^^        ^^  ^^   ^^    ^^   ^^
|          |         |   |    |     |    \__ shell de connexion
|          |         |   |    |     |
|          |         |   |    |     \_______ Repertoire de connexion
|          |         |   |    \_____________ Nom d'utilisateur complet
|          |         |   \__________________ Groupe de l'utilisateur
|          |         \______________________ ID de l'utilisateur
|          \________________________________ Mot de passe chiffre
\___________________________________________ Nom de connexion de
                                             l'utilisateur SLIP
```

Une fois que l'utilisateur est connecté, le programme *login(1)* vérifie que l'utilisateur est valide puis exécute le programme *diplogin*. *dip*, quand il est appelé en tant que *diplogin*, sait qu'il doit automatiquement se comporter comme un shell de connexion. Quand il est démarré en tant que *diplogin*, la première chose qu'il fait est d'utiliser la fonction *getuid* pour obtenir l'ID user de celui qui l'a invoqué. Ensuite il cherche dans le fichier `/etc/diphosts` la première entrée qui correspond soit au ID user soit au nom du dispositif *tty* que l'appel a mis en marche et se configure alors de manière appropriée. Par un choix judicieux, soit en donnant à l'utilisateur une entrée dans le fichier `diphosts`, soit en laissant à l'utilisateur la configuration par défaut, vous pouvez construire votre serveur de telle sorte que vous pouvez avoir un mélange d'utilisateurs dont les adresses sont assignées de manière statique ou de manière dynamique. *dip* ajoutera automatiquement une entrée 'Procuration ARP' lorsque c'est demandé en mode esclave, aussi vous n'avez pas besoin de vous soucier d'ajouter manuellement de telles entrées.

Configurer `/etc/net/diphosts` Le fichier `/etc/net/diphosts` est utilisé par *dip* pour chercher la configuration pré-enregistrée des machines distantes. Ces hôtes distants peuvent être des utilisateurs qui entrent, ou bien sur des machines que avez connectées vous-mêmes.

Le format général pour le fichier `/etc/net/diphosts` est le suivant :

```
Suwalt::145.71.34.1:145.71.34.2:255.255.255.0:SLIP uwalt:CSLIP,1006
ttyS1::145.71.34.3:145.71.34.2:255.255.255.0:Dynamic ttyS1:CSLIP,296
..
```

Les champs sont les suivants:

1. `nom de login`: comme retourné par getpwuid(getuid()) ou nom de tty.
2. `non utilisé`: pour compatiblité avec passwd.
3. `Adresse de l'autre machine`: adresse IP de l'hôte appelant, soit sous forme numérique soit sous forme de nom.
4. `Adresse locale`: adresse de cette machine,soit numérique, soit nominative.
5. `Masque de réseau`: en notation décimale avec points.
6. `Champ commentaires`: à votre guise.
7. `protocole`: SLIP, CSLIP, etc..
8. `MTU`: nombre décimal.

Un exemple d'entrée `/etc/net/diphosts` pour un utilisateur SLIP distant peut-être:

```
Sfredm::145.71.34.1:145.71.34.2:255.255.255.0:SLIP uwalt:SLIP,296
```

qui décrit une liaison SLIP avec une adresse distante de 145.71.34.1 et un MTU de 296, ou bien:

```
Sfredm::145.71.34.1:145.71.34.2:255.255.255.0:SLIP uwalt:CSLIP,1006
```

qui décrit une liaison permettant la compression CSLIP avec un MTU de 1006.

Dès lors, tous les utilisateurs qui auront la permission d'avoir un accès IP statique doivent avoir une entrée dans `/etc/diphosts` et si vous voulez que les utilisateurs appelant un port particulier aient leur adresse allouée dynamiquement vous devez avoir une entrée pour le dispositif `tty` et ne pas configurer une entrée pour les utilisateurs. Vous devez vous rappeler de configurer au moins une entrée pour chaque `tty` de façon à ce que les utilisateurs entrant aient ait une configuration adéquate sans se soucier du modem sur lequel ils sont. Quand un utilisateur se branche, il reçoit une invite normale pour le login et le mot de passe, à laquelle il doit répondre par son ID user SLIP et son mot de passe. Si tout se passe bien, l'utilisateur ne voit pas de message spécial, il se mettra en mode SLIP pour effectuer la connexion, et sera configuré suivant les paramètres du fichier `diphosts`.

12.2.3 Serveur SLIP utilisant le paquetage *dSLIP*.

Matt Dillon, <dillon@apollo.west.oic.com> a rassemblé quelques programmes de gestion des connexions slip, aussi bien en entrée qu'en sortie. Cette collection est une collection de petits programmes et de scripts qui gèrent les connexions pour vous. Vous devrez posséder *tcsh* car au moins l'un des scripts en a besoin. Matt fournit une copie binaire de l'utilitaire *expect* car il est réclamé également par l'un des scripts. Il serait bon que vous ayez quelque expérience de *expect* pour que ce paquetage fonctionne pour vos liaisons, mais que cela ne vous rebute pas.

Matt a bien décrit dans le fichier README les instructions nécessaires, aussi je ne m'ennuyerai pas à les répéter.

Ce paquetage peut être obtenu sur: **appolo.west.oic.com**

```
/pub/linux/dillon_src/dSLIP203.tgz
```

ou bien: **sunsite.unc.edu**

```
/pub/Linux/system/Network/serial/dSLIP203.tgz
```

Lisez le fichier `README` et créez les entrées dans `/etc/passwd` et `/etc/group` avant de faire `make install`.

12.3 Utiliser le démon d'automontage - AMD

Cette section m'a été proposée par `Mitch DSouza`, et je l'ai incluse avec un minimum de changements. Merci Mitch.

12.3.1 Qu'est-ce qu'un automonteur et pourquoi l'utiliser ?

Un *automonteur* propose un moyen pratique de monter des systèmes de fichiers à la demande. Cela permet de réduire les charges du serveur et du client et aussi apporte une grande souplesse même avec des montages autres que NFS. Il offre aussi un mécanisme de redondance grâce auquel on peut passer à un serveur secondaire si le primaire est inaccessible. Un autre montage intéressant est appelé le montage *union* qui donne à l'*automonteur* la possibilité de *fusionner* plusieurs répertoires en un seul. La documentation doit être examinée en détail pour tirer pleinement avantage de sa puissance.

Quelques points importants à se rappeler - (sans ordre particulier) :

- les cartes *amd* ne sont **pas** compatibles avec les cartes SUN, qui à leur tour ne sont **pas** compatibles avec les cartes HP *et ainsi de suite*. Le point clé est que *amd* est libre d'accès et compatible avec tous les systèmes décrit précédemment et d'autres encore, ce qui vous permet de partager les cartes sur votre réseau si vous utilisez *amd* partout sur votre réseau. Mitch l'utilise avec un ensemble de machines Linux/Dec/NeXt/Sun.
- Les cartes de l'automonteur *Sun* peuvent être converties au format *amd* en utilisant un script *perl* dans le répertoire `contrib` - `automount2amd.pl`.

- Vous **devez** avoir le *portmapper* fonctionnant **avant** de lancer *amd*.
- les montages UFS **ne provoquent pas** de timeout.
- les montages UFS, dans le cas de Linux **seulement**, ont été étendus pour gérer **toutes** les variétés de systèmes de fichiers natifs (i.e. minix, ext, ext2, xiafs ...) le défaut étant minix. Cette possibilité non documentée est accessible avec l'option *opts* comme dans :

  ```
  ..., opts:=type=msdos,conv=auto
  ```

- Ne montez pas par dessus des répertoires existants sauf si vous utilisez l'option *direct*, sinon cela revient à monter votre disque sur /home alors que votre répertoire de connexion est `/home/fred`.
- **Toujours** mettre l'option de trace maximum '-x all' si vous avez des problèmes. Vérifier aussi ce que la commande :

  ```
  % amq -ms
  ```

  renvoie; elle vous indiquera les problèmes.
- GNU *getopt()* est parfois trop intelligent pour son propre bien. Utilisez toujours '`--` à la fin des options :

  ```
  # /etc/amd -x all -l syslog -a /amd -- /net /etc/amd.net
  ```

12.3.2 Où trouver AMD, le démon d'automontage

amd peut être obtenu à :

sunsite.unc.edu

```
/pub/Linux/system/Misc/mount/amd920824upl67.tar.gz
```

Il contient les binaires prêts à être installés, les sources complets et la documentation au format texinfo.

12.3.3 Exemple de configuration AMD

Vous ne configurez pas l'*automonteur* à partir du fichier `/etc/fstab`, qui doit déjà contenir les informations à propos de vos systèmes de fichiers. Tout est en fait géré par des commandes.

Pour monter deux systèmes de fichiers *nfs* en utilisant votre fichier `/etc/fstab`, vous devrez mettre les deux entrées suivantes :

```
server-1:/export/disk  /nfs/server-1  nfs  defaults
server-2:/export/disk  /nfs/server-2  nfs  defaults
```

Ce qui signifie que vous montez *nfs* `server-1` et `server-2` sur votre disque linux dans les répertoires `/nfs/server-1` et `/nfs/server-2`.

Après avoir commenté ou détruit les lignes précédentes de votre fichier `/etc/fstab`, vous pouvez utiliser *amd* pour réaliser la même tâche en utilisant la syntaxe suivante :

```
/etc/amd -x all -l syslog -a /amd -- /nfs /etc/amd.server
|      | |    | |       | |     |  | |  | |             |
|      | |    | |       | |     |  | |  | |             |
`______' `____' `_______' `_____' _' `__' `_____________'
 |        |      |         |      |   |    |
 (1)      (2)    (3)       (4)    (5) (6)  (7)
```

où :

1. Le chemin complet du binaire *amd* (évidement optionnel) sauf si votre variable $PATH pointe dessus, si c'est le cas tapez juste *'amd'*.
2. '`-x all`' signifie trace maximum. Lire la documentation pour les autres niveaux de trace.
3. '`-l syslog`' signifie enregistrer les messages via le démon *syslog*. Ce qui peut signifier les envoyer dans un fichier, ou dans une console inutilisée. Cette option (*syslog*) peut être remplacée par le nom d'un fichier. '-l foo' enregistrera dans un fichier nommé *foo*.
4. '`-a /amd`' signifie utiliser le répertoire `/amd` comme répertoire temporaire pour les points d'automontage. Ce répertoire est créé automatiquement par *amd* et doit être détruit avant de lancer *amd* dans vos srcipts `/etc/rc`.
5. '`--`' indique à *getopt()* d'arrêter d'essayer de trouver des options à la suite. C'est particulièrement utile pour passer l'option 'type:=' sur la ligne de commande, sinon *getopt()* tentera de la décoder de manière incorrecte.
6. '`/nfs`' est le point de montage nfs **réel**. A nouveau il est automatiquement créé et ne doit **pas** en général contenir de sous-répertoires sauf si l'option 'type:=direct' est utilisée.
7. La carte *amd* (un fichier) appelé 'amd.server' contient les lignes :

```
# /etc/amd.server
/defaults    opts:=rw;type:=nfs
server-1     rhost:=server-1;rfs:=/export/disk
server-2     rhost:=server-2;rfs:=/export/disk
```

Une fois lancé et pendant l'exécution, vous pouvez demander l'état des points de montage avec la commande :

```
% amq -ms
```

Maintenant si vous tapez :

```
% ls /nfs
```

vous ne devriez voir aucun fichier. Par contre la commande :

```
% ls /nfs/server-1
```

montera la machine 'server-1' automatiquement. *Et voilà* ! *amd* fonctionne. Une fois le délai par défaut écoulé, le répertoire sera automatiquement démonté. Votre fichier `/etc/password` peut contenir les entrées suivantes :

```
 ...
linus:EncPass:10:0:God:/nfs/server-1/home/linus:/bin/sh
mitch:EncPass:20:10:Mitch DSouza:/nfs/server-1/home/mitch:/bin/tcsh
matt:EncPass:20:10:Matt Welsh:/nfs/server-1/home/matt:/bin/csh
```

qui signifient que lorsque Linus, Matt, ou Mitch se connecteront leur répertoire de connexion sera automatiquement monté sur le serveur approprié et démonté quand ils se déconnecteront.

12.4 Utiliser Linux comme routeur

Linux fonctionne très bien comme routeur. Vous devez faire fonctionner un démon de routage comme *gated*, ou si vous avez des besoins très simples utiliser des routes codées en dur. Si vous utilisez un noyau récent (1.1.*) assurez-vous de répondre 'y' à :

```
IP forwarding/gatewaying (CONFIG_IP_FORWARD) [y] y
```

quand vous compilez le noyau.

Le livre d'Olaf Kirch *Network Administrators Guide* traite du sujet du routage et de la mise en place de réseau. Vous pouvez le lire si vous avez besoin de plus d'information. Vous trouverez les références exactes dans la section 'Documentation utile'.

12.5 NIS-Sun Network Information System

Il y a un url `(http://sunsite.unc.edu/mdw/HOWTO/NIS-HOWTO.html)` que vous devez lire si vous êtes intéressés par NIS. Il détaille comment obtenir, installer et configurer le système NIS pour Linux.

13 Modules expérimentaux et en cours de développement

Il y a un grand nombre de personnes travaillant sur des nouveautés et des modules pour le code réseau du noyau Linux. Certains sont dans un état avancé (voir la partie *en fonctionnement*), et c'est ceux là que j'ai inclus dans cette section en attendant qu'ils deviennent standards et donc que je les mettent dans une section précédente.

La section AX25 a été relogée dans `(http://sunsite.unc.edu/mdw/HOWTO/HAM-HOWTO.html)` parce qu'elle est plutôt spécifique aux Radio-amateurs.

13.1 Un agent snmp

`Erik Schoenfelder,` <`schoenfr@ibr.cs.tu-bs.de`> a porté pour Linux un agent snmp expérimental.

Il est disponible à :

ftp.ibr.cs.tu-bs.de

`/pub/local/cmu-snmp2.1.2l3.tar.gz`

Lisez le fichier `cmu-snmp2.1.2l3.README`, il contient certaines informations que vous aurez à connaître à propos de ce paquetage.

Ce paquetage propose un ensemble quasi complet de variables MIB-II. Pour l'instant, vous pouvez seulement lire les variables.

`nstat.tar.gz` contient un formateur pour la sortie de `/proc/net/snmp` appelé *nstat*.

Vous aurez besoin d'utiliser Linux v1.1.60 et libc v4.6.27 ou plus récent pour compiler et faire fonctionner l'agent snmp.

13.2 Pilote Token Ring Experimental

`Peter De Schrijver` <`stud11@cc4.kuleuven.ac.be`> a développé un pilote Token Ring expérimental. Sa dernière version au moment où j'écris est disponible dans les noyaux Linux.1.3.*. Pour les plus vieilles versions, les patches sont situés sur:

linux3.cc.kuleuven.ac.be

`/pub/Linux/TokenRing/TokenRing.patch-1.2.0.gz`

ftp.cs.kuleuven.ac.be

`/pub/unix/linux/TokenRing.patch-1.1.64.gz`

Il y a un grand nombre de patches adaptés à différents noyaux. Prenez celui qui vous convient.

Notez également qu'il existe des versions d'outils réseau adaptés à TokenRing dans le même répertoire sur `linux3.cc.kuleven.ac.be`.

La plupart des cartes basées sur le circuit TROPIC d'IBM peuvent fonctionner maintenant. Les cartes suivantes sont réputées pour fonctionner avec le pilote:

- IBM Token Ring Adapter II
- IBM Token Ring 16/4 Adapter
- IBM Token Ring Adapter/A
- IBM Token Ring 16/4 Adapter/A
- HyperRing Classic 16/4

Les cartes utilisant le circuit TI ou le busmastering DMA ne fonctionneront pas avec le pilote. Cependant quelqu'un travaille sur des adaptateurs pour les bus IBM.

13.3 Carte d'interface V.35

V.35 est un standard C.C.I.T.T. qui offre une interface série équilibrée à haute vitesse, pour des vitesses pouvant aller jusqu'à 2 Mbps environ. L'utilisation d'une transmission équilibrée par couple différentiel permet à l'interface V.35 de supporter des câbles plus longs que l'interface plus commune de type V.24/RS232C, ainsi que des vitesses de transmission plus élevées.

13.3.1 Internet Technologies International XNET-1

Matti Aarnio <`mea@mea.utu.fi`> représente *Internet Technologies International* (`mailto:intec@digiw.fi`). Ils ont une carte 8 bits X.21 appeleée `XNET-1` qui fonctionne avec le V.35 avec un câble approprié. Vous pouvez obtenir des informations sur la carte sur:

```
Internet Technologies International OY,
Helsinki, Finland  <intec@digiw.fi>
Internet Technologies International OY (Ltd)
PO BOX 51
00641 Helsinki
FINLAND
Fax: +356-0-7523939
```

Le pilote prévu pour la carte se trouve sur: *ftp.funet.fi* (`ftp://ftp.funet.fi/pub/Linux/kernel/xnet-sync-driver.tar.gz`)

13.3.2 Carte SDL Communications N2.

SDL Communications (`http://www.ici.net/cust_/sdl/sdl.html`) produit plusieurs cartes contrôleurs supportant V.35 pour des machines ISA.

La version beta du pilote se trouve sur:

ftp.sdlcomm.com

`/business/sdlcomm/n2/linux`

Pour d'autres informations contactez *Rubin Dhillon at SDL Sales* (`mailto:sales@sdlcomm.com`)

13.4 programme passerelle IPX

Vinod G Kulkarni <`vinod@cse.iitb.ernet.in`> a coécrit un programme pour linux qui lui permet d'agir comme passerelle IPX.

On trouve le programme sur:

sunsite.unc.edu

```
/pub/Linux/Systems/Network/router/ipxbridge.tar.gz
```

13.5 support IPX RIP et SAP.

Alex Liu <`labrat@unitrx.com`> a écrit un support pour les protocoles Novell RIP et SAP qui permet à votre machine linux d'agir comme un routeur Novell.

Ce programme est au stade **alpha** et inclut un patch pour le noyau. Soyez prévenu que vous devez prendre les précautions habituelles quand vous essayez ce logiciel.

Vous pouvez obtenir le logiciel sur:

sunsite.unc.edu

```
/pub/Linux/Incoming/ipxripd-002.tar.gz (Jusqu'a ce qu'il soit deplace)
/pub/Linux//system/Network/router/ipxripd-002.tar.gz
```

Un fichier `README` est inclus et vous devez le lire pour les détails d'installation et de configuration.

13.6 Paquetage de demande de numérotation SLIP/PPP

Eric Schenk <`schenk@cs.toronto.edu`> a écrit un démon de demande de numérotation qui fonctionne soit avec SLIP soit avec PPP. Il suppose que vous avez un dispositif SLIP configuré de telle sorte que le démon s'y connecte via un pty. Quand votre connexion SLIP n'est pas active tous les datagrammes des hôtes distants seront routés sur ce dispositif et le démon les détectera, quand il reçoit un datagramme il exécute un script pour activer votre liaison réseau et donc reroute les datagrammes sur cette liaison.

Le logiciel se trouve sur:

sunsite.unc.edu

```
/pub/Linux/system/Network/serial/diald-0.7.tar.gz
```

Note: Vous devez configurer votre noyau avec le pilote SLIP, même si vous voulez tourner avec PPP.

La documentation fournie avec montre comment installer et configurer le programme.

13.7 support ISDN

Il y a plusieurs support ISDN en train. Ceux que je connais sont:

13.7.1 pilote 3Com-Sonix PC/Arpeggio

Alan Cox a commencé un pilote pour la carte ISDN 3Com arpeggio. Actuellement il ne supporte que PPP Async.

sunsite.unc.edu

`/pub/Linux/kernel/patches/network/sonix.tgz`

13.7.2 carte DIEHL SCOM

sunsite.unc.edu

`/pub/Linux/kernel/patches/network/isdndrv-0.1.1.tar.gz`

13.7.3 carte ICN ISDN et/ou Teles ISDN et/ou carte Creatix AVM ISDN

Fritz Elfert, Jan den Ouden et Thinking Objects ont écrit un pilote linux pour la carte ICN ISDN. Une partie de la documentation concernant ce pilote est en langue allemande. Il semble qu'il y ait différentes versions de ce pilote qui apparemment supportent d'autres cartes ISDN, si vous en obtenez un et qu'il ne supporte pas votre carte jetez un coup d'oeil sur un autre dans les différents sites. Vous pouvez en obtenir sur:

ftp.franken.de

`/pub/isdn4linux/`

ou: **ftp.gwdg.de**

`/pub/linux/misc/isdn4linux/`

13.7.4 carte Euro-ISDN ?

Matthias Urlichs <`urlichs@smurf.noris.de`> a développé quelques séquences expérimentales pour supporter ISDN sous Linux. Le plus récent est pour les noyaux version 1.3.* et il semblerait qu'il fonctionne.

On le trouve sur:

ftp.uni-stuttgart.de

`/pub/systems/linux/isdn/ftp.noris.de/`

Attention, la documentation est en Allemand et le code est conçu pour supporter le réseau ISDN européen. Matthias travaille sur une version Anglaise de la documentation.

13.7.5 carte ISDN Combinet's Everyware 1000

Joel Katz <`stimpson@panix.com`> a developpé un pilote pour les séries de cartes enfichables ISDN Combinet's `Everyware 1000`.

On le trouve sur: **sunsite.unc.edu**

`/pub/Linux/kernel/patches/network/combinet1000isdn-1.02.tar.gz`

13.7.6 Plus d'informations:

Harald Milz a mis ensemble une page World Wide Web pour le support ISDN Linux que l'on trouve sur: *Linux ISDN Web Page* (`http://www.ix.de/ix/linux/linux-isdn.html`). Un site plus approprié contenant des informations sur ISDN est accesible sur: *Dan Kegel's ISDN home page* (`http://alumni.caltech.edu/ dank/isdn/`).

13.8 Systèmes de transmission System-V

Le projet LiS (Linux Streams) fut formé pour développer et implémenter les systèmes de transmission System-V pour Linux. Il existe une page World Wide Web page à *Linux Streams Web Page* (`http://www.uc3m.es/ LiS/`) où vous pouvez obtenir l'état du projet et où vous pouvez avoir des informations et le code test.

13.9 Support ATM (Asynchronous Transfer Mode) .

Werner Almesberger a mis au point un site WWW sur le sujet. On le trouve sur *lrcwww.epfl.ch* (`http://lrcwww.epfl.ch/linux-atm/`). Il y a un logiciel expérimental qui supporte les connexions ATM brutes et IP sur ATM, ceci au stade alpha, que l'on peut obtenir sur le site Web. Le site contient un grand nombre d'informations intéressantes sur ATM.

Un abonnement à une liste de courrier a été mis en route pour discuter d'ATM pour Linux, et vous pouvez y souscrire en envoyant au coeur du message `subscribe linux-atm` à: `majordomo@vger.rutgers.edu`

13.10 support X.25.

Beaucoup de personnes se sont inquiétées au sujet d'un support X.25 pour Linux. Il y a eu un certain nombre de développement, mais apparemment la petite communauté qui s'y intéresse n'a pas assez de temps pour produire rapidement. Si vous êtes intéressés par le support X.25, vous devriez vous porter volontaires pour aider et accélérer ainsi l'effort de développement. La seule information que j'ai sur les travaux en cours de développement concernant X.25 est une note sur le serveur Web d'Alan Cox que quelqu'un s'est porté volontaire pour faire le travail et qui donne le site WWW suivant donnant des informations au sujet de deux développements qui semblent avancer:

X.25 pour Linux est à: *www.ping.be* (`http://www.ping.be/ ping0778/linux-x.25/`)

13.11 support Appletalk.

Si vous avez une machine Apple et vous voudriez qu'elle puisse partager sur le réseau des fichiers et des imprimantes avec votre machine linux, alors le support appletalk linux est fait pour vous.

Les informations suivantes ont été fournies pour la plupart par Cees de Groot <`cg@tricbbs.fn.sub.org`> qui l'a compilé à partir de sa propre expérience avec l'aide de Dana Basken <`dana@millenium.tiac.net`> et d'autres sur le réseau.

13.11.1 Construire le noyau avec le support Appletalk DDP

Alan Cox a développé un support expérimental et qui marche pour la spécification Apple Ethertalk. Il fait partie maintenant des derniers noyaux 1.2.* et des noyaux 1.3.*, dès lors pour compiler le support noyau vous n'avez qu'à compiler un noyau récent, et soyez certains d'avoir choisi l'option suivante de configuration:

```
#
# cd /usr/src/linux
# make config
 ...
 ...
    Appletalk DDP ? y
 ...
 ...
#
```

13.11.2 Construire les outils de configuration Appletalk et les démons.

Vous aurez besoin de logiciels supplémentaires car le support du noyau fournit la configuration de base pour réseau, mais pas les protocoles de couche supérieures.

Wesley Craig <netatalk@umich.edu> représente une équipe appelée Research Systems Unix Group à l'Université de Michigan qui a produit un ensemble appelé *netatalk* qui fournit le logiciel implémentant l'ensemble du protocole Appletalk plus quelques utilitaires.

Vous pouvez l'obtenir sur:

University of Michigan (ftp://terminator.rs.itd.umich.edu/unix/netatalk/)

il faut au moins la version 1.3.3b2 car les précédentes ne supportent pas Linux.

Pour construire et installer l'ensemble:

```
# cd /usr/src
# tar xvfz .../netatalk
- Editez ‘Makefile’, si vous voulez changer la variable DESTDIR qui
  definit l’endroit ou les fichiers seront installes plus tard.
  Par defaut c’est /usr/local/atalk qui est un endroit sur.
- Vous n’avez maintenant juste qu’\‘{a} faire ‘make’. Mon programme ‘uname’   n’ etait pas dans le repertoire /bin, j’ai donc editer ‘Makefile’ pour le mentionner.
# make
- en tant que root:
# make install
```

Après avoir terminé vous avez un grand nombre de nouvelles commandes avec lesquelles vous pourrez jouer.

13.11.3 Configurer le logiciel Appletalk.

La première chose à faire pour que tout marche est d'ajouter le contenu du fichier services.atalk à votre fichier /etc/services, juste à la fin de ce fichier.

L'étape suivante consiste à créer les fichiers de configuration appletalk dans le répertoire `/usr/local/atalk/etc` (ou bien celui où vous avez installé l'ensemble).

Le premier fichier à créer est `/usr/local/atalk/etc/Atalkd.conf`. Pour démarrer ce fichier n'a besoin que d'une seule ligne qui donne le nom du dispositif qui supporte le réseau sur lequel sont vos machines Apple:

```
eth0
```

Le programme démon Appletalk ajoutera des choses quand il tournera.

13.11.4 Exporter un système de fichiers Linux via Appletalk.

Vous pouvez exporter des systèmes de fichiers depuis votre machine linux via le réseau de telle manière qu'une machine Apple puisse les partager.

Pour ce faire vous devez configurer le fichier: `/usr/local/atalk/etc/AppleVolumes.system` .

Tous les détails pour le faire et quelles sont les options peuvent être trouvés dans la page d'aide *afpd*.

Un exemple:

```
/tmp Scratch
/home/ftp/pub Public
```

Ce qui exportera votre système de fichiers `/tmp` en tant que volume AppleShare 'Scratch' et votre répertoire public ftp en tant que volume AppleShare 'Public'. Les noms de volume ne sont pas obligatoires, le démon les choisissant pour vous, mais ça ne fait pas de mal de les déclarer quand même.

13.11.5 Partager votre imprimante Linux avec Appletalk.

Vous pouvez partager votre imprimante linux avec vos machines Apple très simplement. Vous devez mettre en route le programme *papd* qui est le démon protole d'accès à l'imprimante Appletalk (Appletalk Printer Access Protocol Daemona). Quand vous utilisez ce programme, il accepte des requêtes provenant de vos machines Apple et met en tâche de fond le travail d'impression sur votre démon de ligne d'imprimante.

Vous aurez besoin d'éditer le fichier `/usr/local/atalk/etc/papd.conf` pour configurer le démon. La syntaxe est la même que votre fichier habituel `/etc/printcap` file. Le nom que vous donnez à la définition est enregistré sous le nom de protocole Appletalk, NBP.

Un exemple de fichier de configuration:

```
TricWriter:\
   :pr=lp:op=cg:
```

Qui rendra votre imprimante nommée 'TricWriter' disponible pour votre réseau Appletalk et acceptera les travaux qui seront imprimés avec l'imprimante linux 'lp' en utilisant *lpd*.

13.11.6 Démarrer avec le logiciel appletalk.

D'accord, vous devriez maintenant être prêtes pour essayer le configuration de base. Il y a un fichier *rc* fourni avec l'ensemble *netatalk* qui devrait vous convenir, aussi vous devriez faire:

```
# /usr/local/atalk/etc/rc.atalk
```

et tout devrait démarrer et tourner convenablement. Vous ne devriez pas voir de messages d'erreur et le programme devrait envoyer des messages sur la console explicitant chaque étape qui démarre.

13.11.7 Tester le logiciel appletalk.

Pour tester si le logiciel fonctionne correctement, sur une de vos machines Apple, faites déroulez le menu Apple, choisissez Chooser, cliquez sur Appleshare, et votre boîte Linux devrait apparaître.

13.11.8 Avertissements concernant le logiciel appletalk.

- Vous pouvez avoir besoin de démarrer le support Appletalk avant de configurer votre réseau IP. Si vous avez des problèmes en démarrant les programmes Appletalk, ou si après avoir démarré vous avez des ennuis avec votre réseau IP, alors essayez de démarrer le logiciel Appletalk avant de mettre en route le fichier `/etc/rc.d/rc.inet1` file.
- Le démon *afpd* (Apple Filing Protocol Daemon) houspille sévèrement le disque dur. Derrière les points de montage il crée deux répertoires: `AppleDesktop` and `Network Trash Folder`. Puis, pour chaque répertoire auquel vous accédez il y crée un répertoire `.Appledouble` qui lui permet de stocker les ressources, etc...Aussi réfléchissez bien avant d'exporter `/`, car vous aurez ensuite un bon moment pour faire le nettoyage.
- Le programme *afpd* attend des Macs des mots de passe en clair. Ceci peut engendrer des problèmes de sécurité, auusi soyez attentifs quand vous mettez en marche ce démon sur un machine connectée à Internet. Vous devez vous en prendre à vous-même si quelqu'un de mal intentionné effectue des choses indélicates..
- Les outils existant de diagnostic tels que *netstat* et *ifconfig* ne supportent pas Appletalk. Les informations brutes sont disponibles dans le répertoire `/proc/net/` si vous en avez besoin.

13.12 IP_MASQUERADE - permet à un certain nombre d'hôtes de prétendre n'être qu'un seul.

Si vous êtes une des nombreuses personnes qui possède une sorte de réseau local chez lui, ou un certain nombre de machines connectées entr'elles via une liaison SLIP/plip/ppp, ou bien encore des machines reliées à un réseau privé ethernet, et qui ont en plus une possiblité de connexion à Internet, vous avez probablement à un moment ou à un autre voulu que les machines non directement connectées soient en mesure de pouvoir communiquer avec Internet. Ceci est normalement accompli en obtenant une adresse IP valide pour votre réseau local puis en demandant à votre Fournisseur de Service Internet de vous fournir gentiment une route réseau pour votre connexion plutôt qu'une route hôte unique qu'il fournissait précédemmment. Malheureusement beaucoup de Fournisseurs vous demanderont beaucoup d'argent pour pouvoir utiliser au mieux la bande passante de votre connexion et sûrement cette idée ne vous enchantera pas.

IP_MASQUERADING apporte une solution intelligente à ce problème en faisant en sorte que toutes les machines de votre propre réseau apparaissent comme une seule machine très active sur le réseau. Il le fait en réalisant en temps réel, à la volée, des transpositions d'adresses. La plupart du code provient de Pauline Middelink <`middelin@polyware.iaf.nl`>.

Ken Eaves <`keves@eves.com`> a fourni la plupart des informations ci-dessous, si vous y trouvez l'aide nécessaire, remerciez Ken.

13.12.1 Obtenir le logiciel 'masq'.

Le logiciel IP_Masquerade est maintenant fourni en tant que patch avec les versions noyaux de linux 1.2.* et une nouvelle version de la commande *ipfw*. Le programme *ipfw* qui vient avec l'ensemble logiciel `net-tools` pour le configurer n'a pas encore l'option 'masquerade' incluse. Le noyau que vous utilisez doir supporter l'option `ip_firewalling` pour permettre à ip_masquerade de fonctionner.

Le fichier patch peut être obtenu sur:

ftp.eves.com

```
/pub/masq
```

Dans ce répertoire vous trouverez les fichiers de patch, les nouvelles versions de la commande *ipfw* et le FAQ original de Masquerade élaboré par Ken's dont nous nous sommes servis.

Le fichier patch s'appelle `masq-patch.1.2.n`, et vous devez le récupérer. Les fichiers `ipfw` et `ipfw-for-1.3.10` sont prévus respectivement pour les noyaux 1.2.* et 1.3.10.

13.12.2 Comment installer et compiler le patch?

Partant du principe que vous avez les sources d'un noyau 1.2.* installées sous le répertoire `/usr/src/linux` vous devrez faire comme suit:

```
#
# cd /usr/src
# patch -p0 <../masq-patch.1.2.n 2>patch.errs
- cherchez dans le fichier the patch.errs le chaine 'fail'. Si vous ne la trouvez pas alors tout s'est bien passe. Si vous l'avez trouvee verifiez bien que vous avez la bonne version des sources noyau.
# cd /usr/src/linux
# make config
- Vous devez repondre 'y' a:
  ...
  IP: forwarding/gatewaying (CONFIG_IP_FORWARD) [y]
  IP: firewalling (CONFIG_IP_FIREWALL) [y]
  IP: masquerading (ALPHA) [y]
  ...
```

```
- Vous repondez comme vous le faites d'habitude aux autres options.
# make dep
# make clean
- Compilez comme d'habitude, c'est a dire:
# make zlilo
#
```

A la fin de la compilation, après avoir vérifier que votre noyau tourne correctement, assurez vous d'avoir une copie qui marche du programme *ipfw*. Celui est normalement installé sous `sbin`, aussi après avoir testé votre version contenant IP_MASQUERADE vous le mettez bien dans le répertoire `/sbin`. C'est tout ce qui est nécessaire pour avoir le support noyau installé.

13.12.3 Comment configurer le logiciel IP_MASQUERADE?

Configurer la fonction IP_MASQUERADE dépend complètement de la manière dont votre réseau est configuré. L'idée générale est la suivante: Tout ce qui est sur votre machine locale et que vous voulez rendre accessible à Internet doit avoir sa route par défaut pointée sur votre machine linux supportant IP_MASQUERADE. Deuxièmement, vous devez avoir une entrée dans la table firewall mentionnant que vos machines locales doivent être sous IP_MASQUERADE.

Pour configurer la fonction IP-MASQUERADE vous devez utiliser la commande *ipfw*. La commande *ipfw* est normalement utilisée pour configurer l'installation de la protection firewall Linux, et comme votre hôte ayant IP_MASQUERADING agit comme un protecteur contre les sorts, *ipfw* a été modifié pour prendre en compte cette fonction additionnelle.

Voici deux exemples de configurations qui couvrent la plupart des situations, et si vous avez quelque chose de plus compliqué que celles-ci vous avez probablement assez d'expérience pour vous débrouiller vous-même.

Deux machines reliées entr'elles par SLIP ou PPP:

```
 .-------.             .-------.
 |       |    SLIP     | linux |   SLIP/ppp
 | local |-----//------| masq  |-----//-------> Vers fournisseur Internet
 |       |             | svr   |
 .-------.             .-------.

         ^            ^         ^
         |            |         |
    192.168.0.2   192.168.0.1  assigne dynamiquement
```

Votre fichier /etc/rc.d/rc.inet1 devrait contenir en plus:

```
 ...
 ...
# Permet a la machine 192.168.0.2 d'etre avec IP_MASQUERADE
# pour tout le monde.
ipfw a m all from 192.168.0.2/32 to 0.0.0.0/0
```

```
...
...
```

Notez la syntaxe `iii.iii.iii.iii/nn`. Vous pouvez vous demander ce que signifie `/nn` et comment le calculer. Cette syntaxe est um moyen d'encoder à la fois une adresse ip et un masque de réseau. Ce que cette syntaxe dit est que ce sont toutes les adresses dont les premiers `nn` bits correspondent à ceux de `iii.iii.iii.iii`. L'exemple ci-dessus est pour un hôte unique, donc tous les bits doivent correspondre car il y a 32 bits dans une adresse IP et la syntax est `/32`. Un réseau privé ethernet de machines partageant une liaison unique internet:

```
.-------.    _
|       | .2 |
| local |----|
|       |    | <-- Ethernet lan
._______.    |     192.168.0.0 netmask 255.255.255.0
             |
             |
.-------.    |        .-------.
|       | .3 |     .1 | linux |   SLIP/ppp
| local |----|--------| masq  |-----//-------> Vers le fournisseur
|       |    |        | svr   |
._______.    -        ._______.

                               ^
                               |
                             assignee dynamiquement
```

Pour cette configuration vous avez une seule entrée qui décrit le réseau en entier, vous devriez avoir quelque chose comme cela:

```
 ...
 ...
# Permet a toutes les machines sur le reseau 192.168.0.0
# d'etre avec IP_MASQUERADE pour tout le monde.
ipfw a m all from 192.168.0.2/24 to 0.0.0.0/0
 ...
 ...
```

Une fois de plus, notez la relation entre le masque de réseau (255.255.255.0) et la syntaxe de *ipfw*. Un réseau de classe C (255.255.255.0) a 24 bits dans son masque de réseau.

Pour vous rendre la tâche plus facile, voici un tableau de masques réseau usuels avec le nombre de bits correspondants:

```
netmask              ipfw
---------------      ----
255.0.0.0            /8
```

```
255.255.0.0.0       /16
255.255.255.0       /24
255.255.255.128     /25
255.255.255.192     /26
255.255.255.224     /27
255.255.255.240     /28
255.255.255.248     /29
255.255.255.255     /32
```

13.12.4 Où discute-t-on de IP_MASQUERADE?

Il y avait une liste mail à masq@eves.com. Malheureusement elle est fermée maintenant.

13.13 IPALIAS - adressage multiple sur une interface.

Si vous utilisez un serveur WWW et voulez supporter plusieurs serveurs virtuels ou votre réseau comprend plusieurs sous-réseaux sur le même câble, alors vous aurez besoin que votre machine réponde à plus d'une adresse par interface.

Le programme *ipalias* vous permet de le faire.

Robert Sanders <`rsanders@mindspring.com`> a écrit un support linux dans ce but. Il est fourni comme une ensemble de patches pour le noyau linux 1.2.1.

Vous pouvez l'obtenir sur:

ftp.procyon.com

```
/pub/linux/ipalias
```

Pour l'installer vous avez besoin des sources du noyau 1.2.1 puis:

```
# cd /usr/src/linux
# patch -p1 .../alias-patch-1.2.1-v1
# make config
# make dep; make clean; make zlilo
```

Vous aurez besoin aussi d'avoir le fichier `alias-net-tools.tar.gz` car il contient les nouvelles versions des commandes *ifconfig* et *netstat*.

13.13.1 Configurer ip aliases.

Pour configurer ip alias utilisez la nouvelle commande *ifconfig* comme suit:

```
# ifconfig eth0 alias 198.71.200.5
```

Ceci suppose que vous avez déjà votre système configuré pour travailler sur votre réseau et que vous voulez que `198.71.200.5` soit une seconde adresse.

14 Outils de diagnostic - Comment trouver ce qui ne va pas?

Dans cette section, je décris brièvement certains des outils usuels de diagnostic que l'on trouve pour votre réseau Linux, et comment vous pouvez les utiliser pour identifier la cause de vos problèmes de réseaux, ou pour vous apprendre un peu plus comment fonctionne un réseau tcp/ip. J'ai passé sous silence quelques détails concernant la manère dont travaille les outils car ce document n'est pas l'endroit adéquat pour les décrire, mais j'espère présenter suffisamment d'informations pour que vous compreniez comment utiliser les outils et que vous compreniez mieux les pages *man* correspondantes, ou tout autre documentation.

14.1 ping - êtes vous là?

L'outil *ping* est compris dans la distribution NetKit-B comme indiqué plus haut dans la section 'Les applications réseau'. *ping*, comme son nom le laisse supposer, vous permet de transmettre un datagramme à un autre hôte qui vous le renverra si il est en activité et qu'il fonctionne correctement, et le réseau intermédiaire aussi. Dans sa forme la plus simple, vous faites:

```
# ping gw
PING gw.vk2ktj.ampr.org (44.136.8.97): 56 data bytes
64 bytes from 44.136.8.97: icmp_seq=0 ttl=254 time=35.9 ms
64 bytes from 44.136.8.97: icmp_seq=1 ttl=254 time=22.1 ms
64 bytes from 44.136.8.97: icmp_seq=2 ttl=254 time=26.0 ms
^C

--- gw.vk2ktj.ampr.org ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 22.1/28.0/35.9 ms
#
```

Ce que *ping* a fait est de résoudre le nom d'hôte en une adresse, et en utilisant le protocole *icmp* il a transmis un paquet *icmp echo request (demande d'écho icmp)* à l'hôte éloigné périodiquement. A chaque *echo request (demande d'écho)* que l'hôte éloigné reçoit un paquet *icmp echo reply (réponse en écho icmp)* vous est transmis en retour. Chaque ligne commençant par '64 bytes from ...' représente un *echo reply (réponse en écho)* reçu en réponse à un *echo request (demande d'écho)*. Chaque ligne donne l'adresse de l'hôte qui répond, le numéro de la séquence correspondante, le champ *time to live (temps maximum de vie)* et la totalité du tempsthe total *round trip time(temps pour faire l'aller et retour)* que cela a pris. Le *round trip time* est le temps compris entre l'envoi du paquet *echo request* et la réponse correspondante *echo reply* Cela peut être une mesure de la rapidité ou de la lenteur du réseau entre les deux machineCela peut être une mesure de la rapidité ou de la lenteur du réseau entre les deux machines.

Les deux dernières lignes vous donnent combien de paquets ont été envoyés, combien de bonnes réponses ont été reçues et le pourcentage de paquets qui ont été perdus. Le pourcentage de pertes est une mesure de la qualité de la connexion. Un pourcentage élevé de pertes indique des problèmes tels qu'un taux d'erreurs élevé quelque part sur la liaison entre les deux hôtes, une capacité épuisée sur un routeur ou une liaison ou un fort taux de collisions sur un réseau ethernet. Vous pouvez utiliser *ping* pour identifier où le problème se situe en effectuant des sessions *ping* sur chacun des points de la route du réseau.

14.2 traceroute - Comment y vais-je?

L'outil *traceroute* se trouve dans la distribution NetKit-A détaillée plus haut. *traceroute* a pour but initial de tester et de montrer le chemin que votre connexion réseau doit prendre pour aller vers un hôte. *traceroute* utilise aussi le protocole *icmp*, mais il utilise un truc intelligent pour obtenir chaque point le long du chemin et renvoie une réponse au fur et à mesure qu'il progresse. Ce truc consiste à manipuler manuellement le champ *time to live* des paquets transmis. Le champ *time to live* est un mécanisme qui s'assure que les paquets solitaires ne soient emprisonnés dans une boucle de routage. Chaque fois qu'un paquet passe par un routeur, il décrémente le champ *time to live* d'une unité. Si *time to live* atteint zéro alors ce routeur ou hôte envoie un message *icmp time to live expired* à l'émetteur du paquet pour lui signifier que ce paquet n'est plus valide. *traceroute* utilise ce mécanisme en envoyant une série de paquets *udp* ayant leur *time to live* commençant à 1 et qui s'incrémente à chaque étape. En enregistrant les adresses qui répondent *icmp time to live expired* lorsque les paquets expirent, il peut déterminer le trajet suivi pour arriver à la destination. Un exemple d'utilisation ressemble à ceci:

```
# traceroute minnie.vk1xwt.ampr.org
traceroute to minnie.vk1xwt (44.136.7.129), 30 hops max, 40 byte packets
 1  gw (44.136.8.97)  51.618 ms  30.431 ms  34.396 ms
 2  gw.uts (44.136.8.68) 2017.322 ms  2060.121 ms 1997.793 ms
 3  minnie.vk1xwt (44.136.7.129) 2205.335 ms  2319.728 ms  2279.643 ms
#
```

La première colonne nous dit combien de bonds plus loin (quelle était la valeur de *ttl*), la seconde colonne quel est le nom de l'hôte et son adresse qui a répondu , si il peut être résolu ou seulement son adresse si ce n'est pas le cas. Les troisièmes, quatrièmes et cinquièmes colonnes sont les temps d'aller et retour pour trois paquets consécutifs vers ce point. Ceci nous dit que le premier bond le long de la route est via gw.vk2ktj et que les trois chiffres suivants sont les allers et retours vers ce routeur. Le bond suivant est via gw.uts.ampr.org et minnie.vk1xwt.apr.org est un bond plus loin. Vous pouvez en déduire des informations sur le chemin réseau en regardant les différences de temps entre chaque étape sur la route. Vous pouvez voir que les allers et retours pour gw sont relativement rapides, c'est un hôte connecté par ethernet. gw.uts est substantiellement plus lent à obtenir que gw, c'est par le biais d'une liason radio de faible vitesse, puis vous avez le temps ethernet et le temps radio ajoutés ensemble. minnie.vk1xwt est un peu plus lent que gw.uts, étant reliés par un réseau à grande vitesse.

Si vous effectuer traceroute et que vous voyez la chaîne `!N` apparaitre aprè le chiffre donnant le temps, cela indique que votre programme traceroute a reçu une réponse *network unreachable (réseau inatteignable)*. Ce message vous indique que l'hôte ou le routeur qui vous a envoyé cd message ne connait par le chemin vers l'adresse de destination. Cela indique habituellement qu'il y a un lien réseau cassé quelque part. La dernière adresse que vous avez obtenue est celle juste avant la liaison en défaut.

De manière similaire si vous voyez la chaîne `!H` cela indique qu'un message *host unreachable (hôte non atteignable)* a été reçu. Cela peut vouloir dire que vous allez jusqu'à la carte ethernet où l'hôte éloigné est connecté, mais l'hôte lui-même ne répond pas ou est en défaut.

14.3 L'utilitaire tcpdump

Adam Caldwell <acaldwel@103mort2.cs.ohiou.edu> a réalisé le portage de *tcpdump* sous linux. *tcpdump* vous permet de tracer les activités réseau en interceptant les paquets entrant et sortant. C'est très utile pour diagnostiquer des problèmes réseau ardus.

Vous pouvez trouver les sources et les binaires à : *103mor2.cs.ohiou.edu* (ftp://ftp.funet.fi/pub/OS/Linux/PEOPLE/Linus/net-source/tools/tcpdump-3.0.3-tar.gz)

tcpdump décode chacun des paquets qu'il intercepte et les affiche sous un format texte légèrement crypté. Vous devriez utiliser *tcpdump* si vous essayez de diagnostiquer un problème comme des erreurs de protocole, ou des déconnexions bizarres, ceci car il vous permet de voir en réel ce qui arrive sur le réseau. Pour bien utiliser *tcpdump* vous devez avoir quelques connaissances sur les protocoles et comment ils fonctionnent, mais il est aussi utile pour quelques services simples comme vérifier que les paquets quittent bien votre machine par le bon port si vous essayez de diagnostiquer des problèmes de routage et pour voir si vous recevez des paquets en provenance de destinations éloignées.

Un exemple de ce que répond tcpdump ressemble à ceci:

```
# tcpdump -i eth0
tcpdump: listening on eth0
13:51:36.168219 arp who-has gw.vk2ktj.ampr.org tell albert.vk2ktj.ampr.org
13:51:36.193830 arp reply gw.vk2ktj.ampr.org is-at 2:60:8c:9c:ec:d4
13:51:37.373561 albert.vk2ktj.ampr.org > gw.vk2ktj.ampr.org: icmp: echo request
13:51:37.388036 gw.vk2ktj.ampr.org > albert.vk2ktj.ampr.org: icmp: echo reply
13:51:38.383578 albert.vk2ktj.ampr.org > gw.vk2ktj.ampr.org: icmp: echo request
13:51:38.400592 gw.vk2ktj.ampr.org > albert.vk2ktj.ampr.org: icmp: echo reply
13:51:49.303196 albert.vk2ktj.ampr.org.1104 > gw.vk2ktj.ampr.org.telnet: S 700506986:700506986(0) win 512 <mss 1436>
13:51:49.363933 albert.vk2ktj.ampr.org.1104 > gw.vk2ktj.ampr.org.telnet: . ack 1103372289 win 14261
13:51:49.367328 gw.vk2ktj.ampr.org.telnet > albert.vk2ktj.ampr.org.1104: S 1103372288:1103372288(0) ack 700506987 win 2048 <mss 432>
13:51:49.391800 albert.vk2ktj.ampr.org.1104 > gw.vk2ktj.ampr.org.telnet: . ack 134 win 14198
13:51:49.394524 gw.vk2ktj.ampr.org.telnet > albert.vk2ktj.ampr.org.1104: P 1:134(133) ack 1 win 2048
13:51:49.524930 albert.vk2ktj.ampr.org.1104 > gw.vk2ktj.ampr.org.telnet: P 1:28(27) ack 134 win 14335

 ..
#
```

Quand vous démarrez *tcpdump* sans arguments il scrute le premier (avec le plus petit chiffre) système réseau qui n'est pas le système loopback. Vous pouvez spécifier quel système vérifier avec un argument comme montré ci-dessus. *tcpdump* décode alors chaque paquet transmis ou reçu et les affiche, ligne par ligne, sous forme de texte. La première colonne est évidemment l'heure à laquelle le paquet a été transmis ou reçu. Le reste de la ligne dépend du type de paquet. Les deux premières lignes de l'exemple montrent à quoi ressemble une requête arp de albert.vk2ktj pour gw.vk2ktj. Les quatre lignes suivantes sont deux pings de albert.vk2ktj pour gw.vk2ktj, notez que *tcpdump* vous donne en fait le nom du paquet *icmp* transmis ou reçu. Le symbole 'plus grand que' (>) indique dans quel sens le paquet a été transmis, c'est à dire de qui vers qui. Il indique de l'émetteur vers le récepteur. Le reste de l'exemple montre l'établissement d'une connexion telnet depuis albert.vk2ktj vers gw.vk2ktj.

Le nombre ou nom à la fin de chaque nom d'hôte indique quel numéro de socket est utilisé. *tcpdump* regarde dans votre fichier /etc/services pour effectuer la traduction.

tcpdump éclate chacun des champs et vous pouvez voir les valeurs des paramètres window et mss de quelques uns des paquets.

La page *man* documente toutes les options disponibles.

Note pour les utilisateurs de PPP: La version de *tcpdump* qui est actuellement disponible ne supporte pas la suite PPP des protocoles. Al Longyear a écrit des patches pour corriger ceci, mais ils ne sont pas encore inclus dans une distribution *tcpdump*. Les fichiers patch sont sités dans le même répertoire que l'ensemble *tcpdump* sur `sunsite.unc.edu`.

14.4 icmpinfo - exploite les messages icmp reçus.

ICMP (Internet Control Message Protocol) donne des informations utiles sur la santé de votre réseau IP. Souvent les messages ICMP sont reçus et agissent en silence, si bien que vous ne percevez pas leur présence. *icmpinfo* est un outil qui vous permettra de voir les messages ICMP beaucoup mieux que le fait tcpdump. Laurent Demailly <`dl@hplyot.obspm.fr`> prit les sources bsd de ping et les modifia profondément.

La version 1.10 est disponible:

hplyot.obspm.fr

```
/net/icmpinfo-1.10.tar.gz
```

La compilation est aussi simple que çà:

```
# cd /usr/src
# cd icmpinfo-1.10
# gzip -dc icmpinfo-1.10.tar.gz | tar xvf -
# make
```

Vous devez être root pour activer *icmpinfo*. *icmpinfo* peut soit envoyer les signaux décodés sur le terminal qui l'a appelé soit les envoyer dans le fichier *syslog*.

Pour vérifier son fonctionnement, démarrez *icmpinfo* puis faites *traceroute* vers un hôte éloigné. Vous verrez les messages icmp que *traceroute* utilise sur la console.

15 Quelques questions fréquemment posées et leurs brèves réponses.

Vous trouverez dans la suite des questions fréquemment posées et leurs réponses.

15.1 Questions d'ordre général:

J'ai seulement un accès en mode terminal à une machine sur le réseau,puis-je l'utiliser comme connexion réseau?

Oui vous pouvez, jetez un coup d'oeil à TERM. TERM vous permet de transformer votre liaison en mode terminal en liaison réseau. Cela nécessite quelques modifications de vos applications réseau, mais la plupart des binaires et des sources sont déjà disponibles pour la majorité des besoins. Regardez le lien *TERM-HOWTO* (`http://sunsite.unc.edu/mdw/HOWTO/Term-HOWTO.html`) pour plus de détails.

Quand je fais telnet/ftp/rlogin vers ma machine, ça dure une éternité, pourquoi?

Vous avez votre résolveur de nom qui n'est pas configuré correctement. Relisez la section concernant `/etc/resolv.conf`. Vous devez avoir au moins une entrée de nom de serveur configuré.

Je veux mettre en place mon propre réseau autonome, quelles adresses dois-je utiliser ?

RFC1597 spécifie quelques adresses IP réservées pour les réseaux privés. Vous devez les utiliser pour éviter d'avoir des problèmes si vous vous retrouvez relié à l'Internet par accident. Ces adresses sont :

```
10.0.0.0        -   10.255.255.255
172.16.0.0      -   172.31.255.255
192.168.0.0     -   192.168.255.255
```

Note: les adresses réseau réservées sont de classes A, B et C, donc vous n'êtes pas restreint dans la conception de votre réseau. Comme vous n'allez pas etre relié à l'Internet, cela ne pose pas de problème que vous ayez la même adresse qu'un autre, du moment que vos adresses soient uniques dans votre réseau.

Je voudrais que mes machines sur mon réseau local puisse utiliser ma connexion Internet, comment faire?

Vous avez trois possibiltés, qui sont:

- Demandez à votre fournisseur de service Internet de vous fournir une route réseau au lieu d'une route hôte. Ce sera cher mais c'est la meilleure option technique et performante.
- Réglez votre machine connectée à votre ligne téléphonique comme ++++'SOCKS' based firewall++++++. Cela fournit par là-même une fonction *procuration (proxy)* où le monde extérieur croit que c'est votre machine routeur qui parle alors qu'en fait elle fait au nom des machines de votre réseau. Vérifiez le *Firewall-HOWTO* (`http://sunsite.unc.edu/mdw/HOWTO/Firewall-HOWTO.html`) pour des informations détaillées sur la manière de faire. Ceci demande que les machines de votre réseau d'avoir des programmes clients qui ont les possiblités SOCKS.
- Configurer et utilisez le logiciel `IP-Masquerade`. C'est probablement le moyen le plus simple, car il est vraiment direct et ne requiert pas de clients spéciaux sur votre machine. Ce logiciel réalise intelligemment les traductions d'adresses qui font que les machines de votre réseau semblent supporter elles-mêmes vos connexions. Il y a quelques limitations pour que ça marche bien, mais c'est moins lourd que de tourner avec une passerelle 'full firewall'.

Si ftp.linux.org.uk est hors-service, comment récupérer les fichiers spécifiés?

Les archives d'Alan sont mirrorisées sur:

ftp.Uni-Mainz.DE

```
/pub/Linux/packages/Net2Debugged
```

et/ou **ftp.infomagic.com**

`/pub/mirrors/linux/sunacm`

Comment savoir quelle est la version du noyau qui tourne ?

Les versions du noyau et du code réseau sont maintenant synchronisées, donc faire :

`uname -a`

ou :

`cat /proc/version`

Comment changer le message que les utilisateurs de telnet reçoivent quand ils se connectent?

Le fichier `/etc/issue` est le message que les utilisateurs normaux de getty reçoivent à la connexion. Quelques programmes *telnetd* utilisent le fichier `/etc/issue.net` à la place. Ainsi si voyez qu'un changement dans votre fichier `issue` n'a pas d'effet, essayez l'autre.

15.2 Messages d'erreur:

Je continue à avoir l'erreur 'eth0: transmit timed out'. Qu'est-ce que cela signifie ?

Cela signifie généralement que votre câble ethernet est débranché ou que vos paramètres pour votre carte (adresse E/S, IRQ, etc.) ne sont pas corrects. Vérifiez les messages de démarrage pour voir si votre carte est reconnue avec la bonne adresse ethernet. Si tout est correct vérifiez que vous n'avez pas de conflit avec d'autres cartes, comme une carte soundblaster partageant la même IRQ ou le même port d'entrées/sorties.

J'ai des erreurs 'check Ethernet cable' quand j'utilise le réseau.

Vous avez probablement une carte ethernet mal configurée. Regardez à deux fois dans le fichier /usr/src/linux/drivers/net/CONFIG. Si tout semble correct, vous pouvez en fait avoir un problème de câblage, vérifiez que vos câbles sont bien branchés.

15.3 Questions de routage:

Pourquoi ai-je le message 'obsolete route request' quand j'utilise la commande route ?

Vous utilisez une version de route antérieure à celle de votre noyau. Vous devez mettre à jour votre version de route. Référez-vous à la section « Les outils de configuration du réseau » de ce document pour obtenir ces outils.

Pourquoi ai-je un message 'network unreachable' quand j'essaie une commande réseau ?

Ce message signifie que votre machine ou une autre ne sait pas comment atteindre une autre machine. Si vous avez ce message avec toutes les machines que vous essayez de joindre cela signifie que votre route par défaut est mal positionnée. Relisez la section 'Routage'.

Je peux faire un ping à mon routeur mais je ne peux pas en faire aux autres machines

C'est probablement dû à un problème de routage. Reliser la section 'Routage' de ce document. Si tout semble correct, assurez-vous que la machine à laquelle vous essayez de vous connecter possède une route vers vous. Si vous vous connectez en appelant par téléphone, une cause courante est que votre serveur ne fait pas fonctionner de programme comme *gated* ou *routed*, ni *'prox arping'* pour vous, sinon vous pourrez envoyer des paquets aux machines distantes mais elles ne pourront pas vous en envoyer.

15.4 Utiliser Linux avec des serveurs de fichiers/NFS:

Comment faire pour utiliser mon serveur de fichier Novell avec ma machine Linux ?

Si vous avez le démon NFS Novell alors c'est simple, monter le volume NFS que vous voulez utiliser. Sinon vous ne pourrez pas le faire directement, il vous faudra un PC libre. Vous pouvez lancer un programme appelé Stan's Own Server sur le PC libre. Ensuite configurer le PC en station de travail Novell avec des cartes pour les répertoires que vous voulez monter NFS, puis lancer SOS et exporter ces cartes de disques. SOS est disponible à spdcc.com:pub/sos/sossexe.zooa Autrement, si vous obtenez une copie de la distribution `Caldera Preview`, il y est inclus un programme client Netware avec plein de possiblités qui vous permettra d'attacher directement et d'organiser les volumes Novell dans votre sytème de fichiers Linux. Je n'ai pas d'expérience moi-même, mais on m'a assuré que ça marche très bien.

Des fichiers sont corrompus quand j'utilise NFS avec un réseau

Certains vendeurs (Sun principalement) vendent leurs machines avec NFS sans vérification des checksums UDP. Bien avec ethernet, suicidaire ailleurs. Les checksums UDP peuvent être activées sur la plupart des serveurs. Linux l'a activée par défaut à partir du pl13 - mais les deux parties doivent l'avoir mis en route...

Pourquoi tous mes fichiers NFS sont-ils en lecture seule ?

Le serveur NFS Linux est par défaut en lecture seule. RTFM :lisez les pages de man de 'exports' et nfsd. Avec des serveurs autres que Linux vous devrez aussi modifier le fichier /etc/exports.

15.5 Questions sur SLIP:

Que faire si je ne connais pas l'adresse IP de mon serveur SLIP ?

dip n'a pas vraiment besoin de connaître l'adresse de votre serveur pour fonctionner. L'option *remote* a été ajoutée pour que *dip* puisse automatiser les commandes *ifconfig* et *route* pour vous. Si vous ne connaissez pas ou ne pouvez pas trouver l'adresse de votre serveur SLIP, Peter D. Junger `Junger@samsara.law.cwru.edu` suggère d'utiliser sa propre adresse où le script dip demande l'adresse distante. C'est un petit truc qui fonctionne bien car l'adresse du serveur n'apparaît jamais vraiment dans les en-têtes SLIP.

dip fonctionne seulement pour root. Comment faire pour que d'autres puissent l'utiliser?

dip a besoin d'être *suid root* pour réaliser certaines tâches. Uri Blumenthal recommande ceci:

- Créez un nouveau groupe nommé *dip* dans votre fichier `/etc/group` et mettez-y chaque personne autorisée.

– En tant que root, faites:

```
# chown root.dip /usr/bin/dip
# chmod u=rx,g=x,o= /usr/bin/dip
# chmod u+s /usr/bin/dip
```

Les utilisateurs entrants ne pourront faire que ce qui est contenu dans le fichier `/etc/diphosts`.

J'obtiens '`DIP: tty: set_disc(1): Invalid argument`', pourquoi?

Cela suggère habituellement que votre noyau n'a pas été compilé avec le support *SLIP*. Vérifiez que `/proc/net/dev` contient bien des dispositifs appelés `sl0`, `sl1`, etc.. Cela peut signifier aussi que votre version de dip est très vieille. Vous devez alors prendre une version récente.

Quand je fais 'ping' sur un hôte j'obtiens '`wrong data byte #17...`, pourquoi?

Cela signifie en général que votre modem est configuré pour un contrôle de flux XON/XOFF. SLIP **doit** avoir une ligne 8 bits propre, vous ne pouvez donc utiliser le contrôle XON/XOFF. L'échange de données avec protocole matériel marche mieux, utilisez le.

Avec SLIP je peux réaliser un ping avec mon serveur, et d'autres machines,mais je ne peux pas faire de ftp ni telnet.

C'est un désagrément causé par l'utilisation de la compression des en-têtes entre votre machine et le serveur. Vérifier que les deux extrémités de la liaison sont d'accord sur l'utilisation ou non de la compression.

Comment raccrocher la ligne quand j'utilise SLIP ?

Si vous utilisez dip pour numéroter la ligne SLIP, faites juste 'dip -k'. Sinon essayez de tuer le processus dip qui tourne. Quand dip meurt il libère la ligne. Pour lui donner le plus de chance de bien se terminer, essayez la séquence suivante : 'kill <pid>', 'kill -hup <pid>', et puis 'kill -9 <pid>' si le processus ne veut toujours pas mourir. La même méthode peut être appliquée à tout processus unix que vous voulez tuer.

Je vois beaucoup d'erreurs 'overrun errors' sur mon port slip, pourquoi ?

Les outils réseau plus anciens rapportaient incorrectement le nombre de paquets compressés en les comptant comme paquets perdus. Ceci a été corrigé et ne devrait pas vous arriver si vous utilisez une version récente du noyau et des outils. Si cela continue à se produire, cela signifie probablement que votre machine n'arrive pas à suivre le flux de données arrivant. Si vous n'utilisez pas des UART 16550AFN, vous devriez les mettre à niveau. Les 16450 et les 8250 génèrent une interruption pour chaque caractère qu'ils reçoivent et dépendent donc beaucoup de la capacité du processeur à trouver le temps de lire le caractère avant qu'un autre arrive. Les 16500AFN ont un tampon de 16 caractères et ne génèrent une interruption que lorsque le tampon est quasiment plein, ou quand ils ont eu un caractère à attendre. Moins d'interruptions sont donc générées pour la même quantité d'information et donc moins de temps est dépensé à contrôler le port série. Si vous voulez utiliser plusieurs ports série vous devrez obligatoirement utiliser des UART 16550AFN de toute façon.

Puis-je utiliser deux interfaces SLIP ?

Oui. Si vous avez, par exemple, trois machines que vous voulez interconnecter. Vous pouvez utiliser deux interfaces SLIP sur une machine et connecter les deux autres machines à la première. Configurez simplement la seconde interface de la même manière que la première. NOTE : la seconde interface aura besoin d'une adresse IP différente de la première. Vous aurez aussi besoin de jouer un peu avec le routage, mais cela devrait fonctionner.

J'ai une carte d'entrées/sorties multiports,comment faire pour utiliser plus de 4 ports SLIP ?

Le noyau est configuré par défaut avec un maximum de 4 pilotes SLIP, ceci est configuré dans le fichier `/usr/src/linux/drivers/net/slip.h`. Pour l'augmenter, disons à `16`, mettez `16` pour `#define SL_NRUNIT` à la place du `4` qui doit y être. Vous aurez aussi besoin d'éditer le fichier `/usr/src/linux/drivers/net/Space.c` pour ajouter les entrées pour `sl4`, `sl5` etc. Vous pouvez copier les définitions existantes comme modèles. Vous devrez bien sûr recompiler le noyau pour que les changements soient effectifs.

15.6 Questions sur PPP.

Référez-vous à *PPP-HOWTO* (`http://sunsite.unc.edu/mdw/HOWTO/PPP-HOWTO.html`) pour une liste de questions-réponses sur PPP rassemblées par Al Longyear.

16 Guide rapide - Serveur SLIP ou PPP

Configurer votre machine linux comme serveur SLIP ou PPP est d'une simplicité désespérante. Le processus en est simple, mais il y a un nombre de différents aspects pour la configuration et comprendre comment chaque étape interfère avec les autres vous aidera à diagnostiquer les problèmes que vous rencontrerez. Voici les étapes que vous devez suivre pour configurer votre machine linux comme serveur SLIP ou PPP:

1. Assembler votre matériel. Evitez les conflits d'IRQ et de mémoire partagée. Testez chaque port série en connectant un terminal muet à chaque port et en utilisant un câble null modem et un programme de communication tel que *minicom* ou *seyon* pour leur parler. Soyez certains d'émettre et de recevoir des caractères. Si vous avez l'intention d'utiliser un grand nombre de ports série, alors essayez d'utiliser une carte série correcte et d'utiliser les UART `16550AFN`. Ceci vous facilitera le travail pour manipuler les interruptions générées par les ports série.

2. Construire le noyau, soyez certains qu'il y a possiblité de réseau, avec `IP Forwarding` si vous voulez que des utilisateurs veulent router via votre serveur et `SLIP` ou `PPP` configurés pour votre besoin. Soyez sûrs de configurer `CSLIP` si vous voulez l'utiliser. Vérifiez bien que vous avez `IP Forwarding` si ce n'est pas le cas, alors les utilsateurs entrants ne pourront que se connecter à votre machine.

3. Installer votre noyau. Testez le. Vérifiez le fichier `/proc/net/dev` et soyez certains que vous avez les dispositifs `sl*` `ppp*`. Si ce n'est pas le cas, alors vous avez probablement fait une erreur en configurant votre noyau, ou bien vous ne tournez pas réellement sous votre nouveau noyau. Vérifiez bien que vous avez utilisé lilo pour installer le nouveau noyau.

4. Configurer un *getty* sur le(s) port(s) série(s) que vous voulez utiliser pour les appels entrants. Référez vous à *Serial-HOWTO* (`http://sunsite.unc.edu/mdw/HOWTO/Serial-HOWTO.html`) pour avoir une description de ce qu'il faut faire. Pensez à configurer votre modem de telle sorte que la broche `DCD`

détecte la porteuse reçue, c'est ainsi que la plupart que les programmes *getty* détectent un appel, (`mgetty` fait exception et utilise le signal venant de `RING`, aussi soyez sûrs que votre modem est configuré pour le générer: `ATQOV1` est habituellement suffisant).

5. Testez *getty* pour être sûr qu'il fonctionne. C'est important de le faire avant de vous soucier de la configuration de SLIP ou de PPP . Essayez de vous connecter à votre système, vous devriez obtenir une invite `login:` et être capables de faire tourner un shell normalement. Vérifiez les caractères perdus ou bizarres qui peuvent indiquer un problè de contrôle de flux.
6. Décider comment votre serveur allouera les adresses. Si vous voulez que les utilisateurs obtiennent toujours la même adresse à chaque connexion, alors vous voulez un serveur `statique`, si vous voulez minimiser le nombre d'adresses utilisées et ne pas vous soucier de la manière dont les adresses seront allouées, alors vous voulez un serveur `dynamique`.
7. Décider comment vous allez construire votre serveur SLIP ou PPP, soit vous allez utiliser *sliplogin*, *dip* ou *dSLIP* pour SLIP par exemple. Si vous voulez utiliser l'ensemble *sliplogin* alors voyez la section "Serveur SLIP utilisant sliplogin". Pour utiliser *dip* voyez la section "Serveur SLIP utilisant DIP". Pour *dSLIP* voir "Serveur SLIP utilisant dSLIP". Pour PPP vous devrez utiliser le programme *pppd*, lisez donc *PPP-HOWTO* (`http://sunsite.unc.edu/mdw/HOWTO/PPP-HOWTO.html`) pour savoir comment le configurer. Regardez bien si vous avez le logiciel approprié, qu'il est d'une version récente et qu'il est compilé et installé si besoin est.
8. Si les adresses que vous allouez à vos utilisateurs SLIP font parties de votre réseau ethernet, lisez bien la section Procuration ARP et configurez une procuration arp pour chaque adresse. Vous pouvez faire cela dans les fichiers `/etc/sliploginet /etc/SLIPlogout` si vous utilisez l'ensemble *sliplogin*. Si vous configurez un serveur PPP, soyez sûrs d'utiliser d'utiliser l'option *pppd* `proxyarp`. Si les adresses que vous allouez sont sur un réseau séparé de votre réseau ethernet , vous avez besoin d'un démon de routage tel que *gated*. Lisez alors la section "gated".
9. Tester votre serveur SLIP/PPP.
10. Demandez à quelqu'un d'autre de tester votre serveur SLIP/PPP.

17 Bogues connus

Le code réseau de Linux est toujours en évolution. Il y a encore des problèmes mais les rapports d'anomalies deviennent de moins en moins fréquents. La page *Linux Networking News* (`http://iifeak.swan.ac.uk/NetNews.html`) est un document hypertexte WWW maintenu par Alan Cox qui contient des informations sur l'état du code réseau NET3. Pour avoir les dernières informations, lisez le fichier `/usr/src/linux/net/inet/README` qui accompagne les sources du noyau ou rejoignez le canal NET.

18 Droits d'auteurs

19 Divers et remerciements

Il y a tant de personnes qui ont contribué par des commentaires et des suggestions à cette mise à jour que je vous ai oublié **Merci**.

SVP, si vous avez des commentaires ou des suggestions, envoyez les moi par courrier email. Je suis bien occupé ces jours-ci, aussi je ne vous répondrai pas tout de suite, mais je suis sûr de prendre en considération toute suggestion de votre part.

Le code de réseau pour Linux a fait un long chemin et ça n'a pas été facile, mais les développeurs, tous, ont fait un excellent travail pour faire ensemble quelque chose de fonctionnel, souple d'utilisation, flexible et libre d'utilisation pour nous. Nous leur devons de grands remerciements. Linus, Ross, Fred, Alan, les testeurs Alpha/Beta, les développeurs d'outils et ceux qui ont offert un support moral ont tous contribué au code tel qu'il est aujourd'hui.

Pour ceux qui ont quelque chose qui les démange et qui veulent se gratter, bonne promenade, c'est là.

salutations `Terry Dawson, vk2ktj.`

<`94004531@postoffice.csu.edu.au`>, ou <`terry@perf.no.telecom.com.au`>