(re)discovering the Debian Installer

Cyril Brulebois
kibi@debian.org

Paris Mini-DebConf, november 2012
Foreword

Many options to install Debian!

Using the Debian Installer

How do we build it?

How do we release it?

Future plans
Foreword

Many options to install Debian!

Using the Debian Installer

How do we build it?

How do we release it?

Future plans
Some terminology

- d-i: Debian Installer
- g-i: Graphical Installer (using X.org since squeeze)
- udeb: $mu$deb, with relaxed policy
  - no copyright
  - maint scripts: postinst at most
  - config/templates
  - possibly built against a limited number of libraries
  - heavily compressed (xz) by default, thanks to debhelper
Foreword

Many options to install Debian!

Using the Debian Installer

How do we build it?

How do we release it?

Future plans
Supported architectures in squeeze

- amd64
- armel (+ armhf in wheezy)
- kfreebsd-i386
- kfreebsd-amd64
- i386
- ia64
- ia64
- mips
- mipsel
- powerpc
- s390 (+ s390x in wheezy)
- sparc
Image type:

- **netinst**: small CD
  - special case: *multi-arch* (amd64/i386)
  - mini-CD (businesscard): dropped for wheezy
- **CD/DVD**
  - download: HTTP/FTP, jigdo, BitTorrent
  - alternate desktops for CD#1: KDE, LXDE, Xfce
- **Blu-ray** (jigdo only)
- **netboot / PXE**
  - for network-based boot
  - **not** the same as netinst
  - frequent kernel/module version mismatch during development!
Options/availability depending on:

- image type
- architecture (e.g. no CD for s390/s390x)
- considered build:
  - stable (official releases, e.g. squeeze 6.0.6)
  - testing (official releases, e.g. wheezy beta 4)
  - testing (daily/weekly builds)
Foreword

Many options to install Debian!

**Using the Debian Installer**

How do we build it?

How do we release it?

Future plans
Modularity

Installation interface:

▶ text-based install (terminal)
▶ graphical install (X.org)
▶ text-based install, with speech synthesis (s, Enter)

Advanced options:

▶ expert install
▶ rescue mode
▶ automated install: preseed
▶ desktop environment choice
Contents of an installation image:
- a bootloader
- documentation
- kernel(s)
- initramfs(es)
- data: debs + udebs
What happens under the hood? (2/2)

How does the installer work?

1. show a bootloader
2. user picks an interface, options, etc.
3. start a debconf frontend
4. user answers lots of questions
5. udebs are loaded on the fly, more questions are asked
6. famous question: which tasks to install?
7. install install install, eject, reboot, done
Demonstration-time!

Selected environment:
- virtualbox
- netinst i386 image, official wheezy beta 4 build

Two features:
1. graphical automated installation, started with:
   url=http://10.0.2.2/~kibi/preseed.cfg
2. rescue mode, trying to save an encrypted LVM setup (removed/broken cryptsetup, oops!)
Foreword

Many options to install Debian!

Using the Debian Installer

How do we build it?

How do we release it?

Future plans
Starting from the venerable svn...

$ svn ls svn://svn.debian.org/svn/d-i/trunk
  .mrconfig
  README
  manual/
  packages/
  scripts/

▶ manual/ → installation-guide source package
▶ packages/ → translation efforts (only contains po/)
▶ .mrconfig → helps keep all repositories up to date

Initial checkout:

mr bootstrap http://anonscm.debian.org/viewvc/d-i/trunk/.mrconfig?view=co d-i
... to many git repositories

At the top-level:

▶ di-autobuild/ → automation for d-i builds (on buildds) [git]
▶ installer/ → debian-installer source package [git]

Under packages/:

$ ls -d packages/*/  
packages/aboot-installer/
packages/anna/
packages/apt-setup/
packages/arcboot-installer/
packages/auto-install/
...

→ one repository per source package [git]
→ each can produce deb and/or udeb packages
udeb-producing packages (1/2)

- packages maintained by debian-boot
  - anna-udeb
  - cdebconf-udeb
  - grub-installer
- Linux & FreeBSD kernels: kernel-wedge
  - input-modules-version-di
  - usb-modules-version-di
- core packages:
  - e2fsprogs-udeb
  - gnupg-udeb
  - util-linux-udeb
udeb-producing packages (2/2)

- GTK+ stack:
  - libcairo2-udeb
  - libpango1.0-udeb
  - libgtk2.0-0-udeb
- X.org stack:
  - many libx*-udeb packages
  - a generic input driver: evdev
  - a generic video driver: fbdev
  - a minimal X server
- accessibility stack:
  - brltty
  - espeakup
The debian-installer source package: layout

- **build/** → heavy work
  - Makefile → entry-point for building everything
  - boot/ → arch-specific conf. for boot loaders
  - config/ → arch-specific variables, calls to various utils, ...
  - localudebs/ → local dropzone!
  - pkg-lists/ → what do we need for each arch and image?
  - util/
- **debian/** → *almost* straightforward packaging
  - unstable or UNRELEASED as target → small differences
  - debian-installer-images_20121114_amd64.tar.gz
- **doc/** → development, i18n, talks
The debian-installer source package: machinery

What happens in build/?

- Use sources.list for udebs:
  
  ```
  deb http://mirror/debian wheezy main/debian-installer
  ```

- Support for local udebs is present

- `util/get-package` is used to:
  - fetch udebs, resolving dependencies
  - install them into a rootfs

- Call more tools to handle partitioning etc.
Foreword

Many options to install Debian!

Using the Debian Installer

How do we build it?

How do we release it?

Future plans
Many people to talk to/work with

- individual package maintainers/packaging teams
- debian-boot
- debian-release
- debian-cd
- ftpmasters
- debian-live
Process outline

1. upload of udeb-producing packages to unstable
2. test of daily builds
3. manual coordination with RT to migrate the udebs to testing
4. upload of debian-installer to unstable
5. ftpmasters: dak copy-installer
6. RT: urgent debian-installer
7. CD/DVD images built using debian-cd
8. cdimage.debian.org: last-minute checks
9. release announce: mail to dda@/dd@, and website update

Timing:
- 3 weeks for development (before step 4 happens)
- 1 week for releasing
- 1 week for the unexpected
Foreword

Many options to install Debian!

Using the Debian Installer

How do we build it?

How do we release it?

Future plans
Future plans

Release candidate 1
- Target: December/January
- Support for Versatile Express (armhf in qemu)
- Lots of small bug fixes in various areas
- Possibly more EFI bug fixes

Long term:
- What to do with Secure Boot?
- Regression testing using VMs
- What do you want to see implemented in d-i?
Thanks!

In addition to everyone involved:

- Previous d-i release managers for their hard work
- Christian Perrier, l10n coordinator
- CD team: Steve McIntyre
- All installation-reports submitters!

On a personal note:

- Josselin Mouette, for his great *Please save the graphical installer* blog post
Appendix: preseed.cfg

d−i keyboard−configuration/xkb−keymap select fr

d−i debian−installer/locale string fr_FR

d−i mirror/country string manual

d−i mirror/http/hostname string 10.0.2.2:9999

d−i mirror/http/directory string /debian

d−i partman/auto/method string crypto

d−i partman/lvm/device_remove_lvm boolean true

d−i partman/lvm/confirm boolean true

d−i partman/lvm/confirm_nooverwrite boolean true

d−i partman/auto/choose_recipe select atomic

d−i partman/choose_partition select finish

d−i partman/confirm boolean true

d−i partman/confirm_nooverwrite boolean true

d−i cdrom−detect/eject boolean false

d−i passwd/make−user boolean false

d−i passwd/root−password password root

d−i passwd/root−password−again password root

d−i finish−install/reboot_in_progress note

d−i partman−crypto/passphrase password root root

d−i partman−crypto/passphrase−again password root root

d−i partman−crypto/weak_passphrase boolean true