

# Sommaire

<b>Installation d'un scanner SCSI grâce à SANE.....</b>	<b>1</b>
Récupérer les sources du noyau.....	1
Pour les novices.....	1
Pour les plus expérimentés :.....	2
Récupérer les sources de SANE.....	2
Récupérer les sources du patch pour les Snapscan.....	2
Installation du noyau avec support des modules et du SCSI.....	3
Configuration des modules et des devices.....	5
Les modules.....	5
Les devices.....	6
Installation des bibliothèques SANE.....	6
Configuration des bibliothèques SANE.....	7
Utilisation de SANE.....	7
Configuration en réseau.....	8
Sur le poste serveur nommé : serveur.....	8
Sur le poste client nommé : client1.....	9
Les mots de la fin :.....	9
Quand allumer votre scanner ?.....	10
Le fichier dll.conf.....	10
<b>Copyright.....</b>	<b>11</b>

# Installation d'un scanner SCSI grâce à SANE

Installation d'un scanner SCSI  
grâce à SANE

par Fred et Christian Vivet.

Comme moi, vous laissez Win\$\$\$ encombrer votre disque pour pouvoir scanner de belles photos...  
Et bien ça ne sert à rien, Linux fait aussi bien et cela sans enrichir l'homme le plus riche du monde.

---

Les informations contenues dans cette page ne sont aucunement garanties. Je les ai mises à la disposition du plus grand nombre pour rendre service, pas pour lire des reproches. Néanmoins, j'apprécierai grandement toutes les critiques constructives, en particulier, celles liées à mon orthographe déplorable, à ma syntaxe difficile à suivre et aux erreurs qui se sont certainement glissées subrepticement au sein de ce texte. Si vous voyez quelque chose à ajouter, je vous serais reconnaissant de bien vouloir me mailer les modifications que vous apportez à ce document. En plein accord avec l'esprit de liberté qui préside aux développements de linux, faites ce que vous voulez de ce texte, sauf prétendre que c'est vous qui l'avez écrit.

*Ce document concerne l'installation d'un scanner et plus spécifiquement d'y—celui que j'ai chez moi : le Snapscan 310 SCSI. En fait, je pense que la procédure est globalement la même quelque soit le scanner, excepté l'application du patch spécifique au Snapscan (évidemment ;—). Ce document est écrit sous la forme d'une seule page, il est, de cette façon, plus facile de le télécharger.*

Remarque :

Les scanners USB commencent à être supportés également par Linux. Voir la [usbscan.php3 rubrique correspondante].

**Pour installer un scanner SCSI sous Linux vous aurez besoin de :**

- Un scanner SCSI compatible avec Sane 1.0.5, voir sur le site de SANE.
- Une carte SCSI compatible avec Linux (la plupart des cartes SCSI le sont...)
- Un PC (compatible Intel, je suppose)
- Une distribution basée sur les noyaux 2.x.x (elles le sont toutes...)
- Vos mains (pour faire les manipulations) et votre cerveau (pour comprendre ma prose)

*Note : l'ensemble de la procédure d'installation décrite se déroule avec les droits de root donc faites attention à ce que vous taperez. Certaines parties de la procédure ne nécessitent pas réellement ces droits, mais ce sera plus simple pour moi de décrire l'ensemble de la procédure avec ces droits—ci. Si le coeur vous en dit essayez vous—même de vous passer de ces droits, linux vous préviendra quand vous essayerez de les outrepasser.*

## Récupérer les sources du noyau

### Pour les novices

Votre distribution vous fournit forcément les sources du noyau. Pour les besoins de l'explication j'utiliserai dans ce document l'exemple de la Mandrake 6.0 ou plus (fonctionne sur une Mandrake 8.0).

**1. Il faut monter le CD de votre distribution.**

```
[root@becane home]# mount /mnt/cdrom
```

Vous pouvez/devez vous passez de cette étape si vous utilisez supermount.

## 2. Il faut installer les sources du noyau.

Avec la Mandrake 6.0 les sources du noyau portent le joli nom de :  
kernel-source-2.2.9-19mdk.i586.rpm. Pour les installer, tapez :

```
[root@becane home]# rpm -i /mnt/cdrom/Mandrake/RPMS/kernel-source-*.rpm
```

Si vous n'avez pas la Mandrake 6.0, mais une autre distribution basée sur le système de package RPM la procédure est identique. Il faut seulement changer le chemin d'accès aux packages et le nom du package (ce sera toujours quelque chose du type kernel-source\*, kernel-src\*).

## Pour les plus expérimentés :

Récupérez les derniers sources du noyau sur <ftp://ftp.fr.kernel.org/pub/linux/kernel/v2.4>. Ainsi, si des problèmes ont été réglés par les développeurs du noyau, vous bénéficierez de ces améliorations.

Vous les décompressez dans le répertoire /usr/src :

```
[root@becane root]# cd /usr/src
```

```
[root@becane src]# tar xzvf /ou/trouver/les/sources/kernel-xxx.tar.gz
```

## Récupérer les sources de SANE

Je ne décrirai rien de particulier, les sources de SANE sont disponibles sur le site de SANE. La version des bibliothèques SANE que j'ai utilisées est la 1.0.5 ; *chez moi* elles fonctionnent correctement avec la Mandrake 8.0.

Décompressez les sources :

```
[root@becane root]# cd /usr/src
```

```
[root@becane src]# tar xzvf /ou/trouver/les/sources/sane-backends-1.0.5.tar.gz
```

```
[root@becane src]# tar xzvf /ou/trouver/les/sources/sane-frontends-1.0.5.tar.gz
```

## Récupérer les sources du patch pour les Snapscan

Les sources du patch sont disponibles, sur le site du rédacteur du "backend" du Snapscan

### Plusieurs conseils :

1. Si vous n'avez pas de Snapscan : **Ne télécharger aucun des ces patches** : ces patches sont d'après le rédacteur en beta.
2. Si vous avez un Snapscan : **Télécharger l'un de ces patches !**

Bon maintenant vous avez le patch, décompressez-le :

```
[root@becane root]# cd /usr/src/sane-backends-1.0.5/backend
```

```
[root@becane backend]# tar xzvf /la/ou/est/snapscan-<version>.tar.gz
```

C'est bon tout est prêt ! On va maintenant configurer votre carte SCSI, [scanner.php3#sane\_install passez ce qui suit] si votre carte SCSI est correctement configurée avec le support des périphériques génériques (c'est certainement déjà le cas si vous avez un graveur de CD SCSI correctement configuré).

## Installation du noyau avec support des modules et du SCSI

Vérifions que le noyau que vous utilisez ne contient pas déjà tout ce qu'il faut :

```
[root@becane home]# ls /lib/modules/`uname -r`/scsi
53c7,8xx.o aha1542.o eata_pio.o initio.o qllogicfas.o sym53c8xx.o AM53C974.o aha1740.o fdomain.o ips.o
qllogicfc.o t128.o BusLogic.o aic7xxx.o g_NCR5380.o megaraid.o qllogicisp.o tmcsim.o NCR53c406a.o
atp870u.o gdth.o ncr53c8xx.o scsi_debug.o u14-34f.o a100u2w.o dtc.o ide-scsi.o pas16.o seagate.o
ultrastor.o advansys.o eata.o imm.o ppa.o sg.o wd7000.o aha152x.o eata_dma.o in2000.o psi240i.o
sym53c416.o
```

Si le module `sg.o` existe, après avoir trouvé dans `/usr/doc/HOWTO/BootPrompt-HOWTO` le nom du module que vous devez charger pour votre carte SCSI, vous pouvez passer à [scanner.php3#scsi\_config la section suivante].

Sinon faites ce qui suit :

Pensez à lire la [../kernel/kernel.php3 rubrique noyau].

A partir de maintenant vous devrez tout savoir sur votre carte SCSI : son *nom*, le *numéro de l'interruption* (IRQ) ainsi que la *plage d'adresse* (IO) qu'utilise votre carte SCSI. Toutes ces informations peuvent être obtenues :

- soit directement sur la carte
- soit sur le mode d'emploi de la carte
- soit, si vous utilisez Win\$\$\$, dans les informations système du panneau de configuration si vous avez réussi à la configurer (ce qui devrait être le cas).

*Pour certaines cartes, l'interruption et l'adresse sont détectées automatiquement ; si vous avez l'une d'entre elles, vous n'avez pas besoin de vous en soucier ; vous pouvez voir si c'est le cas dans /usr/doc/HOWTO/BootPrompt-HOWTO.*

1. On va compiler tout ce qui nous intéresse sous forme de modules. Pour cela, on se place dans le répertoire contenant les sources du noyau `/usr/src/linux` :  

```
[root@becane root]# cd /usr/src/linux
```
2. On configure le noyau :  

```
[root@becane linux]# make xconfig
```

A partir de là, nous utilisons un programme de configuration qui est plus "convivial" que la ligne de commande mais peut-être quand même un peu abscons. Je vais donc décrire tout ce qui concerne le SCSI car moi aussi j'ai galéré quand j'ai compilé pour la première fois un noyau Linux, mais pour ce qui concerne le reste de votre configuration je vous renvoie à la [../kernel/kernel.php3 rubrique noyau]. *Si vous avez déjà compilé votre noyau pour une autre raison, vous n'avez qu'à modifier votre configuration que sur ce qui suit.*
3. Activez le support des modules dans la rubrique : `<Loadable module support>`, vous activez (vous cochez les `<y>`) :
  - ◆ `<Enable module support>` (pour pouvoir utiliser les modules ;-)
  - ◆ `<Set version information on all symbols for modules>` (pour que le noyau y retrouve ses petits nous lui demandons de mettre des informations concernant le numéro de version du noyau utilisé pour la compilation des modules dans les modules)
  - ◆ `<Kernel module loader>` (pour que le noyau charge tout seul comme un grand les modules)
4. Dans la rubrique `SCSI support`, vous activez en tant que modules (les `<m>` doivent être cochés) :

## Hardware-hard\_imp-scanner

- ◆ SCSI support (évidemment)
  - ◆ SCSI disk support (seulement si vous avez des disques scsi, un zip, un jazz etc...)
  - ◆ SCSI tape support (si vous avez un lecteur de bande SCSI)
  - ◆ SCSI CD-ROM support (si vous avez un lecteur de cdrom ou un graveur SCSI)
  - ◆ SCSI generic (si vous avez un scanner ;- ) ou un graveur SCSI)
5. Dans la rubrique SCSI low-level driver vous activez le module correspondant à votre carte en cochant le <m> (LISEZ l'aide disponible par le bouton <Help> afin de voir de quel module vous avez besoin), et vous désactivez tous les autres modules en cochant les <n>.

Notez le nom du module correspondant à votre carte SCSI (***dans la rubrique*** <Help> *en face de chaque module, le nom est indiqué par :*

*The module will be called : XXXXXXXX.o*

*seul le XXXXXXXX nous intéresse, il sera désigné à partir de maintenant par VOTRE-MODULE).*

ATTENTION : les cartes AVA 1505 XXX sont reconnues par linux comme étant des cartes AHA152x (nom du module aha152x) ! Certaines sont Plug&Play et paradoxalement plus compliquées à paramétrer avec Linux (cf : [pnp.php3 plug&play]).

1. Vous configurez le reste de votre noyau et de vos périphériques (cf : [../kernel/kernel.php3 rubrique noyau]).
2. Il est conseillé de sauver votre configuration dans un fichier en cliquant avant sur <Store Configuration to File> (un bon endroit est /root/kernel-config).
3. Vous sauvez votre configuration de noyau en cliquant sur <Save And Exit>.
4. Vous compilez et installez le noyau et les modules :

*Pour faire un peu de ménage :*

```
[root@becane linux]# make clean
```

*Pour que linux y retrouve ses petits :*

```
[root@becane linux]# make dep Pour créer votre noyau :
```

```
[root@becane linux]# make zImage Pour créer vos modules :
```

```
[root@becane linux]# make modules Pour installer votre noyau (attention cela modifie votre répertoire /boot et lance lilo):
```

```
[root@becane linux]# make install Pour installer vos modules dans le répertoires ad hoc :
```

```
[root@becane linux]# make modules_install
```

1. A partir de maintenant vos modules sont prêts ! Mais non fonctionnels, il faut rebooter votre nouveau noyau et recenser les modules (c'est automatique) :

```
[root@becane linux]# reboot
```

Bon, effectivement il faut rebooter une fois, mais c'est tout, après ce ne sera plus la peine (sauf si vous ajoutez un nouveau périphérique...).

Tout ceci est inutile si vous utilisez la Mandrake ou la RedHat (et certainement la plupart des distributions) car les distributions modernes disposent déjà de tous les modules précompilés. Mais je vous conseille de compiler votre noyau au moins une fois en n'activant que ce que vous utilisez (mais tout ce que vous utilisez : n'oubliez pas les modules ppp – il m'est arrivé de galérer plusieurs heures pour me reconnecter à Internet parce que j'avais oublié ce \$\*?! de module –, les modules de gestions de l'imprimantes, etc...), de cette façon votre noyau sera adapté à votre configuration et pas à toutes les configurations possibles (ce dont vous devez vous moquez comme de mon premier win\$\$\$).

## Configuration des modules et des devices

### Les modules

Maintenant, il vous faut configurer les modules ! Donc, éditez le fichier `/etc/conf.modules`, ce fichier dit au chargeur de modules de quelle manière doivent être chargés ces modules ; pour gérer votre carte SCSI, il doit contenir (au moins) les lignes suivantes :

```
alias scsi_hostadapter VOTRE-MODULE
```

```
options VOTRE-MODULE VOTRE-MODULE=VOTRE-IO,VOTRE-IRQ
```

ATTENTION : je ne bégaie pas ! VOTRE-MODULE est répété deux fois ! Le premier indique à quel module ce qui suit s'adresse, le second indique quel paramètre passer au noyau lors du chargement de ce module.

Le VOTRE-MODULES est le nom du module que vous avez compilé (le XXXXXXXXX que je vous ai dit plus haut de noter). Pour ce qui concerne les options, je n'ai indiqué que celles des cartes compatible aha152x (c'est celle que j'utilise...).

Pour plus d'information sur les options des modules voir les HOWTO qui leur sont consacrés (par exemple : dans `/usr/doc/HOWTO/BootPrompt-HOWTO`, cherchez VOTRE-MODULE et lisez bien ce qui est écrit, normalement ce sont des options à passer au noyau quand le driver correspondant n'est pas compilé comme un module, mais il se trouve que ce sont les mêmes que celles à passer à `modprobe`. Pour certaines cartes il n'y a pas d'option : tant mieux c'est que le driver détecte tout tout seul, dans ce cas, seule la ligne "alias" est nécessaire).

Pour ma carte AVA 1505 qui utilise l'interruption 9 et l'adresse 0x340 (ou 340h d'après Win\$\$\$), j'ai inséré les lignes :

```
alias scsi_hostadapter aha152x options aha152x aha152x=0x340,9
```

Maintenant si tout a bien fonctionné vous devez pouvoir charger le module de votre carte SCSI, allumez votre scanner et tapez :

```
[root@becane linux]# modprobe VOTRE-MODULE
```

Si vous n'avez pas de message d'erreur, c'est que vous êtes sur la bonne voie, une petite vérification supplémentaire, pour voir ce que Linux trouve sur votre chaîne SCSI, tapez :

```
[root@becane linux] cat /proc/scsi/scsi
```

```
Attached devices:
```

```
Host: scsi0 Channel: 00 Id: 01 Lun: 00
```

```
Vendor: RICOH Model: MP6200S Rev: 2.20
```

```
Type: CD-ROM ANSI SCSI revision: 02
```

```
Host: scsi0 Channel: 00 Id: 04 Lun: 00
```

```
Vendor: AGFA Model: SNAPSCAN 310 Rev: 1.20
```

```
Type: Scanner ANSI SCSI revision: 02
```

```
Host: scsi0 Channel: 00 Id: 05 Lun: 00
```

```
Vendor: IOMEGA Model: ZIP 100 Rev: J.02
```

```
Type: Direct-Access ANSI SCSI revision: 02
```

Là, j'ai mis ce que donne le *scanne* de ma chaîne SCSI. La partie intéressante est celle qui indique que Linux a trouvé un scanner. Que d'efforts pour si peu de chose ! Mais maintenant que tout fonctionne, vous n'aurez plus jamais (à voir...) à taper ce genre de commandes. C'est tout l'intérêt de modifier le fichier

`/etc/conf.modules` : il indique au chargeur de modules les paramètres à passer à ceux-ci.

## Hardware-hard\_imp-scanner

Remarque : dans les informations qui sont retournées par `cat /proc/scsi/scsi` :

- Channel : 00, signifie que votre périphérique est branché sur la première carte SCSI (si vous êtes fortunés, cette information peut être intéressante)
- Id : 04, signifie que votre périphérique a le numéro 4 dans votre chaîne SCSI (normalement le périphérique doit l'indiquer)
- Lun : 00, indique sûrement quelque chose, mais quoi ?

## Les devices

L'accès à votre scanner se fera par l'intermédiaire de périphériques, aussi appelés *devices* (les "fichiers" spéciaux contenus dans `/dev` comme par exemple `/dev/hda1` qui permet d'accéder à la partition numéro 1 du disque maître sur la première nape IDE), mais pour cela il faut qu'ils existent ;-). Normalement, ce doit être déjà le cas, mais vérifions que vous disposez bien de ces devices : les fichiers `/dev/sg*` :

```
[root@becane /dev]# ls /dev/sg* -l
crw----- 1 root sys 21, 0 May 5 1998 /dev/sga
crw----- 1 fred sys 21, 1 May 5 1998 /dev/sgb
crw----- 1 root sys 21, 2 May 5 1998 /dev/sgc
crw----- 1 root sys 21, 3 May 5 1998 /dev/sgd
crw----- 1 root sys 21, 4 May 5 1998 /dev/sge
crw----- 1 root sys 21, 5 May 5 1998 /dev/sgf
crw----- 1 root sys 21, 6 May 5 1998 /dev/sgg
crw----- 1 root sys 21, 7 May 5 1998 /dev/sgh
Si ces devices n'existent pas il faut les créer !
```

```
[root@becane dev]# mknod /dev/sga c 21 0
[root@becane dev]# mknod /dev/sgb c 21 1
[root@becane dev]# mknod /dev/sgc c 21 2
[root@becane dev]# mknod /dev/sgd c 21 3
[root@becane dev]# mknod /dev/sge c 21 4
[root@becane dev]# mknod /dev/sgf c 21 5
[root@becane dev]# mknod /dev/sgg c 21 6
[root@becane dev]# mknod /dev/sgh c 21 7
```

Bon, ça devrait suffire. Évidemment, votre distribution devrait déjà avoir créé tous ces périphériques (en cas de problème : changez de distribution, c'est pour ça qu'il y en a plusieurs ! ;-)

Remarque : pour ceux que ça intéresse :

```
mknod /dev/sga c 21 0
```

crée un fichier spécial (un device) appelé `sga` dans le répertoire `/dev` (normalement tous les périphériques sont créés dans ce répertoire) dont l'accès se fera en mode caractère (c'est le paramètre `c`) dont le numéro de majeur (qui indique de quel type de périphérique il s'agit : ici un périphérique SCSI dont l'accès par Linux ne se fera pas au travers d'un driver spécifique) est 21 et le numéro de mineur (qui indique de quel périphérique de ce type il s'agit) est .

## Installation des bibliothèques SANE

Si vous n'avez jamais compilé un programme sous Linux, ne vous effrayez pas, vous allez voir que dans le monde GNU/autoconf tout est *simple* :

Rendez vous directement dans le répertoire où vous avez mis SANE et ne touchez pas 20.000 francs (pas encore) :

## Hardware-hard\_imp-scanner

```
[root@becane root]# cd /la/ou/vous/avez/mis/sane-backends-1.0.5/
```

Compilez et installez SANE dans le répertoire /usr/local (note : si gimp 1.1.x est installé, le programme configure croit qu'il s'agit de gimp 1.0.x et essaie de compiler sane avec le support de gimp : ce qui ne marche pas, le mode de fonctionnement de gimp avec les plugins ayant changé entre ces deux versions, il faut donc, le temps de la compilation, déplacer le répertoire /usr/include/libgimp, merci à Yves Chaufour) :

```
[root@becane sane-backends-1.0.5]# ./configure --prefix=/usr/local ; make ; make install
```

```
[root@becane root]# cd /la/ou/vous/avez/mis/sane-frontends-1.0.5/
```

```
[root@becane sane-frontends-1.0.5]# ./configure --prefix=/usr/local ; make ; make install
```

Trouvez où est connecté votre scanner :

```
[root@becane sane-backends-1.0.5]# tools/sane-find-scanner
```

```
find-scanner: found scanner "AGFA SNAPSCAN 310 1.20" at device /dev/XXXXXX
```

Pour permettre à SANE de trouver tout de suite votre scanner, créez un lien vers ce périphérique :

```
[root@becane sane-backends-1.0.5]# ln -sf /dev/XXXXXX /dev/scanner
```

C'est fait !

## Configuration des bibliothèques SANE

Normalement, SANE doit maintenant fonctionner sans configuration particulière. Pour le vérifier, il suffit de lui demander d'afficher la liste des scanners qu'il trouve sur votre chaîne SCSI :

```
[root@becane home]# scanimage -L
```

```
device `snapscan:/dev/scanner' is a AGFA SNAPSCAN 310 flatbed scanner
```

Si SANE trouve plusieurs fois votre scanner sur plusieurs périphériques différents, ce n'est pas que Linux dispose de la faculté de multiplier les scanners comme d'autre multiplient les pains : c'est qu'il existe plusieurs périphériques qui ont le même numéro de mineur et de majeur, ou que plusieurs liens ont été créés vers le même périphérique. De toute façon ça n'empêchera pas SANE de fonctionner correctement. Au pire il vous demandera quel scanner vous voulez utiliser.

Remarque : les fichiers de configuration de SANE se trouvent (si vous avez suivi la procédure proposée) en /usr/local/etc/sane.d. Le fichier essentiel est /usr/local/etc/sane.d/dll.conf qui indique à SANE quels sont les scanners à rechercher. Vous pouvez commenter les scanners que vous savez ne pas avoir. Suivant le nom de votre scanner un autre fichier est important : /usr/local/etc/sane.d/VOTRE-SCANNER.conf : il indique à SANE où est potentiellement connecté votre scanner. Il est déconseillé (sauf si vous aimez vous em.d.r) d'enlever l'entrée : /dev/scanner car elle permet de changer rapidement de scanner actif (au cas où vous en avez plusieurs) en modifiant seulement le lien /dev/scanner, par contre vous pouvez enlever toutes les autres (si vous avez créé le lien /dev/scanner).

## Utilisation de SANE

L'utilisation de SANE est très simple. La distribution standard de SANE comporte deux programmes permettant d'utiliser votre scanner : **scanimage** et son homologue sous X : **xscanimage**.

Pour ce qui est de scanimage, je vous renvoie à la man page scanimage (1).

Pour utiliser xscanimage tapez :

```
[root@becane root]# xscanimage
```

Une fenêtre apparaît, vous permettant de saisir ou de choisir le fichier de sortie. Le réglage des paramètres est suffisamment simple pour que l'on ne s'y attarde pas. Sachez que les paramètres disponibles dépendent de votre scanner. Vous pouvez choisir la partie de l'image à scanner grâce à la fenêtre de prévisualisation.

## Hardware-hard\_imp-scanner

On peut configurer `xscanimage` en tant que plugin pour **Gimp** (pour cela il faut que, **avant la compilation de SANE**, vous ayez installé les fichiers d'include de `gimp` dans `/usr/include` (ou tout autre répertoire qui sera accessible par votre compilateur sans directive particulière, en effet il y a un *bug* dans le programme de configuration de `sane` qui lui empêche – sur ma machine – de trouver ces includes s'ils sont ailleurs, avec la vérité par exemple), ce qui n'est pas automatique. Pour remédier à cela, faites :

```
[root@becane home]# rpm -i gimp-devel.xxx.rpm
```

Puis, il suffit de créer un lien entre `xscanimage` et le répertoire de plugins de `gimp` :

```
[root@becane root]# ln -s `which xscanimage` ~/.gimp/plug-ins
```

Et c'est tout, vous relancez `gimp` et un menu proposant d'acquiescer une image apparaît dans `Xtms` ! En fait ce lien n'est pas celui qu'il faudrait créer si vous voulez que tous les utilisateurs puissent utiliser votre scanner :

```
[root@becane root]# ln -s `which xscanimage` /usr/share/gimp/plug-ins
```

ou quelque chose d'approchant serait mieux venu.

Il existe d'autres programmes pouvant utiliser SANE. Je n'en connais que deux : `xsane` et `ksane`.

J'ai testé **xsane**. Personnellement, je continue à utiliser `xscanimage` pour les scans simples et `xsane` pour le reste. Mais bon, tout cela reste affaire de goût, donc ne vous privez pas d'essayer les deux. Les plus de `xsane` :

- il peut être utilisé avec la branche de développement de `gimp`, donc si vous utilisez cette version de `gimp` : vous n'aurez pas le choix car `gimp 1.1.x` ne fonctionne pas avec `xscanimage`.
- il permet de faire des photocopies et d'envoyer des fax (si vous avez réussi à configurer `mgetty+sendfax`) en utilisant un unique programme.
- il permet de sauver directement l'image dans divers formats (et pas seulement `pnm`).

Quant à `ksane` je ne l'ai pas utilisé et je ne sais donc pas ce qu'il vaut. Le développement de ce programme semble être au point mort.

## Configuration en réseau

Nous devons cette partie à : Christian Vivet.

### Sur le poste serveur nommé : serveur

1. il faut configurer le fichier `/usr/local/etc/sane.d/saned.conf` et y ajouter le nom du client qui va utiliser le scanner en accès distant exemple de fichier `saned.conf` :

```
# saned.conf
#
client1
```
2. il faut ajouter une ligne dans le fichier `/etc/inetd.conf` pour démarrer le daemon `saned` :

```
sane stream tcp nowait root /usr/local/sbin/saned saned
```
3. il faut ajouter une ligne dans le fichier `/etc/services` en fin de fichier :

```
sane 6566/tcp # SANE en reseau serveur
```

## Sur le poste client nommé : client1

1. il faut configurer le fichier `/usr/local/etc/sane.d/net.conf` et ajouter le nom de la machine serveur du scanner : `serveur`.

Exemple de fichier `net.conf` :

```
# net.conf
```

```
#
```

```
serveur
```

2. il faut ajouter une ligne dans le fichier `/etc/services` du client (`client1` dans l'exemple) :

```
sane    6566/tcp          # SANE en reseau client
```

Et bien sur installer `sane`, `xsane` sur la machine serveur et sur la machine cliente puis les relancer. Voilà, ce n'est pas très compliqué à mettre en oeuvre et cela marche parfaitement.

## Les mots de la fin :

Maintenant, quelques *trucs*

Utilisation depuis un autre compte que celui de root Si vous voulez scanner depuis un autre compte que root, vous devez autoriser l'accès en lecture/écriture au périphérique `/dev/sgXXX` correspondant à votre scanner :

```
[root@becane root]# chmod a+rw /dev/sgXXX
```

pour que tout le monde puisse utiliser votre scanner, ou :

```
[root@becane root]# chown monuser:mongroup /dev/sgXXX
```

pour que seul monuser puisse utiliser votre scanner.

Enfin, pour ceux d'entre vous qui utilisent PAM (par exemple si vous utilisez la RedHat ou la Mandrake ou d'autres ?) vous pouvez aussi configurer le gestionnaire de sécurité PAM en modifiant le fichier `/etc/security/console.perms` afin de donner le périphérique `/dev/sgXXX` à l'utilisateur connecté à la console (vous par exemple). Il suffit pour cela, si votre scanner est connecté au périphérique `/dev/sgb` de rajouter à `/etc/security/console.perms` les lignes :

```
<scanner>=/dev/sgb
```

```
<console> 0600 <scanner> 0600 root
```

Ce qui suit `<scanner>=` est une expression régulière : vous pouvez mettre `/dev/sg[ab]` pour autoriser les accès à `/dev/sga` et `/dev/sgb` si vous avez plusieurs scanner. Mais faites attention : les `/dev/sg*` sont aussi les périphériques SCSI dans leur ensemble, donc si vous avez un disque SCSI ou un graveur SCSI l'un des `/dev/sg*` **est** celui-ci, vous devez absolument éviter de donner ce périphérique à l'utilisateur qui dispose de la console. Sinon vous aurez un gros trou dans la sécurité de votre système. (Pour plus de renseignements concernant ceci : voir la manpage `console.perms`) Ce paragraphe est en version alpha (je ne suis pas absolument sûr que les `/dev/sg*` soient effectivement tous les périphériques de votre chaîne SCSI, mais dans le doute ne t'abstiens pas ! Surtout que ça ne coûte pas cher de faire attention à la sécurité dans ce cas précis. Si quelqu'un en sait plus : qu'il m'écrive, ça m'intéresse.).

## Quand allumer votre scanner ?

Lorsque vous le souhaitez, pourvu que vous respectiez la règle suivante : le module correspondant à la carte SCSI sur laquelle est branché votre scanner, ne doit pas être chargé à ce moment là. Pour le vérifier, tapez :

```
[root@becane usr]# lsmod | grep VOTRE-MODULE
```

Si cette commande renvoie quelque chose, c'est que votre module est chargé : pour utiliser votre scanner, vous DEVEZ le décharger. Pour cela deux méthodes existent. La méthode Win\$\$\$ : on allume le scanner puis on reboot. La méthode Linux est moins bourrin : vous déchargez juste tous les modules qui utilisent votre carte scsi : `sd_mod.o`, `sg.o` et `sr_mod.o` :

```
[root@becane usr]# rmmod sr_mod
```

```
[root@becane usr]# rmmod sd_mod
```

```
[root@becane usr]# rmmod sg
```

```
[root@becane usr]# rmmod VOTRE-MODULE
```

Si vous obtenez un message d'erreur, c'est que soit ce module n'est pas chargé, soit l'un des modules est utilisé par un processus en cours de fonctionnement. Trouvez lequel, tuez-le, recommencez. Puis vous pouvez lancer `xscanimage` comme d'habitude. Il devrait trouver votre scanner (les modules nécessaires se chargeront alors automatiquement).

## Le fichier dll.conf

Le fichier `/usr/local/etc/sane.d/dll.conf` sert à dire à SANE quels sont les scanners dont il doit vérifier la présence dans votre chaîne SCSI. Vous pouvez commenter toutes les lignes qui ne correspondent pas à votre scanner (en les commençant par #).

Dans ce fichier, on trouve deux lignes qui ne correspondent pas réellement à des scanners : `net` et `pnm`.

- Le premier permet de chercher un scanner sur le réseau (ça dépasse le cadre de cet article : une autre possibilité de SANE : il permet de partager un scanner sur un réseau, pour savoir comment faire, la man page de `saned` est le point de départ obligé ! (Si quelqu'un a réussi à le faire fonctionner en réseau, un mail en expliquant le fonctionnement serait le bienvenu, car pour l'instant je n'ai pas eu le temps d'essayer)).
- Le second (`pnm`) est un backend pour SANE d'un genre un peu particulier, vous lui donnez un fichier source au format `pnm` et SANE fait comme si ce fichier était le résultat d'un scan. Cela permet de vérifier, lorsque quelque chose ne fonctionne pas, si c'est le scanner ou SANE qu'il faut mettre en cause. Si ce backend fonctionne, c'est que SANE est correctement configuré.

Quelques ressources concernant les scanners (mailez-moi pour que je rajoute vos liens) :

- Une page concernant l'installation d'un scanner AGFA SnapScan 6xx (idem 3xx). A utiliser si vous ne comprenez rien à ce que j'ai expliqué ;-). Il n'utilise pas le patch, certainement parce que son SnapScan n'est pas un 310, à vous de voir s'il est nécessaire d'appliquer le patch. Je serais content que vous me mailer vos expériences concernant ce patch (en particulier l'utilisation d'une nouvelle version).
- La page de SANE.
- La page de celui qui a écrit le backend pour le Snapscan.

Cette page est issue de la documentation 'pré-wiki' de Léa a été convertie avec HTML::WikiConverter. Elle fut créée par Frédéric Bonnaud le 19/11/1999.

# Copyright

Copyright © 19/11/1999, Fr