

COPET Mathieu  
TANI Raphaël



# **Etude du système FAI (Fully Automatic Installation)**



## Sommaire

Sommaire .....	2
Introduction .....	3
1. Contexte : Installer un cluster .....	3
1.1. Définition d'un cluster .....	3
1.2. Cluster Beowulf et OpenSSI .....	3
1.2.1. Cluster Beowulf .....	3
1.2.2. OpenSSI .....	4
2. Présentation générale et fonctionnement de FAI .....	5
2.1. Présentation générale .....	5
2.2. Fonctionnement de FAI .....	5
2.2.1. Les parties engagées dans FAI .....	5
2.2.2. Fonctionnement de FAI .....	6
3. Installation de FAI .....	7
4. Configuration .....	12
4.1. Notions .....	12
4.1.1. Arborescence du serveur FAI .....	12
4.1.2. Le concept de classes .....	13
4.2. Configuration des clients .....	13
5. Résultats .....	14
Conclusion .....	16
Bibliographie .....	16
Annexe 1 .....	17
Annexe 2 .....	21
Annexe 3 .....	23
Annexe 4 .....	24
Annexe 5 .....	25
Annexe 6 .....	26
Annexe 7 .....	27
Annexe 8 .....	28
Annexe 9 .....	30

## Introduction

Dans le cadre de notre formation, nous avons eu à réaliser un projet sous la tutelle de M. DEXHEIMER que nous tenons à remercier.

Ce projet consiste à étudier un outil d'installation automatique de système Debian. Cet outil a été développé par Thomas Lange de l'université de Cologne et s'appelle FAI (acronyme de Fully Automatic Installation).

Ce projet a été mené en collaboration avec les moyens informatiques du Loria pour les besoins d'installation de machines devant être intégrées dans un cluster.

Dans ce rapport, nous allons tout d'abord expliquer plus précisément le contexte des installations. Nous ferons ensuite une présentation de l'outil FAI en expliquant son fonctionnement. Nous verrons également comment installer FAI et configurer les installations. Enfin, nous finirons par expliquer les résultats obtenus.

## 1. Contexte : Installer un cluster

### 1.1. Définition d'un cluster

Un cluster est un ensemble de machines appelées noeuds devant mettre en commun leur ressource pour fournir des services. On distingue donc plusieurs types de cluster.

Cluster haute disponibilité

Cluster haute performance (type Beowulf)

Cluster de stockage

Cluster de base de données

Cluster de service Web

Les noeuds possèdent un système d'exploitation et une couche supplémentaire logicielle capable de gérer les différents aspects propres au cluster comme par exemple la répartition de charges, ou l'ajout d'un nouveau noeud. Un cluster satisfaisant doit être capable de proposer la disponibilité attendue, supporter le passage à l'échelle, être facilement administrable et utilisable.

### 1.2. Cluster Beowulf et OpenSSI

#### 1.2.1. Cluster Beowulf

Beowulf est une architecture multi-ordinateurs qui peut être utilisée pour la programmation parallèle. On dénombre deux types de noeuds : les noeuds serveurs et les noeuds clients. Les noeuds sont généralement liés entre eux grâce à une liaison Ethernet haut débit. Le noeud serveur contrôle l'ensemble du cluster et sert de serveur de fichiers. C'est également la passerelle vers l'extérieur. C'est pourquoi pour améliorer les performances, il existe des clusters Beowulf avec plusieurs noeuds serveurs, éventuellement dédiés. Dans la mesure où quelques machines voire une seulement contrôlent tout le cluster, on déduit que Beowulf est considéré comme une seule machine virtuelle de hautes performances.

### 1.2.1.1. Classification des clusters Beowulf

Il existe deux classes de cluster Beowulf. La classe du cluster dépend du test Computer Shopper.

Les clusters de classe 1 sont constitués de pièces qui peuvent être facilement trouvées. L'avantage de cette classe est le prix. En effet des pièces facilement disponibles sont souvent moins chères que des pièces particulières d'un constructeur particulier. Un autre avantage est qu'un tel cluster peut supporter des logiciels libres et gratuits. Cependant le matériel commun ne peut satisfaire totalement en termes de performance.

Les clusters de classe 2 sont quant à eux composés de pièces plus rares. Ils offrent souvent des performances supérieures, mais les composants sont dépendants d'un constructeur. Le prix est donc plus élevé et il se peut que le matériel disparaisse, ce qui peut poser des problèmes pour la maintenance.

Le choix d'une classe de cluster dépend surtout de l'application, mais aussi des liens entre les différents noeuds. FAI fournit les classes nécessaires pour construire un cluster avec trois machines (un noeud serveur et deux noeuds clients).

### 1.2.2. OpenSSI

OpenSSI est une couche logicielle capable de gérer un cluster (type Beowulf notamment). OpenSSI doit être intégré dans le noyau en re-compilant le noyau.

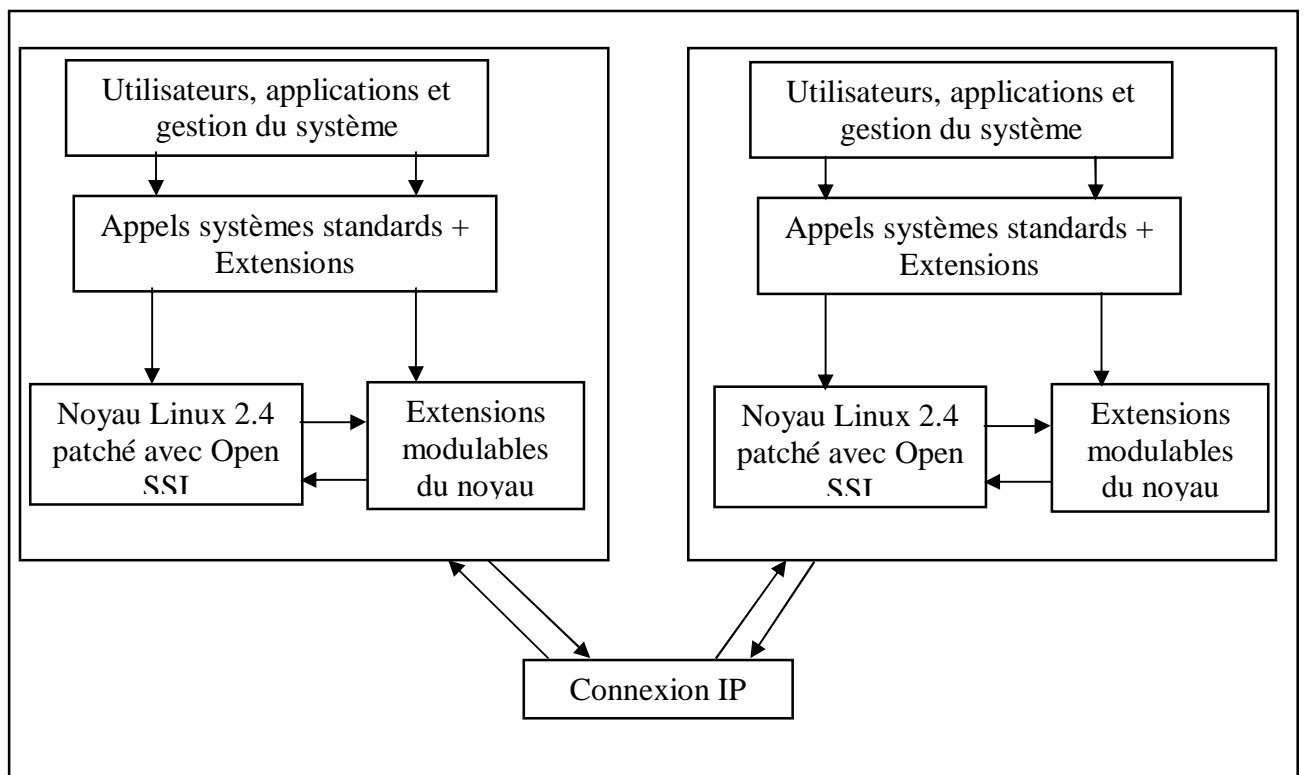


Figure 1. Extrait de la présentation Dr Bruce Walker sur la cluster OpenSSI Linux.

## Conclusion

Un cluster est donc composé d'un ensemble de machines. Les besoins des applications parallèles peuvent imposer un nombre important de machines (on peut trouver des clusters de plus de 1000 noeuds), or ces noeuds doivent être installés et configurés. Si les installations sont réalisées manuellement, elles peuvent prendre un temps important et mobiliser beaucoup de personnels. De plus les opérations répétitives de configuration peuvent induire des erreurs et poser des problèmes sur le fonctionnement global du cluster. C'est pourquoi il existe des outils d'installation et de configuration automatique, tel que FAI.

## 2. Présentation générale et fonctionnement de FAI.

### 2.1. Présentation générale

FAI est un outil non interactif d'installation et de configuration du système Debian. Il est non interactif, car il ne nécessite aucune surveillance pendant l'installation. Il permet de gérer l'installation et la configuration de machines.

Ce projet est né lorsque Thomas Lange eu besoin d'installer un cluster avec un noeud maître et 16 noeuds clients. Pour automatiser l'installation, il s'inspira de l'outil JumpStart sous Solaris, qui réalise des installations automatiques. Il adapta les scripts pour installer un système Debian de façon automatique. Comme FAI est inspiré de l'outil JumpStart, il existe une version de FAI capable d'installer le système Solaris. De plus, FAI est évolutif et il est possible que l'on puisse trouver dans le futur une version de FAI pour les distributions orientées paquets rpm.

### 2.2. Fonctionnement de FAI

#### 2.2.1. Les parties engagées dans FAI

Quatre parties sont importantes pour comprendre FAI. Tout d'abord, FAI possède un *serveur d'installation* qui est une machine avec le système Debian et les packages *fai* et *fai-kernels* installés. Nous verrons dans la partie sur l'installation de FAI comment installer et configurer l'outil.

Les clients doivent être capable de démarrer un système minimal, à partir d'une disquette ou à partir du réseau. Pendant ce projet, nous avons seulement utilisé la méthode de démarrage par le réseau.

Une partie *configuration* est également impérative. On peut y trouver l'ensemble des packages à installer, la configuration des clients. La syntaxe des fichiers correspondant à l'installation des packages est simple; il s'agit d'une liste; mais la configuration des clients doit être décrit dans des scripts écrits dans un langage particulier, comme perl, sh ou cfengine. Cette phase de configuration est importante, une phrase redondante dans la documentation de FAI est :

*Plan your installations and FAI installs your plans.*

Il est donc nécessaire de s'interroger sur la façon de configurer les clients.

La dernière partie est la racine NFS, il s'agit du système de fichiers présent sur le serveur que les clients vont monter en lecture pendant l'installation.

### *Le démarrage depuis le réseau*

Nous avons utilisé sur les machines clientes des cartes capables de supporter la norme PXE. La configuration (placée dans le répertoire du serveur /boot/fai/pxelinux.cfg/) est reçue via TFTP. Elle est générée par la commande *fai-chboot -IFv <hostname>*. L'argument F indique que les paramètres passés au noyau sont ceux par défaut. L'environnement PXE utilise l'image originale du noyau (/boot/fai/vmlinuz-install). Comme nous ne savions pas si la configuration des cartes était fixe ou pas, nous avons installé le package tftp spécial recommandé dans la documentation de FAI. Il s'agit de tftpd-hpa, il faut ensuite modifier le fichier inetd.conf.

## **2.2.2. Fonctionnement de FAI**

En résumé, lorsqu'un client démarre, il récupère une configuration et un noyau minimal sur le serveur, puis monte la racine NFS. Ensuite les scripts contenus dans la racine NFS vont :

- formater le ou les disques en suivant les instructions décrites par l'administrateur
- installer le noyau Linux souhaité
- installer les packages voulus
- configurer le système, en exécutant les scripts de configuration

On peut voir à travers ces différentes phases que FAI suit les instructions définies par l'administrateur dans ses fichiers de configuration. Le succès d'une installation dépend donc de sa configuration. Plusieurs configurations sont disponibles lors de l'installation de l'outil et nous ont servi d'exemple.

### **2.2.2.1. Description du serveur d'installation**

Le serveur d'installation est une machine fonctionnant sous Debian. Elle constituée de plusieurs parties :

- Un serveur DHCP
- Un serveur TFTP
- Une racine NFS, crée lors de l'installation de FAI
- Un accès à un miroir Debian : FAI fournit un script pour créer un miroir local, cependant la documentation précise qu'il s'agit d'un miroir pour la version Woody 3.0 (version stable), alors que nous utilisons une Debian Sarge (version testing). Nous avons privilégié l'accès à un miroir extérieur via FTP.
- Un noyau compilé avec les drivers de la carte réseau et faisant un montage de son système de fichier par NFS. Le package fai-kernels fournit un noyau 2.4 et un noyau 2.6 compilés avec de nombreux drivers 3Com. De plus, le fichier de configuration est fournit avec le package. Par conséquent, lors d'une re-compilation de noyau, on peut prendre ce fichier et ajouter les drivers de la carte.

Nous allons décrire ici le déroulement d'une installation. Cependant, il est nécessaire que FAI soit installé et configuré avant. Les configurations des clients doivent également être dans les différents répertoires de configuration (/usr/local/share/fai par défaut, peut être changé grâce à la variable FAI\_LOCATION ; la documentation explique que cela peut être un répertoire d'un serveur CVS).

### 2.2.2.2. Description de la phase d'installation

Lorsque le client démarre, il émet une requête DHCP, le serveur d'installation va lui répondre. Il est nécessaire de s'assurer qu'il n'y a pas d'autres serveurs DHCP sur le réseau qui puissent répondre avant le serveur DHCP de notre propre serveur d'installation. Si c'est le cas, il faut empêcher les autres serveurs de répondre aux requêtes des clients FAI en retirant leurs adresses MAC des serveurs. Puis on récupère par TFTP un noyau qui possède déjà les drivers de la carte réseau et qui est capable de monter son système de fichier via NFS. Le processus `init` est ensuite lancé. Ce processus démarre le script le plus important de FAI : `rcS_fai` (présent dans le répertoire `sbin` sur la racine NFS du serveur d'installation `/usr/lib/fai/nfsroot`), qui va ensuite en appeler d'autres (notamment `subroutine-linux`).

Tout d'abord, une tâche d'initialisation est appelée, elle a pour but de créer le ramdisk (il sera créé dans `/tmp`), de lire le fichier `fai.conf` et de récupérer toute la configuration du DHCP. Le noyau monte son système de fichiers et l'espace de configuration présent sur le serveur, via NFS.

Le noyau charge ensuite des modules (`06hwdetect.source`) et définit l'ordre des classes appelées pendant l'installation.

Les disques sont ensuite partitionnés en suivant les instructions de configuration du répertoire `/fai/disk_config`. Les systèmes de fichiers locaux sont donc créés et l'installation des packages peut commencer. Une fois terminée, les scripts de configuration (`/fai/script`) sont exécutés.

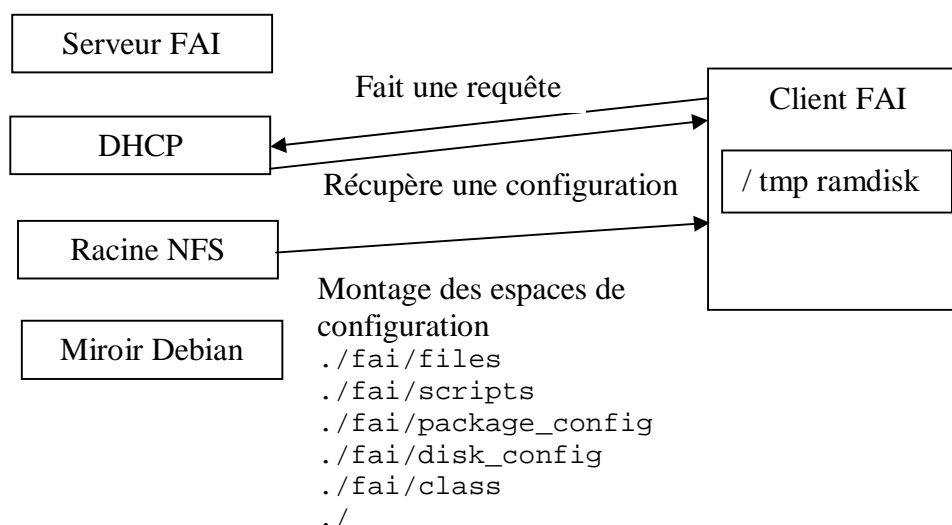


Figure 2. Diagramme de l'installation d'une machine via FAI.

## 3. Installation de FAI

Dans cette partie, nous allons voir étape par étape les différentes actions à effectuer pour installer l'outil d'installation automatique FAI. Nous verrons en même temps les différents problèmes que l'on peut être amené à rencontrer en cas de mauvaises configurations des fichiers.

Avant toutes choses, il est nécessaire d'installer certains paquets qui se révéleront utiles pour la suite. Ils correspondent en fait aux différentes parties énoncées dans la description du serveur d'installation. Mais pour cela, il faut indiquer un endroit où on peut

télécharger les dernières versions de ces paquets. Il suffit donc d'ajouter au fichier `/etc/apt/sources.list` la ligne suivante (correspondant au site officiel de FAI) :

```
deb http://www.informatik.uni-koeln.de/fai/download/
```

Ensuite, il faut exécuter la commande suivante :

```
# apt-get install mknbi dhcp3-server tftpd-hpa rsh-server wget syslinux
```

Les différents paquets installés correspondent à :

`mknbi` : crée des images bootables sur le réseau (Etherboot ou Netboot)

`dhcp3-server` : serveur DHCP

`tftpd-hpa` : serveur TFTP

`rsh-server` : serveur RSH

`wget` : télécharge des documents sur Internet via http ou FTP

`syslinux` : package comprenant pxelinux permettant les boots via PXE

A présent, il faut télécharger FAI, qui se compose de deux paquets : *fai* et *fai-kernels*.

Il faut donc exécuter la commande :

```
# apt-get install fai fai-kernels
```

Le contenu de ces deux packages est listé en Annexe 1.

On obtient ainsi deux fichiers importants pour le paquet FAI :

- Le premier est le fichier `/etc/fai/fai.conf`. Il contient la configuration du paquet FAI.
- Le deuxième est le fichier `/etc/fai/make-fai-nfsroot.conf`. Ce dernier contient les informations nécessaires à la création de la racine NFS (`nfsroot`).

Voyons justement un peu plus en détails les différentes variables du premier fichier `fai.conf` :

- `installserver` est l'adresse IP du serveur d'installation. Ici nous avons donc mis `installserver=152.81.9.95`.

- `mirrorhost` est le nom du miroir utilisé pour l'installation des paquets. Nous avons utilisé le miroir de l'école des mines de Nancy : `mirrorhost=debian.mines.inpl-nancy.fr`.

- `FAI_DEBMIRROR` sert uniquement en cas d'utilisation d'un miroir Debian local.

- `SERVERINTERFACE` correspond à l'interface par laquelle le serveur est relié aux clients : `SERVERINTERFACE=eth0`

- `FAI_REMOTESH` et `FAI_REMOTECP` servent à préciser si on utilise RSH ou SSH pour la copie des fichiers de log du client FAI au cours de son installation. Ici, voici un exemple avec `rsh` :

```
FAI_REMOTESH=rsh
```

```
FAI_REMOTECP=rcp
```

- `FAI_CONFIGDIR` précise l'emplacement de l'espace de configuration sur le serveur d'installation : `FAI_CONFIGDIR=/usr/local/share/fai`

- `FAI_LOCATION` correspond au nom d'hôte et au répertoire distant de l'espace de configuration, qui sera monté via NFS. Sa valeur par défaut est `/usr/local/share/fai` et il doit être exporté à tous les clients pour que tous les fichiers puissent être lus par root : `FAI_LOCATION=$installserver:$FAI_CONFIGDIR`

- NFSROOT correspond à l'emplacement du répertoire sur le serveur d'installation où la racine NFS de FAI est créée : NFSROOT=/usr/lib/fai/nfsroot

Le fichier entier est disponible en Annexe 2.

A présent, voyons les variables importantes du fichier make-fai-nfsroot.conf, qui utilise des variables définies dans le fichier fai.conf :

- FAI\_DEBOOTSTRAP permet de spécifier l'adresse d'un miroir Debian et le nom de la distribution souhaitée. Cette information sera utilisée par la commande debootstrap pour construire la nfsroot. Dans notre cas, nous avons précisé FAI\_DEBOOTSTRAP="sarge http://\$mirrorhost/debian".

- NFSROOT\_ETC\_HOSTS contient normalement le nom et l'adresse IP du miroir Debian que l'on utilise. Mais si les clients ont accès à un serveur DNS, ce qui est notre cas, alors cette variable est inutile.

- KERNELPACKAGE permet de choisir le noyau que l'on souhaite installer sur les clients. Ainsi, pour un noyau 2.4.27, on mettra

```
KERNELPACKAGE=/usr/lib/fai/kernel/kernel-image-2.4.27-fai_1_i386.deb
```

et pour un noyau 2.6.8, on mettra

```
KERNELPACKAGE=/usr/lib/fai/kernel/kernel-image-2.6.8-fai_1_i386.deb.
```

- NFSROOT\_PACKAGES contient une liste des paquets complémentaires qui seront ajoutés à la nfsroot.

- FAI\_BOOT indique pour quel(s) protocole(s), de DHCP et/ou BOOTP, le serveur doit créer des installations lors de l'exécution du script make-fai-nfsroot. Par défaut, il doit créer l'installation pour les deux protocoles. Nous avons conservé la valeur par défaut, à savoir FAI\_BOOT="dhcp bootp".

- FAICLIENTS permet de spécifier pour quels clients on souhaite exporter la racine NFS. Dans notre cas, nous avons précisé un réseau entier, comme ceci : FAICLIENTS="152.81.0.0/20".

Le fichier entier est disponible en Annexe 3.

Après avoir essayé une première fois d'installer FAI, nous nous sommes rendus compte qu'il y avait des erreurs dans le script de création de la racine NFS (make-fai-nfsroot). En effet, dans la fonction upgrade\_nfsroot(), au moment de mettre à jour le fichier sources.list avec la commande apt-get update, il n'existe sur la racine NFS aucune configuration réseau, ce qui provoque une erreur. Nous avons donc modifié le script de sorte à ce qu'il copie le fichier de configuration de l'interface réseau dans la racine NFS avant de faire la mise à jour. Puis nous supprimons le fichier copié à la fin de la fonction puisque le script prévoit la copie de ce fichier par la suite. La fonction upgrade\_nfsroot() modifiée du script make-fai-nfsroot est disponible en Annexe 4.

Un autre problème que nous avons rencontré est le téléchargement du package *raidtools2*. En effet, dans le fichier make-fai-nfsroot.conf, il y a une liste de packages importants voire indispensables à FAI qui doivent être téléchargés. Du jour au lendemain, le

package *raidtools2* est devenu indisponible. La seule solution pour que l'installation de FAI se termine correctement a été de supprimer *raidtools2* de la liste.

Enfin, afin de lancer l'installation de l'outil FAI, il ne reste plus qu'à exécuter le script `fai-setup` qui s'appuie sur les fichiers de configuration `fai.conf` et `make-fai-nfsroot.conf` et qui appelle d'autres scripts comme `make-fai-nfsroot` :

```
# fai-setup
```

Pendant l'exécution de ce script, au moment de la création de la racine NFS, certaines erreurs vont se produire concernant des dépendances non satisfaites. En regardant dans la documentation, il est apparu que ces erreurs étaient normales, qu'elles ne nuisaient en aucun cas au bon déroulement de l'installation et qu'il fallait par conséquent ne pas en tenir compte. La trace d'une installation de FAI réussie est disponible en Annexe 5.

Une fois l'installation de FAI terminée, le script précise que nous n'avons aucune configuration pour FAI. Cependant, des exemples simples sont fournis et il convient donc de les copier dans les bons répertoires :

```
cp -a /usr/share/doc/fai/examples/simple/* /usr/local/share/fai/
```

Il est maintenant temps de passer à la configuration des différents serveurs DHCP, NFS, RSH, ... Il est tout d'abord nécessaire de récupérer les adresses MAC des différents clients que nous souhaitons installer. Les fichiers qui vont être à présent détaillés prennent en compte un serveur que nous appellerons sirus d'adresse IP 152.81.9.95 et deux clients qui ont respectivement les noms et adresses IP demohost – 152.81.2.12 et demohost2 – 152.81.2.55.

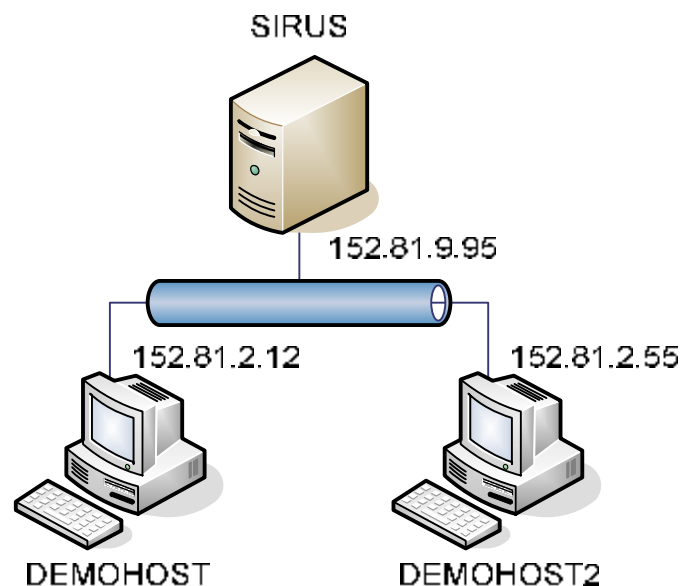


Figure 3. Topologie de test d'installation par FAI.

On doit tout d'abord configurer le serveur DHCP afin qu'il donne toutes les informations nécessaires aux clients au début de leur installation. Le fichier permettant de configurer le serveur est le fichier `dchpd.conf`. C'est donc dans ce fichier que l'on va spécifier le réseau et les machines auxquelles le serveur DHCP devra répondre. Ces différentes valeurs sont :

- subnet et netmask : le réseau et le masque sur lequel le serveur DHCP va officier
- option routers : adresse IP de la passerelle
- option domain-name-servers : adresse IP du serveur DNS
- server-name : adresse IP du serveur d'installation
- option root-path : emplacement de la racine NFS sur le serveur d'installation
- host : nom du client FAI
- hardware ethernet : adresse MAC du client FAI

Pour mieux connaître la syntaxe, le fichier `dhcpd.conf` se trouve en Annexe 6.

Ensuite, il faut ajouter nos machines dans le fichier `/etc/hosts` afin de faire le lien entre leurs noms et leurs adresses IP. On va donc retrouver des lignes se présentant comme ceci :

```
<adresse_IP> <nom_complet> <nom_machine>  
Ex : 152.81.9.95   sirus.loria.fr   sirus
```

Le fichier complet se trouve en Annexe 6.

Il faut également donner accès aux clients NFS afin qu'ils puissent exporter les systèmes de fichiers. Pour cela, il suffit de mettre les chemins que l'on veut exporter aux clients NFS et de préciser les droits accordés dans le fichier `/etc/exports`. Ainsi, nous avons autorisé les clients NFS à accéder en lecture aux répertoires `/usr/local/share/fai` (espace de configuration sur le serveur d'installation) et `/usr/lib/fai/nfsroot` (racine NFS utilisée par le client lors du démarrage). La syntaxe complète de ce fichier est disponible en Annexe 6.

Ensuite, il faut choisir si on souhaite utiliser le protocole RSH ou le protocole SSH. En effet, pendant l'installation des clients NFS, on a une remontée de fichiers de log vers le serveur, contenant des traces d'exécution de certaines tâches ou les erreurs qui ont pu se produire. Pour cela, les clients se connectent sur le serveur via RSH ou SSH. Voici comment configurer les deux protocoles.

Si on utilise RSH, il faut bien sûr le préciser dans le fichier `/etc/fai/fai.conf` et configurer le fichier `/home/fai/.rhosts`. Dans ce fichier, il faut inscrire tous les clients susceptibles de se connecter pour envoyer des fichiers de log. La syntaxe utilisée dans ce fichier est :

```
<nom_complet_client> <login>
```

Ainsi dans notre cas, nous avons inscrit nos deux clients `demohost` et `demohost2` avec les noms de connexion qu'ils peuvent utiliser pour se connecter, à savoir le super utilisateur « root » ou l'utilisateur normal « fai ». Le fichier entier est disponible en Annexe 6. Il faut aussi penser à donner au fichier `.rhosts` les droits 755, sinon les clients ne pourront pas se connecter.

Si on utilise SSH, c'est un peu la même chose. Il faut comme auparavant le préciser dans le fichier `/etc/fai/fai.conf` mais cette fois-ci utiliser le fichier `.shosts` au lieu de `.rhosts`. Il faut cependant quand même modifier une variable dans le fichier `fai.conf` : `FAI_FLAGS`. Lors de l'appel à la commande `fai-chboot`, l'option `-F` prend les paramètres par défaut de la variable `FAI_FLAGS`. Si on veut utiliser SSH pour les connexions à distance, il est donc nécessaire d'enlever l'option `-F` lors de l'appel à la commande `fai-chboot` et de préciser

« sshd » dans la variable FAI\_FLAGS du fichier fai.conf afin de démarrer le démon ssh. Il faut savoir que cette solution n'a pas été testée.

Maintenant, il faut penser à lancer le serveur TFTP au démarrage. Pour cela, il faut ajouter la ligne suivante dans le fichier /etc/inetd.conf disponible en Annexe 7 :

```
tftp dgram udp wait root /usr/sbin/in.tftpd in.tftpd -s /boot/fai
```

Enfin, une fois que tous les fichiers de configuration sont prêts, il ne reste plus qu'à redémarrer tous les demons en effectuant les commandes suivantes :

```
/etc/init.d/inetd restart  
/etc/init.d/nfs-kernel-server restart  
/etc/init.d/nfs-common restart  
/etc/init.d/dhcpd restart
```

Nous avons utilisé PXE pour démarrer les clients FAI. La dernière étape consiste donc à indiquer au client de charger le noyau d'installation et d'effectuer une installation au boot suivant. Pour cela, il suffit d'exécuter la commande suivante sur le serveur d'installation, avec chaque client FAI en paramètre :

```
fai-chboot -IFv demohost demohost2
```

A présent, il ne reste plus qu'à démarrer les clients et à laisser l'installation automatique faire son travail. La trace d'une installation réussie est disponible en Annexe 8.

## 4. Configuration

### 4.1. Notions

#### 4.1.1. Arborescence du serveur FAI

Les principaux répertoires pour le serveur FAI sont les suivants :

- /usr/lib/fai/

Dans ce répertoire, on retrouve trois sous répertoires importants :

-> /usr/lib/fai/kernel qui contient les noyaux à installer sur les disquettes lors de l'utilisation du script make-fai-bootfloppy ;

-> /usr/lib/fai/sbin qui contient des scripts utilisés durant l'installation ;

-> /usr/lib/fai/nfsroot qui est la racine NFS utilisée par le client lors du démarrage.

- /usr/share/fai

Ce répertoire contient un ensemble de routines et de fichiers de configurations qui vont être recopiés sous « /usr/lib/fai/nfsroot/... » lors du lancement de la procédure « fai-setup ». Ces fichiers seront ensuite utilisés lors de la phase d'installation du client.

- /usr/local/share/fai

C'est sous ce répertoire que nous allons trouver les différentes « classes » définies pour notre système d'installation automatique, ainsi que plusieurs scripts écrits en perl, bourne shell...

- /etc/fai

Il contient principalement le fichier de configuration fai.conf pour l'installation du serveur FAI.

### 4.1.2. Le concept de classes

C'est ce concept qui fait de FAI un outil puissant, évolutif et adaptatif. Cette définition de « classes » permet d'identifier différentes machines en fonction : du service, du matériel, de l'affectation, de la localisation... Il permet aussi une évolution facile et une modification rapide de l'automate d'installation. Chaque machine va appartenir à plusieurs classes. Les classes sont définies par l'administrateur du serveur FAI par l'intermédiaire de scripts exécutés lors de l'installation du client. Toutes les classes sont identifiées par un nom en majuscule, sauf la classe qui correspond au nom de la machine cliente qui sera en minuscule.

Il existe deux types de classes :

- Les classes par défaut : DEFAULT, hostname et LAST ;
- Les classes spécifiques : toutes les autres classes.

L'affectation de « classes » à un client est faite par l'intermédiaire de scripts mais aussi grâce à l'existence d'un fichier portant le nom du client, le hostname. Ce fichier, s'il existe, est parcouru. Il ne doit comporter que des noms en majuscules identifiant différentes classes qui seront utilisées lors de l'installation du client. Le fichier doit se trouver sous le répertoire /usr/local/share/fai/class. Ce répertoire contient tous les scripts d'initialisation du système de classes des clients.

Ainsi, on retrouve plusieurs sous répertoires dans /usr/local/share/fai :

- class : c'est celui dont on parlait précédemment, qui contient les différents scripts d'initialisation du système de classes des clients ;
- disk\_config : dans ce répertoire, on trouve un script qui contient les informations pour la configuration du disque dur, à savoir les différentes partitions ;
- files : ce répertoire contient des fichiers de configuration ;
- hooks : on trouve dans ce répertoire un script qui gère les messages d'erreurs ;
- package\_config : ce répertoire contient pour chaque classe les packages à installer ;
- scripts : dans ce répertoire, on trouve les scripts de configuration des différents packages qui vont être installés ainsi que d'autres configurations comme celle du réseau par exemple.

## 4.2. Configuration des clients

Une fois l'installation d'une machine via FAI terminée, il nous a été demandé d'installer toujours via FAI des serveurs de terminaux X. Pour installer un tel serveur, il faut télécharger deux principaux packages : un qui va gérer l'authentification des terminaux X auprès du serveur (xdm) et un autre qui sera l'interface graphique des clients une fois qu'ils seront connectés (icewm). Ainsi, dans le répertoire /usr/local/share/fai/package\_config, nous avons créé une classe XSERVER dans laquelle on a spécifié l'installation des packages xdm et icewm. Le fichier est disponible en Annexe 9.

Mais il faut prévoir la configuration de ces packages après leur installation sur les clients. Pour cela, il existe des scripts de configuration, écrits en cfengine ou sh. Dans notre cas, nous devons modifier deux fichiers :

- Xaccess : fichier qui contient les clients ayant le droit d'accéder aux différents serveurs de terminaux X. Nous avons choisi de donner l'accès à tous les clients souhaitant se connecter, donc à tout le monde.
- Xservers : fichier qui contient la liste des serveurs de terminaux X. Nous avons donc inscrit dans ce fichier nos deux clients FAI demohost et demohost2 puisqu'ils sont censés être installés en serveurs de terminaux X.

Nous avons donc créé un script, que l'on a appelé S30, dans le répertoire /usr/local/share/fai/scripts/DEFAULT. Ainsi, une fois les packages installés, les scripts de configuration sont appelés et les serveurs de terminaux X sont prêts. Le script cfengine S30 est disponible en Annexe 9.

Enfin, dans nos deux classes demohost et demohost2 situées dans le répertoire /usr/local/share/fai/class, il suffit d'ajouter la ligne XSERVER faisant appel à la classe XSERVER que l'on a créée auparavant. Le fichier demohost est lui aussi disponible en Annexe 9.

Nous aurions également du faire la même chose pour installer des clusters. Il faut savoir que des classes d'installation de clusters sont déjà définies et prêtes à être utilisées. Ces classes permettent d'installer différents packages, dont des packages de calcul et de gestion de clusters.

## 5. Résultats.

Malgré de nombreuses tentatives, nous n'avons réussi à installer une machine de base à l'aide du système FAI qu'une seule fois. La machine était complète et prête à l'emploi. Nous avons réessayé par la suite, mais sans réussite. En effet, nous avons rencontré de nombreux problèmes, dont la plupart vous ont été expliqués précédemment. A présent, les fichiers de configuration sont bons mais nous avons des problèmes au cours de l'exécution de certains scripts.

Par conséquent, et comme vous avez pu le voir, nous avons regardé comment installer des serveurs de terminaux X mais nous n'avons malheureusement pas pu faire de tests. De plus, il nous paraissait normal de valider cette étape avant de passer à la suite, et nous n'avons donc pas pu étudier la configuration et l'installation de clusters.

Cependant, nous avons quand même trouvé des résultats d'autres personnes pouvant donner un ordre d'idée de ce qu'il est possible de faire avec le système d'installation automatique FAI. Ainsi, voici un petit tableau montrant des temps d'installation de machines en fonction de leurs caractéristiques et du nombre de packages installés.

Host		RAM	Disk type	Software installed	Installation time
CPU	MHz				
Pentium 4	2800	1024MB	IDE	948 MB	5 min
Athlon XP	1433	896MB	SCSI	1 GB	6 min
AMD-K7	500	320MB	IDE	780 MB	12 min
Pentium III	850	256MB	IDE	820 MB	10 min
Pentium III	850	256MB	IDE	180 MB	3 min
PentiumPro	200	128MB	IDE	800 MB	28 min

**Tableau 1. Temps d'installation de machines en fonction de leurs caractéristiques.**

En consultant le site officiel de FAI, on peut voir les remarques de personnes ayant déjà utilisé FAI. Ainsi on se rend vite compte de la diversité dans l'usage de cet outil. Il sert principalement à installer un grand nombre de machines plus rapidement, mais aussi de système de réinstallation après des pannes. D'autre part, la plupart des utilisateurs travaillent dans des centres de recherche ou dans des universités. Enfin, on voit que cet outil est largement utilisé, ou tout du moins testé, puisqu'on retrouve des commentaires de personnes n'ayant toujours pas réussi à installer une machine jusqu'à des utilisateurs expérimentés ayant déjà installé 300 machines et affirmant qu'ils ne peuvent pas s'en passer.

Nous avons utilisé la version 2.6.7 de FAI au cours de ce projet. Néanmoins, il faut savoir que la version 2.7 est désormais disponible. Les améliorations qu'elle apporte sont importantes, car elles corrigent plusieurs problèmes qui peuvent se poser pendant l'installation.

Tout d'abord le logiciel raidtools2 a été supprimé de la liste des logiciels à installer dans le fichier de configuration `make-fai-fsroot.conf`. Le logiciel a été remplacé par `mdadm`. Après les tests, il s'est avéré que cela ne posait plus de problèmes, mais à présent il s'agit du logiciel `ext2resize` qui n'est plus disponible.

Le problème de la configuration réseau lors de création de la racine NFS a été fixé, désormais le script `make-fai-nfsroot` utilise bien le fichier `resolv.conf` de la machine pour télécharger les logiciels.

La nouveauté de la version 2.7 est la possibilité de créer un CD (ou un DVD) pour réaliser l'installation automatique. Le réseau n'est donc plus nécessaire car tout le système peut être installé depuis le CD. Le script `fai-cd` peut créer une image ISO avec l'espace de configuration, la racine NFS et un miroir Debian partiel.

## Conclusion

Ce projet avait pour but, dans un premier temps, d'étudier le système d'installation automatique de machines, Fully Automatic Installation, au travers des techniques qu'il utilise pour réaliser des installations automatiques. Dans un second temps, il s'agissait de réaliser une maquette.

Concernant la documentation, nous nous sommes entièrement tournés vers Internet afin de nous familiariser avec l'outil FAI. Nous avons trouvé des didacticiels et très rapidement, nous nous sommes orientés vers la pratique, à savoir l'installation d'une machine.

Mais nous avons rencontré de nombreux problèmes, de nature très différente, et avons donc passé beaucoup de temps avant d'arriver à installer une machine avec succès. Par conséquent, nous avons simplement préparé les configurations en vue de l'installation de serveurs de terminaux X, sans pouvoir faire le moindre test. Quant à la partie sur l'installation de clusters, nous n'avons pas eu le temps de la traiter.

Sur un plan plus personnel, ce projet nous a permis d'acquérir quelques compétences en système Debian, qui nous était pratiquement inconnu avant de commencer le projet. Ce projet était très intéressant et paraissait accessible à première vue, mais nous nous sommes très vite rendus compte des problèmes que peuvent poser un certain manque de connaissance en Linux. Bien que nous n'ayons pas été plus loin que l'installation d'une simple machine au cours de ce projet, il apparaît que le système FAI est un outil pouvant s'avérer très utile et efficace lors de l'installation d'un grand nombre de machines ou de clusters. Il n'est de plus pas très difficile à mettre en œuvre, à condition de disposer de quelques notions fondamentales !

## Bibliographie

<http://www.informatik.uni-koeln.de/fai/>

[http://sari.inpg.fr/rubriques/themes/zone\\_publicque.groupe\\_linux/FAI-LMC.html](http://sari.inpg.fr/rubriques/themes/zone_publicque.groupe_linux/FAI-LMC.html)

<http://www.math.u-bordeaux.fr/math/cellule/>

<http://www.lea-linux.org/>

## Annexe 1

### Contenu du package FAI :

#### conf/

- menu.lst
- make-fai-nfsroot.conf
- install\_packages.conf
- sources.list
- fai.conf
- apt.conf
- pxelinux.cfg
- dhclient.conf
- fai\_modules\_off
- apt.conf.nfsroot

#### debian/

- changelog
- control
- premm
- copyright
- rules
- postinst
- docs
- dirs
- conffiles
- compat
- undocumented
- postrm
- preinst
- doc-base.package

#### doc/

##### entities/

- fai-guide.sgml
- QUESTIONNAIRE
- classes\_description.txt
- links.html
- README.package\_problems
- Makefile
- common.ent
- FAQ
- README.disk\_config
- changelog.old
- QUESTIONNAIRE
- README.package\_config
- version.ent
- fai-boot-floppy.txt
- README.collect\_sysinfo
- fai\_technical\_report.ps.gz

##### examples/

- beowulf/
- class/
- etc/
- hooks/
- simple/
- utils/

**kernel/**

Makefile  
README  
enable-dhcp  
fai-kernel-config-bootp  
kernel-config  
emptydosdisk.gz  
imagegen\_firstblock  
config  
config.diff  
config-2.2.15

**lib/**

fai-savelog-ftp  
task\_sysinfo  
prepare\_apt  
fai-savelog  
create\_ramdisk  
fai-divert  
get-boot-info  
fai-mount-disk  
create\_resolv\_conf  
check\_status  
load\_keymap\_consolechars  
disk-info  
mount2dir  
list\_disks  
load\_keymap\_consolechar  
Fai.pm  
subroutines  
subroutines-linux  
subroutines-sunos

**man/**

fai-cd.8  
fai-mirror.1  
install\_packages.8  
fai-debconf.1  
fcopy.8  
make-fai-nfsroot.8  
fai-setup.8  
fai-chboot.8  
bootsector.8  
make-fai-bootfloppy.8  
fai-do-scripts.1  
faimond.8  
fai-class.1  
fai-start-stop-daemon.8  
fai-do-script.1  
ftar.8

**scripts/**

make-fai-nfsroot  
rcS\_fai  
fai-mirror  
fai-cd  
fcopy  
install\_packages  
fai-do-scripts  
fai-debconf

```
fai-class
fai-chboot
make-fai-bootfloppy
fai-setup
ftar
setup_harddisks
fai-start-stop-daemon
faireboot
bootsector
faimond
dhclient-perl
start-stop-daemon
mount2dir
device2grub
mount2target
make-fai-tftpimage
fcopy.pl
mk3comimage
dhclient-script

share/
  subroutines-linux
  subroutines
  Fai.pm
  subroutines-sunos

sunos/
  bin/
  class/
  disk_config/
  files/
  package_config/
  scripts/
  makefile
  README.sunos
  rules

templates/
  class/
  debconf/
  disk_config/
  fai_config/
  files/
  hooks/
  package_config/
  scripts/

utils/
  mkdebmirror
  chkdebnames
  softupdate
  rshall
  create-nfsroot-tar
  prtnetgr
  all_hosts
  tlink

NEWS
Makefile
README
VERSION
```

**THANKS**  
**TODO**  
**README.build-sources**  
**INSTALL**

**Contenu du package FAI-KERNELS :**

debian/  
    **changelog**  
    **rules**  
    **control**  
    **compat**  
    **copyright**  
    **docs**  
    **dirs**

**NEWS**  
**README**  
**fai-kernel-config**  
**fai-kernel-config-2.4**  
**versions**  
kernel-config  
kernel-config-2.4

## Annexe 2

### Fichier /etc/fai/fai.conf

```
# $Id: fai.conf,v 1.79 2004/07/08 12:33:40 lange Exp $
# /etc/fai/fai.conf -- configuration for FAI (Fully Automatic Installation)

# installserver must be the name seen by the install clients
installserver=152.81.9.95

# the name of the Debian mirror
mirrorhost=debian.mines.inpl-nancy.fr

# Don't use the variable FAI_SOURCES_LIST any more. Instead use
/etc/fai/sources.list

# Access to Debian mirror via NFS mounted directory
# If FAI_DEBMIRROR is defined, install clients mount it to $MNTPOINT
# FAI_DEBMIRROR=$mirrorhost:/files/scratch/debmirror

# if your install server has multiple ethernet device, use this one to
determine # its hostname. Default eth0. Set to the interface to which the
Beowulf clients # are connected.
SERVERINTERFACE=eth0

# LOGUSER: an account on the install server which saves all log-files and
which # can change the kernel that is booted via network. Configure .rhosts
for this
# account and PAM, so that root can log in from all install clients without
# password. This account should have write permissions for /boot/fai. For
# example, you can use write permissions for the group linuxadm. chgrp
linuxadm
# /boot/fai;chmod g+w /boot/fai. If the variable is undefined, this feature
is
# disabled
LOGUSER=fai
# use ssh or rsh for copying log files to user fai and for changing tftp
# symbolic link
FAI_REMOTESH=rsh
FAI_REMOTECP=rcp

# set protocol type for saving logs, default is rcp/scp. Set to ftp if
desired.
FAI_LOGPROTO=
# Name of log-server. If undefined, the install server will be used.
LOGSERVER=
# writable directory on remote server, when using FTP protocol
LOGREMOTEDIR="upload"
# password for login to log server, when using FTP protocol
LOGPASSWD=

# the configuration space on the install server
FAI_CONFIGDIR=/usr/local/share/fai
# the location of the config space, as seen by the install client
# it can also be overwritten with T170 via BOOTP
FAI_LOCATION=$installserver:$FAI_CONFIGDIR

# the following variables are read only for most users
```

```
# mount point where the mirror will be mounted
MNTPOINT=/mnt2

# directory on the install server where the nfsroot for FAI is
# created, approx size: 160MB, also defined in bootptab or dhcp.conf
NFSROOT=/usr/lib/fai/nfsroot

# the local configuration directory on the install client
FAI=/fai
# the type of operating system (linux, sunos)
OS_TYPE=`uname -s | tr /A-Z/ /a-z/`
```

## Annexe 3

### Fichier /etc/fai/make-fai-nfsroot.conf

```
# these variables are only used by make-fai-nfsroot(8)
# here you can use also variables defined in fai.conf (like $mirrorhost)
# Add a line for mirrorhost and installserver when DNS is not available
# on the clients. This line(s) will be added to $nfsroot/etc/hosts.
#NFSROOT_ETC_HOSTS="192.168.1.250 $mirrorhost"

FAI_DEBOOTSTRAP="sarge http://$mirrorhost/debian"

# your extra packages which will be installed into the nfsroot, space
# separated
NFSROOT_PACKAGES="expect" # Mettre les noms de packages derrière le mot clé
# expect

# this local repository holds your local packages that can be installed to
# the install clients. Don't forget to create the index file Packages.gz!
FAI_LOCAL_REPOSITORY="deb file:/fai/files packages/"

# the encrypted root password on all install clients during
# installation process; used when log in via ssh; pw is: fai
FAI_ROOTPW="56hNVqht51tzc"

# this kernel package will be used when booting the install clients
KERNELPACKAGE=/usr/lib/fai/kernel/kernel-image-2.4.27-fai_1_i386.deb
# KERNELPACKAGE=/usr/lib/fai/kernel/kernel-image-2.6.8-fai_1_i386.deb

# location of a identity.pub file; this user can log to the install
# clients in as root without a password; only useful with FAI_FLAGS="sshd"
# SSH_IDENTITY=/home/admin/.ssh/identity.pub

# which of DHCP and/or BOOTP should the server create setups for.
# Default are to create setups for both
FAI_BOOT="dhcp bootp"

# export $NFSROOT to this netgroup or this range of IP addresses
# (eg. FAICLIENTS="192.168.1.0/24")
FAICLIENTS="152.81.0.0/20"

# - - - - -
# following lines should be read only for you when you are using fai on
# i386

FAI_DEBOOTSTRAP_OPTS="--arch i386 --exclude=pcmcia-
cs,ppp,pppconfig,pppoe,pppoeconf,dhcp-client,exim4,exim4-base,exim4-
config,exim4-daemon-
light,mailx,at,fdutils,info,modconf,libident,logrotate,exim"

# size of the nfsroot. Only informational purpose
nfssize="250MB"

# FAI needs these packages that are install into the nfsroot
# Suppression package raidtools2
packages="module-init-tools dhcp3-client ssh file rdate hwinfo portmap
bootpc rsync wget rsh-client less dump reiserfsprogs usbutils
ext2resize hdparm smartmontools parted lvm2
dnsutils ntpdate dosfstools cfengine cvs jove xfsprogs xfsdump
sysutils dialog discover mdetect libnet-perl netcat libapt-pkg-perl"
```

## Annexe 4

### Fonction `upgrade_nfsroot()` du script `make-fai-nfsroot`

```
upgrade_nfsroot() {  
  
# Ligne originale enlevée  
#   cp -p $v /etc/resolv.conf $NFSROOT/etc/resolv.conf-installserver  
#### Modification  
    cp -p -v /etc/resolv.conf $NFSROOT/etc/resolv.conf  
    cp -p -v /etc/network/interfaces $NFSROOT/etc/network/interfaces  
####  
  
# Mise à jour du fichier /etc/apt/sources.list  
    $ROOTCMD apt-get update  
# Installation de packages :  
# -f : essaie de réparer un système dont les dépendances sont défectueuses  
# -y : répond « oui » automatiquement  
# -u : affiche les paquets mis à niveau  
    $ROOTCMD apt-get -fyu install  
# Mise à jour du cache des paquets et recherche des dépendances  
# défectueuses  
    $ROOTCMD apt-get check  
  
    rm -rf $NFSROOT/etc/apm  
    mount -t proc /proc $NFSROOT/proc  
#### Modification  
    rm -rf $NFSROOT/etc/network/interfaces  
####  
  
# fake start-stop-daemon  
fdivert /etc/init.d/rcS /sbin/start-stop-daemon /sbin/discover-modprobe  
cp /sbin/fai-start-stop-daemon $NFSROOT/sbin/  
ln -s /sbin/fai-start-stop-daemon $NFSROOT/sbin/start-stop-daemon  
$ROOTCMD apt-get -y dist-upgrade  
}
```

## Annexe 5

### Trace d'une installation réussie de FAI

```

sirus[~]# fai-setup
Adding system user fai...
Adding new user fai (101) with group nogroup.
Creating home directory /home/fai.
/home/fai/.rhosts created.
User account fai set up.
Creating FAI nfsroot can take a long time and will
need more than 230MB disk space in /usr/lib/fai/nfsroot.
Creating nfsroot for sarge using debootstrap
dpkg: base-passwd: dependency problems, but configuring anyway as you request:
base-passwd depends on libc6 (>= 2.3.2.ds1-4); however:
Package libc6 is not installed.
dpkg: base-files: dependency problems, but configuring anyway as you request:
...
Automatically converting /etc/network/interfaces succeeded.
Old interfaces file saved as interfaces.dpkg-old.
Creating base.tgz
`/etc/fai/sources.list' -> `etc/apt/sources.list'
Upgrading /usr/lib/fai/nfsroot
Adding additional packages to /usr/lib/fai/nfsroot:
module-init-tools dhcp3-client ssh file rdate hwinfo
bootpc rsync wget rsh-client less dump reiserfsprogs usbutils
dpkg-dev ext2resize hdparm smartmontools parted raidtools2 lvm2
dnsutils ntpdate dosfstools cfengine cvs jove
sysutils dialog discover mdetect libnet-perl netcat libapt-pkg-perl
grub lilo read-edid kudzu hwtools dmidecode
hostname: Host name lookup failure
/var/lib/dpkg/tmp.ci/preinst: line 7: lvscan: command not found
Creating SSH2 RSA key; this may take some time ...
Creating SSH2 DSA key; this may take some time ...
starting OpenBSD Secure Shell server: sshd.
`/etc/fai/fai.conf' -> `/usr/lib/fai/nfsroot/etc/fai/fai.conf'
`/etc/fai/make-fai-nfsroot.conf' -> `/usr/lib/fai/nfsroot/etc/fai/make-fai-nfsroot.conf'
`/etc/fai/sources.list' -> `/usr/lib/fai/nfsroot/etc/fai/sources.list'
DHCP environment prepared. If you want to use it, you have to enable the dhcpd and the
tftp-hpa daemon.
Image Creator for MBA ROMs v1.01, Date: Nov 26, 2001
Design and Coding by Nick Kroupetski <NickKroupetski@hotmail.com>
Usage: imggen [OPTION] inputfile outputfile
-a, Add 3Com MBA/BootWare support
-r, Remove 3Com MBA/BootWare support from image file
-i, Show information on an image
-h, Help screen
In filename: /boot/fai/installimage
Out filename: /boot/fai/installimage_3com
Adding MBA support...
MBA support has been succesfully added
BOOTP environment prepared.
make-fai-nfsroot finished properly. <= *
Stopping NFS kernel daemon: mountd nfsd.
Unexporting directories for NFS kernel daemon...done.
Exporting directories for NFS kernel daemon...done.
Starting NFS kernel daemon: nfsd mountd.
You have no FAI configuration. Copy the simple examples with:
cp -a /usr/share/doc/fai/examples/simple/* /usr/local/share/fai
Then change the configuration files to meet your local needs.
FAI setup finished. <= *

```

Remarque : Les deux lignes marquées par une étoile sont importantes. La première signifie que le script de création de la racine NFS s'est bien déroulé et la deuxième que le script d'installation de FAI s'est bien terminé.

## Annexe 6

### Fichier /etc/dhcp3/dhcpd.conf

```
# dhcpd.conf for fai
# replace faiserver with the name of your install server

option dhcp-max-message-size 2048;

allow bootp;
allow booting;

filename "pxelinux.0";

# the server from which to load the initial boot file if different
# from server-name (if the DHCP server is not also the TFTP server)
# next-server faiserver;

subnet 152.81.0.0 netmask 255.255.240.0 {
    option routers 152.81.1.1;
    option domain-name-servers 152.81.1.25;
    server-name "152.81.9.95";
    option
        "152.81.9.95:/usr/lib/fai/nfsroot,rsize=8192,wsiz=8192,acregmin=1800,acreg
max=1800,acdirmin=1800,acdirmax=1800";
}

# perl -ane ' {print "host atom {hardware ethernet $1;fixed-address
atom}";}'
host demohost {hardware ethernet 00:c0:4f:10:d5:e3; fixed-address
demohost;}
host demohost2 {hardware ethernet 00:c0:4f:10:d6:7c; fixed-address
demohost2;}
```

### Fichier /etc/hosts

```
127.0.0.1 localhost.localdomain localhost
152.81.9.95 sirus.loria.fr sirus
152.81.2.12 demohost.loria.fr demohost
152.81.2.55 demohost2.loria.fr demohost2
152.81.1.25 preny.loria.fr preny
```

### Fichier /etc/exports

```
# /etc/exports: the access control list for filesystems which may be
exported
#          to NFS clients.  See exports(5).
/usr/local/share/fai 152.81.0.0/255.255.240.0(async,ro)
/usr/lib/fai/nfsroot 152.81.0.0/255.255.240.0(async,ro,no_root_squash)
```

### Fichier /home/fai/.rhosts

```
demohost.loria.fr root
demohost.loria.fr fai
demohost2.loria.fr root
demohost2.loria.fr fai
```

## Annexe 7

### Fichier /etc/inetd.conf

```
# /etc/inetd.conf:  see inetd(8) for further informations.
#
# Internet server configuration database
#
#
# Lines starting with "#:LABEL:" or "#<off>#" should not
# be changed unless you know what you are doing!
#
# If you want to disable an entry so it isn't touched during
# package updates just comment it out with a single '#' character.
#
# Packages should modify this file by using update-inetd(8)
#
# <service_name> <sock_type> <proto> <flags> <user> <server_path> <args>
#
#:INTERNAL: Internal services
#echo          stream      tcp    nowait      root    internal
#echo          dgram udp    wait    root    internal
#chargen      stream      tcp    nowait      root    internal
#chargen      dgram udp    wait    root    internal
#discard      stream      tcp    nowait      root    internal
#discard      dgram udp    wait    root    internal
#daytime      stream      tcp    nowait      root    internal
#daytime      dgram udp    wait    root    internal
#time         stream      tcp    nowait      root    internal
#time         dgram udp    wait    root    internal

#:STANDARD: These are standard services.

#:BSD: Shell, login, exec and talk are BSD protocols.
shell         stream      tcp    nowait      root    /usr/sbin/tcpd
             /usr/sbin/in.rshd
login         stream      tcp    nowait      root    /usr/sbin/tcpd
             /usr/sbin/in.rlogind
exec          stream      tcp    nowait      root    /usr/sbin/tcpd
             /usr/sbin/in.rexecd

#:MAIL: Mail, news and uucp services.

#:INFO: Info services
ident         stream      tcp    wait    identd      /usr/sbin/identd identd

#:BOOT: Tftp service is provided primarily for booting.  Most sites
# run this only on machines acting as "boot servers."
tftp dgram udp wait root /usr/sbin/in.tftpd in.tftpd -s /boot/fai

#:RPC: RPC based services

#:HAM-RADIO: amateur-radio services

#:OTHER: Other services
```

## Annexe 8

### Trace d'une installation via FAI réussie

```

Managed PC Boot Agent (MBA) v4.00
.
.
Pre-boot eXecution Environment (PXE) v2.00
.
.
DHCP MAC ADDR: 00 04 75 74 A2 43
DHCP.../
CLIENT IP: 152.81.2.12 MASK: 255.255.240.0 DHCP IP: 152.81.9.95
GATEWAY IP: 152.81.1.1

PXELINUX 2.04 (Debian, 2004-06-24) Copyright (C) 1994-2003 H. Peter Anvin
UNDI data segment at: 0009D740
UNDI data segment size: 3284
UNDI code segment at: 00090000
UNDI code segment size: 24C0
PXE entry point found (we hope) at 9D74:00F6
My Ip address seems to be C0A801C0 192.168.1.12
ip=192.168.1.150:192.168.1.250:192.168.1.254:255.255.255.0
TFTP prefix:
Trying to load pxelinux.cfg/00-04-75-74-A2-43
Trying to load pxelinux.cfg/C0A801C0
Loading vmlinuz-install.....Ready.
Uncompressing Linux... OK, booting the Kernel.
Linux version 2.4.26 (root@kueppers) (gcc version 2.95.4 20011002)
.
.
.
Sending DHCP requests ., OK
IP-Config: Got DHCP answer from 152.81.9.95, my address is 152.81.2.12
IP-Config: Complete:
device=eth0, addr=192.168.1.12, mask=255.255.240.0, gw=152.81.1.1,
host=demohost, domain=localdomain, nis-domain=(none),
bootserver=152.81.9.95, rootserver=152.81.9.95,
rootpath=/usr/lib/fai/nfsroot,rw,rsize=8192,wsiz=8192,Looking up port of RPC 1000003/2
on 152.81.9.95
Looking up port of RPC 1000005/1 on 152.81.9.95
VFS: Mounted root (nfs filesystem).
.
.
-----
Fully Automatic Installation for Debian GNU/Linux
FAI 2.6, 26 july 2004
Thomas Lange <lange@informatik.uni-koeln.de>
-----

Calling task_confdir
Kernel parameters: ip=dhcp devfs=nomount FAI_ACTION=install root=/dev/nfs
FAI_FLAGS=verbose,sshd,createvt,Defining variable: ip=dhcp
Defining variable: devfs=nomount
Defining variable: FAI_ACTION=install
Defining variable: root=/dev/nfs
Defining variable: FAI_FLAGS=verbose,sshd,createvt,syslogd
Defining variable: BOOT_IMAGE=vmlinuz-install
Reading /tmp/fai/boot.log
FAI_FLAGS: verbose=1
FAI_FLAGS: sshd=1
FAI_FLAGS: createvt=1
FAI_FLAGS: syslogd=1
Configuration space /fai mounted from faiserver:/usr/local/share/fai
Monitoring to server faiserver enabled.
Calling task_setup
.
.
Calling task_defclass
/usr/bin/fai-class: Defining classes.

```

```
.  
.  
Calling task_action  
FAI_ACTION: install  
Performing FAI installation. All data may be overwritten!  
.  
.  
Disable swap device /dev/hda5  
Fri Mar 25 19:10:26 CEST 2005  
The installation took 296 seconds.  
Calling task_chboot  
Calling hook: savelog.LAST  
No ERRORS found in log files. See /tmp/fai/error.log.  
savelog.LAST OK.  
Calling task_savelog  
Save log files via rsh to fai@faiserver:demohost/install-20040725_220535  
Calling task_faiend  
Press <RETURN> to reboot or ctrl-c to execute a shell
```

**Remarque :** Les traces d'installation correspondent aux fichiers .log dans le répertoire */home/fai/user\_fai/date\_install/*.

## Annexe 9

### Fichier /usr/local/share/fai/package\_config/XSERVER

```
PACKAGES install
```

```
xdm  
icewm
```

### Fichier /usr/local/share/fai/scripts/DEFAULT/S30

```
#!/usr/bin/cfengine  
editfiles:  
  any::  
    { ${target}/etc/X11/xdm/Xaccess  
      InsertLine  "* CHOOSER BROADCAST"  
    }  
    { ${target}/etc/X11/xdm/Xservers  
      LocateLineMatching  ":0 local /usr/X11R6/bin/X vt7 -dpi 100  
-nolisten tcp"  
      InsertLine  "demohost:0 foreign"  
    }  
}
```

### Fichier /usr/local/share/fai/class/demohost

```
# these are classes used for a Debian demo host  
  
# class for partitioning  
SMALL_IDE  
  
# needed for booting  
MBR GRUB  
  
# configure network and become a dhcp client  
NETWORK  
DHCP  
  
DEMO  
  
# uncomment the next lines if you want to have a more software  
XSERVER
```