

Deploying Debian systems with FAI

SWIRL

FAI helps you automate the process of installing and configuring new Debian systems.

**BY STEFFEN BORNEMANN
AND CHRISTOPH KARG**

The Fully Automatic Installation (FAI) framework developed at the University of Cologne, Germany, relieves the headache of mass installing Debian systems. FAI offers several customization features and allows you to set up virtual systems. The new 3.2.9 [1] version supports Debian Lenny and includes a new partitioning tool.

How It Works

FAI manages the installation process by automating the steps executed by the

Debian installer. As shown in Figure 1, an FAI server system runs the FAI daemon and also hosts the services necessary to support network-based installation, including NFS, TFTP, and DHCP. In the most basic scenario, a client with a preboot execution environment (PXE)-based network adapter remote boots, gets an IP address from the DHCP server, then obtains the necessary files through TFTP to launch the remote installation.

FAI also works with older computers that do not have PXE support. If you don't have PXE support, you will need to launch the setup manually from a boot image on a floppy disk, CD-ROM, or USB stick. Whether or not your system supports PXE, the start routine will load a bootable kernel. The client then contacts the FAI server and uses NFS to bind its root and configuration directories. These directories contain all the information required to install the client.

FAI lets you define classes of computers with a common location or role. The class definition associates scripts, configurations, and a custom set of software

Other Installation Tools

The *setup_harddisks* and *setup-storage* programs partition the hard disk. These packages also include the *fcopy* and *ftar* commands, which copy files to the client independent of its class. The *fai-doc* package includes the FAI documentation and some sample configurations. After installing the package, you will find the files in the `-usr/share/doc/fai-doc` directory.

The *fai-nfsroot* package is not intended for installation on normal computers. Instead, the FAI setup procedure downloads this package from the mirror while configuring the server, dropping its content into the `/srv/fai/nfsroot` directory. The clients use NFS to mount this directory as their root.

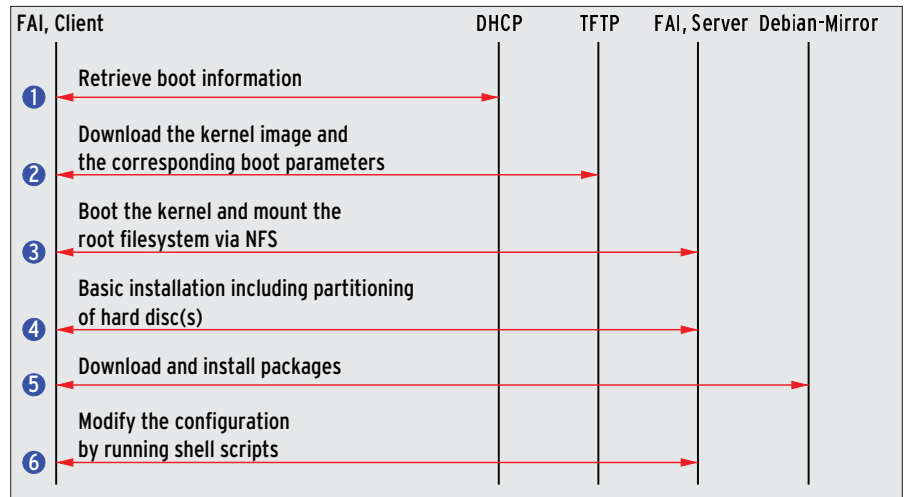


Figure 1: FAI goes through several phases to install a Debian system.

Debian mirror before going on to install the packages with Apt. Finally, the client is customized according to the specs provided through FAI.

Installing the Server

The FAI server contains several components. Figure 2 illustrates the installation steps. The first step is to set up the daemon responsible for distributing the configuration. To do so, you need a Debian machine with NFS, TFTP, and DHCP. In addition, I recommend the FAI packages *fai-server*, *fai-doc*, *fai-quickstart*, and *fai-nfsroot*. These packages are available through the normal Debian repositories.

The *fai-server* package is the starting point for the server installation. The list of dependencies includes packages for NFS, DHCP, and the TFTP server. Also, do not forget the *fai-setup* command for the central server configuration. *make-fai-bootfloppy* creates boot media such as CD-ROMs and floppies for hardware without PXE boot support. *fai-client* provides the programs run by the client during the installation.

fai-quickstart installs all of the tools necessary for the FAI installation and copies the sample configurations to the correct location. It installs all the required packages and creates configuration templates in the */etc/fai* directory. Unless you are using NFS to mount a Debian mirror, you won't need to change the parameters in the *fai.conf* file that define the central settings for the FAI server.

The *make-fai-nfsroot.conf* file tells FAI how to configure the NFS

root. If the FAI server does not have a DNS entry, you need to comment out the *NFSROOT_ETC_HOSTS* line. The file already has default IP addresses and names, which you can modify to match your local environment. You also need to

packages with the client. As you will learn later in this article, a client can belong to more than one class.

An init script launches the local installation. The script investigates the hostname and the mounted configuration data to decide to which installation class the client belongs. The installation process formats the client's hard disk and downloads the required packages from a

Listing 1: DHCP Configuration

```
01 group {
02     allow booting;
03     next-server 192.168.0.100;
04     filename "pxelinux.0";
05     option root-path "/srv/fai/nfsroot";
06     host demohost {
07         hardware ethernet 00:e0:81:5c:aa:82;
08         fixed-address 192.168.1.25;
09     }
10 }
```

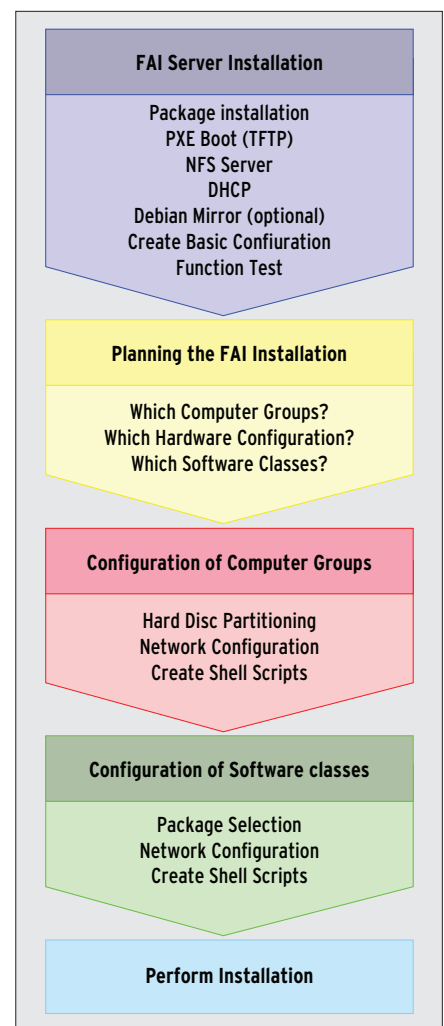


Figure 2: An FAI installation requires careful planning.

Listing 2: TFTP Configuration

```
01 # Select boot behavior
02 # (please comment out one of the following two lines)
03 default fai-generated
04 # default localboot
05
06 # Network boot and FAI start
07 label fai-generated
08 kernel vmlinuz-2.6.25-2-486
09 append initrd=initrd.img-2.6.25-2-486 ip=dhcp \
10     root=/dev/nfs nfsroot=/srv/fai/nfsroot boot=live \
11     FAI_FLAGS=verbose,sshd,createvt
12 ACTION=install
13
14 # Boot from hard disc
15 label localboot
16 localboot 0
```

specify the mirror from which to download the packages to create the NFS root.

Using Debian Standards

FAI uses the normal Debian Apt system, which lists its repositories in `/etc/apt/sources.list`. FAI adds the distributor's standard mirrors and the mirror at Cologne University to this list. If you prefer to use a different set of mirror servers, you need to modify the file. To avoid unnecessary complications, let FAI create the NFS root with the same version of Debian you will be using for the clients.

After correctly configuring all of these parameters, the next step is to create the NFS root directory. To do so, run the `fai-setup -v` command. After downloading some 330MB of package data, `/srv/fai/nfsroot/live/filesystem.dir` will contain the NFS root. The script also exports the directories required for the client installation on NFS and drops a matching kernel image into the `/srv/tftp` directory.

Test Drive

To test the server, try a single client installation. You can use the pre-defined `DEMOHOST` package class for this. It contains a minimal system with client hardware requirements to match: A standard PC with at least 500MB hard disk space and (to keep things simple) a PXE-capable network adapter should suffice.

All the required network services, such as the nameserver and NFS, should be installed on the FAI server by now. To help the clients boot, set up an `/etc/dhcp3/dhcpd.conf` file similar to that

shown in Listing 1 on your DHCP server. Then assign an IP address and a DNS name to the MAC address and specify the TFTP image for booting the client.

The client's IP address needs to be in hexadecimal notation: COA80119 for 192.168.1.25 in this case. Then create a `/srv/tftp/fai/pxelinux.cfg/COA80119` file with the contents of Listing 2. The `default` setting controls the client's boot behavior; start by selecting `fai-generated`. It is a good idea to use the `fai-chboot -Ifv demohost` command to create the file; this will automatically add the current values. After booting the client, FAI takes over the helm and installs a basic Debian system without a GUI. After the installation is complete, you can change the boot behavior for `fai-generated` to `uf localboot` and relaunch the client, which is now ready to run.

Besides the official Debian mirrors, FAI will also work with local mirror servers. If you have a large number of clients and have to install regularly, you will appreciate the benefits of your own mirror, in that it avoids the need for FAI to download the packages time and time again through your Internet connection. Additionally, a local installation over Fast or gigabit Ethernet will take less time.

Another benefit is that you can freeze the content of your local mirror. This makes the local installation independent of the current repository state and guarantees that all your clients have identical software, no matter when you install them. Also, you can add a repository on the mirror server to distribute packages you have built yourself.

After installing the `demohost` testing client to test FAI's functionality, you can go on to plan additional installations. Follow the FAI motto: "Plan your installation, and FAI installs your plans."

Working with Classes

FAI classes provide a versatile means for assigning packages and settings to client systems. For example, the Computer Science department at Aalen University, Aalen, Germany, organizes computers into pools and labs, with identical machines in many rooms. Some labs have computers with different architectures, so a separate kernel with hardware-specific drivers for each computer type is needed.

The configuration is based on a common class for each room and a class for each computer type. Because FAI also is

Table 1: The Class Concept

Class	Short Form	Example	Explanation
Room class	RC	RC_L024	Room class Labor 0.24
		RC_P130	Room class Pool 1.30
		RC_O227	Room class Office 2.27
Computer class	CC	CC_SUN	Computer class Sun computers
		CC_HPCOMPAQ	Computer class HP-Compaq computers
		CC_VMWARE	Computer class for VMware machines
Basis class	BC	BC_LINUX	Basic class for all FAI clients
		BC_KDE	Basic class for KDE
		BC_BASE	Basic class for pool and lab machines
Service class	SC	SC_LDAP	Service class for LDAP-based authentication
Program class	PC	PC_KIOSKTOOL	Program class for the Kiosktool program
		PC_BOINC	Program class for the Boinc service

also used to install VMware machines, a separate class for virtual machines is needed. In some rooms with a variety of computer architectures, the admins use a combination of room and computer classes.

Common sets of software applications can define different software class categories – for example, a basic class (BC), a service class (SC), and a program class (PC). The basic class includes the basic Debian system, programs, configuration files, and scripts that are valid for all pool and lab machines. The basic GUI client classes are *BC_X11*, *BC_KDE*, and *BC_GNOME*. A service class contains the information needed to prepare a client for a specific service. For example, the *SC_LDAP* class contains all the packages and configuration data required to access the faculty's own OpenLDAP directory service.

Many student projects require the installation of programs on a group of computers. One approach to handling this is to define a separate program class for each project.

Classes also can define a common system configuration. For example, you could disable the KDE screen-locking mechanism for all the computers in the pool. These settings are typically supported by an entry in an existing file or by a script that adds the entry. If settings of this kind are necessary, you can add another class to the machines.

Assigning programs to classes of their own also gives you an easy way to enable or disable them. This said, it is better not to define too many program classes or you will lose track of them. Table 1 provides an overview of defined classes with examples.

The */srv/fai/config/* configuration directory and its subdirectories contain the

Listing 3: 50-host-classes Structure

```
01 #!/bin/bash
02 # assign classes hosts
03 case $HOSTNAME in
04   # L0.24 - CC_SUN (computer class _ sun)
05   L024-*)
06     echo "BC_LINUX CC_SUN BC_X11 BC_KDE BC_BASE RC_L024 SC_LDAP" ;;
07   # Linux Debian VMware Class: CC_VMWARE
08   vmware-1|vmware-2)
09     echo "BC_LINUX CC_VMWARE SC_LDAP" ;;
10   # default class
11   *)
12     echo "BC_LINUX" ;;
13 esac
14 (ifclass I386 || ifclass AMD64) && echo GRUB
15 exit 0
```

THE MATHEMATICS OF HUMOUR

TWELVE Quirky Humans,
TWO Lovecraftian Horrors,
ONE Acerbic A.I.,
ONE Fluffy Ball of Innocence and
TEN Years of Archives
EQUALS
ONE Daily Cartoon that Covers the
 Geek Gestalt from zero to infinity!

Over Two Million Geeks around the world can't be wrong!
COME JOIN THE INSANITY!



UserFriendly.Org

Listing 4: BC_LINUX.var Parameters

```

01 # Allow unsigned repositories:      10 time_zone=Europe/Berlin
02 FAI_ALLOW_UNSIGNED=1                11
03                                     12 # Call "openssl passwd -1" to create
04 # German keyboard:                  13 # for the new system; supported
05 FAI_KEYMAP=de-latin1-nodeadkeys     14 formats are md5 and crypt:
06                                     15 rootpw='$1$865YxXkZ$xRxd/
07 # UTC=yes, if the system clock is set  WUPLPU4gZfVmilax1'
   to UTC.                             16 servicepw='$1$arm/
08 UTC=yes                             urPR$5w0Q7Z3i2cRSq1t0q/zyK/'
09 TIMEZONE=Europe/Berlin

```

information FAI needs to install a client. Each subdirectory has a different task. The *class/* subdirectory contains the files and scripts that define the individual classes and variables, like *50-host-classes*, in which the administrator stores the classes assigned to each client group.

Listing 3 is an example of the *50-host-classes* group. The file is a shell script that displays the class names depending on the *\$HOSTNAME* variable. If the client's hostname starts with, say, *L024*, it is a computer in lab L0.24. FAI will install the basic classes *BC_LINUX*, *BC_X11*, *BC_KDE*, and *BC_BASE*; the computer class *CC_SUN*; the room class *RC_L024*; and the service class *SC_LDAP* on this client type. If FAI identifies the client as *vmware-1* or *vmware-2*, it is a virtualized machine. In this case, FAI installs

the classes *BC_LINUX*, *CC_VMWARE*, and *SC_LDAP*.

Local Environment

For each class, you can define defaults for the keyboard layout, time zone, user passwords, or similar settings. The values are stored in the *classname.var* file. Listing 4 is a template for the *BC_LINUX* class. FAI parses the list of classes defined in *50-host-classes* from left to right. If multiple classes define the same variable, the rightmost value in the list prevails. Clever administrators replace generic with specific values this way.

If the computers in Lab L0.24 are assigned the *BC_LINUX* class settings with different passwords, you can create a new *RC_L024.var* file with lines 12 through 15 of Listing 4 with a new password. Because *50-host-classes* lists the

Listing 5: Package Configuration in BC_LINUX

```

01 # packages for all systems
02 PACKAGES aptitude
03 alsa-base
04 alsa-utils
05 discover
06 ...
07
08 PACKAGES aptitude BC_KDE
09 firefox-locale-de-de
10 thunderbird-locale-de

```

RC_L024 class after the *BC_LINUX* class, the details in the new file prevail.

The *debconf/* subdirectory is used for the class-independent configuration of the Debconf system parameters. Debian uses *debconf/* to store global settings such as the default editor or the default web browser. Storage media are configured in the *disk_config* subdirectory (see the "Enhanced Partitioning" box).

The *package_config/* directory defines the packages required by the target system for each class, usually with the *PACKAGES aptitude package name...* command. De-installation is also supported by the *PACKAGES remove package name...* command. Listing 5 shows how to define dependencies between classes. The *PACKAGES aptitude BC_KDE* com-

Enhanced Partitioning

Earlier versions of FAI used the *setup_harddisks* script to partition the client hard disk. The tool works fine with normal hard disks, but it does not support RAID systems or Logical Volume Manager (LVM2). As of version 3.2.8, FAI provides the *setup-storage* tool as an alternative to *setup_harddisks*.

If FAI finds a *USE_SETUP_STORAGE=1* line in a configuration class, it uses *setup-storage* to partition hard disks instead of *setup_harddisks*. Unfortunately, the tool is not downwardly compatible; its configuration file syntax has changed.

Listing 6, a sample configuration for *setup-storage*, partitions the first hard disk on a dual-boot system, reserving the first partition for Windows XP. Each line of the script contains a command with matching options. The *disk_config* command defines global settings for partitioning the whole hard disk. The *disk 1* option selects the first hard disk. If this is an IDE hard disk, *setup-storage* uses the internal partition

designator *hda*; this is *sda* for SATA or SCSI disks.

The *preserve_always:1* option tells the partitioning tool not to modify the first primary partition, thus leaving the Windows installation in place. *setup-storage* will delete all the other partitions. The *bootable:2* option enables the second partition as the boot partition. The *primary* and *logical* commands tell *setup-storage* to create a primary or logical partition. The partition numbers depend on the order in which they are listed in the file.

Line 2 of Listing 6 uses the *primary* option to tell *setup-storage* to create an *sda2* partition with a size of 200MB. The partition contains an ext2 filesystem and uses */boot* as its mount point. The first *logical* command in line 3 creates a swap filesystem with a size of 4,000MB in the first logical partition, *sda5*. The logical partitions *sda6* and *sda7* for */tmp* and */var* follow. The last command creates the *sda8* partition. The

3000- parameter tells the program to use the remaining disk space, but with at least 3,000MB for the partition. In case of insufficient disk space, the installation will display an error message and quit. FAI identifies the partition names autonomously; the administrator only needs to say whether they are primary or logical partitions.

Listing 7 is an example of partitioning a RAID 1 with two SCSI disks. First, *setup-storage* partitions the *sda* and *sdb* disks identically but does not assign the filesystem or mount points. Commands in the *disk_config raid* block tell *setup-storage* to set up a RAID system with three partitions and format the partitions as specified.

The *setup-storage* tool is a very powerful utility with far more functionality than *setup_harddisks*. The configuration command syntax is specified by EBNF rules. These rules and more information on using *setup-storage* are available from the man page and the FAI wiki [2].

mand installs the named packages only if they are part of the *BC_KDE* class, which lets administrators design and implement highly granular structures.

Your Own Taste

The *files/* directory stores by class all the files that FAI will copy to the target system. The hierarchy below *files/* is based on the Debian directory tree. To illustrate this point: If you select the *CC_HPCOM-PAQ* class, you will expect to find the *menu.lst* file in the client's */boot/grub* directory later. For this to happen, you need to copy the contents of *menu.lst* into the *files/boot/grub/menu.lst/CC_HPCOMPAQ/* directory. FAI does not automatically copy user-defined files below *files/* to the target system; this only happens if the configuration script contains an *fcopy* command.

The command

```
fcopy -iM /boot/grub/menu.lst
```

copies the GRUB bootloader configuration file *menu.lst* to the client. It is part of the *BC_LINUX* class.

The */hooks* directory contains programs or scripts for customizing the installation process. Finally, the *scripts/* directory contains the scripts that FAI runs during the install. You can easily use *fcopy* to copy files on the basis of the client class or as a local command with the *{ROOTCMD}* command.

In typical Unix style, FAI offers a number of scripting options. Besides legacy shell scripts, FAI will also run Perl or

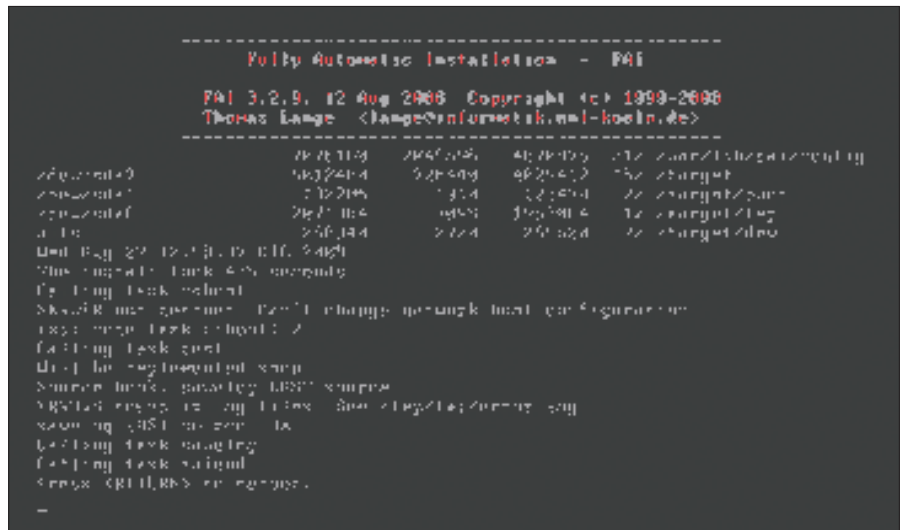


Figure 3: At the end of the installation, FAI outputs a summary of installation information. Pressing Return restarts the new Debian system.

Cfengine scripts. Cfengine is a good choice for modifying configuration files.

Diagnostics

After planning, collating, and configuring installation files and scripts, you are finally ready to use FAI to automate the installation process on a large number of computers. The last step is to extend the DHCP and TFTP configurations in a way that will allow the new clients to boot off the network and automatically launch into the FAI installation. To do so, you just need to apply the settings for the first test run. Once the FAI client has completed the installation, it will display a summary onscreen and wait for somebody to press Return to trigger a reboot (see Figure 3).

The FAI deployment framework stores all installation processes in logfiles on the server. If necessary, you can evaluate the files in the */var/log/fai/* directory and check them for errors.

Remote Control

FAI is a versatile tool for automating the installation of DEB-based Linux distributions. In addition to x86, the automatic installer also supports Sparc and PowerPC architectures. The FAI installation solution is ideal for many situations, from computer clusters to heterogeneous IT infrastructures in enterprises. Once you have negotiated the learning curve, FAI provides a powerful tool capable of automating almost any Debian-based installation. ■

Listing 6: Hard Disk Partitioning

```
01 disk_config disk1 preserve_always:1 bootable:2
02 primary /boot 200 ext2 rw
03 logical swap 4000 swap rw
04 logical /tmp 10000 ext2 defaults
05 logical /var 5000 ext3 defaults
06 logical / 3000- ext3 defaults,errors=remount-ro
```

Listing 7: Partitioning for a RAID 1 System

```
01 disk_config sda
02 primary - 256 - - 10
03 primary swap 1024 swap sw
04 primary - 0- - -
05
06 disk_config sdb
07 primary - 256 - - 10
08 primary - 1024 - -
09 primary - 0- - -
10
11 disk_config raid bootable:1
12 raid1 /boot sda1,sdb1 ext2
    rw,errors=remount-ro
13 raid2 swap sda2,sdb2 swap rw
14 raid3 / sda3,sdb3 ext3 rw
```

INFO

- [1] Fully Automatic Installation (FAI):
<http://www.informatik.uni-koeln.de/fai/>
- [2] FAI wiki entry on *setup-storage*:
<http://faiwiki.informatik.uni-koeln.de/index.php/Setup-storage>

THE AUTHORS

Christoph Karg is a professor of computer science at Aalen University, Germany, and has been interested in Linux and open source since his days as student.

Steffen Bornemann has worked for Aalen University as a computer scientist for three years. At the Faculty of Electronics and Computer Science, he is responsible for managing Linux computers and the Linux network.