FAI: Fully Automatic Installation of Debian GNU/Linux

Niall Young <niall@holbytla.org>

Debian mini-conf

@Linux.conf.au 2003

What is the problem?

Linux installation and configuration is:

- Time intensive
- Interactive
- Repetitive
- Boring

i.e. a complete waste of our precious time

- The majority of servers in a managed environment are 90% identical, only the applications differ once the core utilities and network transports have been chosen

- Why sit there manually partitioning disks, installing packages and configuring the system when you already know exactly how you want it setup? Why perform the work of a monkey?

FAI

What is FAI?

- FAI is the “Fully Automatic Installation” for Debian GNU/Linux
- Its goal is to perform non-interactive, unattended Debian GNU/Linux installations
- It’s modular and very extensible, can be customised for any host or site

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

How does it work?

Basic process flow:

- **FAI boots via Floppy or PXE capable Ethernet card**
- **Retrieves IP configuration from DHCP/BOOTP**
- **Mounts an NFSROOT**
- **Partitions disks, creates filesystems**
- **Installs Debian packages non-interactively**
- **Performs any required configuration**
- **Finish/Reboot***

All in about 3-5 minutes!

FAI Class Concept

FAI classes are similar to task-* packages, but provide more than just a list of dependencies.

They represent extra functionality/configuration information above and beyond the base Debian install

e.g. class SCSI may provide the scsitools package, extra kernel modules

Class X11 may provide the xserver-xfree86 and sawfish packages, conffile changes

Class HOSTNAME may provide all of the above and more, or less, whatever you want!

Each host may belong to multiple classes, e.g.:

Host1 (SCSI)

Host2 (SCSI, X11, MULTIMEDIA)

Host3 (Host3)

Class definitions may consist of:

- Partition and filesystem information
- A list of packages to install or remove
- A set of scripts to do stuff*
- Extra files to install
- Hooks to override default FAI behaviour

Classes can be statically defined or dynamically generated on the fly, e.g.:

Detect/kudzu may detect a DAC960 RAID card -> add Class RAID to the installation

Host1 (SCSI) -> Host1 (SCSI, RAID)

Classes may potentially do anything!

Shell/Perl/Expect/Cfengine scripts may perform extra configuration steps, define more classes on the fly, perform a remote query to retrieve configuration information, anything!

Class Hooks may override default FAI behaviour, e.g.:

A diskless workstation will fail partition/filesystem creation, so FAI allows any class to re-define an FAI component or skip it entirely

Example Class

```
cat disk_config/EXAMPLE

# <type> <mountpoint> <size in mb> [mount options]     [;extra options]

disk_config hda

primary  /fai-boot      2       rw,errors=remount-ro    ; -c
primary  /              50      rw,errors=remount-ro    ; -c
logical  swap           200
logical  /var           200
logical  /usr           1500    rw
logical  /tmp           300     rw,nosuid               ; -m 1
logical  /home          700     rw,nosuid               ; -m 0
logical  /files/scratch 0-      rw,nosuid               ; -i 50000 -m 1
```

```
cat files/etc/apt/sources.list/EXAMPLE

deb http://ftp.iinet.net.au/debian/debian woody main
  contrib non-free
deb http://ftp.iinet.net.au/debian/debian-non-US
  woody/non-US main contrib non-free
deb http://ftp.iinet.net.au/debian/debian-security
  woody/updates main contrib non-free
```

```
cat hooks/partition.EXAMPLE

#! /bin/sh

# install a diskless workstation
# this is just an example, but not perfect code
#
# (c) Thomas Lange, 2001, lange@debian.org

…

# now skip some default tasks
skiptask partition mountdisks
```

```
cat package_config/EXAMPLE

PRELOADRM
   http://www.location.org/rp8_linux20_libc6_i386_cs1_rp
   m /root

PACKAGES taskinst
german science

PACKAGES install
adduser netstd ae less passwd realplayer

PACKAGES remove
gpm xdm

PACKAGES dselect-upgrade
ddd install a2ps install
```

```
scripts/EXAMPLE/*

S01_foo
…
S99_bar
```

What if you have no NFS or network connectivity at all?

FAI BootCD

- Inspired by other attempts, particularly Marc Schaefer’s
- Goals: self contained with package repository, as few changes to FAI itself as possible

http://www.iinet.net.au/~niall/fai/

A single CDROM could contain a complete FAI Server along with Class definitions, all required Debian packages and (most likely) room to spare.

Why a CDROM?

- Bootable
- Well supported
- Popular
- Read-only

How does it work?

“El Torito” bootable CD – pretends to be a Floppy drive and acts as the boot media

- Loads a boot Floppy image from CD
- Loads bootloader (Grub*/Syslinux)
- Loads a Linux kernel with initrd
- Initrd detects CDROM hardware, locates BootCD media and mounts
- Initrd switches to CD as root filesystem before kernel finishes loading
- Starts a regular FAI installation off CD
- Finish/Reboot*

Why Grub?

- Grub supports “fallback” – if default boot option fails it will try a second, e.g. try HD, if that fails try CD, install, reboot, HD succeeds. Potentially leave CD inserted indefinitly*
- Supports ext2 natively, kernel change without modifying MBR, uses absolute filename to kernel on disk
- Device naming – hd0 is first HD, SCSI/IDE*

Why an initrd?

- We need to find the right CDROM and mount in a known location before regular SysV init
- We don’t know where CDROM is, could be IDE/SCSI, Master/Slave, start of chain, end of chain, multiple CDROM drives…
- We may need to load extra kernel modules

Demonstration

Success!

✓Self-contained

✓0 changes to FAI

✓Read-only media – increased security

✓Bloody useful :-)

Live CDs

There are many other LiveCD projects:

- Linux distro CDs
- CommunityWireless.org
- MoviX
- “Firewall in a box” products
- Knoppix

Knoppix Demonstration

- LiveCDs act like a read-only “console”
- Think of Nintendo/Playstation consoles: fixed hardware, read-only Application content

- Why not GNU/Linux consoles/appliances?

GNU/Linux appliances:

- Linux on PS2/X-Box
- MAME/OpenGL gaming
- MP3/DivX multimedia box
- Wireless AP
- Caching appliance
- Router
- Home Gateway

Build new content as a functional FAI Class, optimise it, then burn to CD.

- Re-use same software and development process regardless if aimed for disk/CD

But traditional appliances have fixed hardware!

- We don’t, but we can make reasonable assumptions
- We can abstract hardware layers and use APIs, virtual machine etc.
- Worst-case: define a minimum spec

Tying it all Together

FAI – Debian GNU/Linux replication, potentially describe how to build *any* Debian system perfectly

LiveCDs – optimised appliance

What if we combine the two? :-)

e.g. Define FAI_SERVER and APPLICATION_X

Install both to a LiveCD:

- Boot with APPLICATION_X functionality; or
- Install APPLICATION_X functionality to disk using FAI_SERVER on CD!

i.e. Reproduction

But they’re static, no bio-diversity!

No, just add a CD burner…

- Boot APPLICATION_X
- Install to disk
- Upgrade/modify
- Create new ISO
- Burn to CDROM

i.e. Evolutionary reproduction

So what?

- Rapid deployment of Debian GNU/Linux
- Disaster recovery
- Spend more time coding and creating content
- Framework for common Debian GNU/Linux solutions, minimise duplication of effort
- Gather splintering child distros and projects back into Debian main?

TODO

- Class content creation
- Modularise Class definition so they’re portable, share via p2p, uDeb packages
- Merge FAI functionality and Class concept into new debian-installer, imagine if every Official CD did this – just add a local configuration floppy with your Classes
- More generic, mature LiveCD toolkit
- Simplify Class creation and customisation

Thankyou!

FAI

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

FAI BootCD

http://www.iinet.net.au/~niall/fai/

niall@holbytla.org