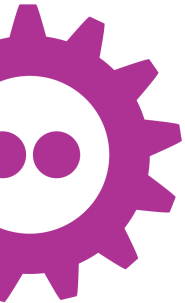
COLLABORA



Managing build infrastructure of a Debian derivative

Andrej Shadura

4 February 2018

FOSDEM 18

COLLABORA

Presentation Outline

Who am I

Enter Apertis

Build infrastructure

Packaging workflows

Image builds

Andrej Shadura

- contributing to Debian since 2007
- Debian Developer since 2013
- working for Collabora since 2015

COLLABORA

Andrej Shadura

- contributing to Debian since 2007
- Debian Developer since 2013
- working for Collabora since 2015

- doing packaging since 2010
- never ran any ‘real’ Debian infrastructure
- only used mini-dinstall to publish packages

Presentation Outline

Who am I

Enter Apertis

Build infrastructure

Packaging workflows

Image builds

Apertis

- Debian derivative tailored for automotive needs
- originally developed for infotainment systems
- fit for a wide variety of electronic devices

Apertis

- code hosting
- code review tools
- package build services
- image generation services
- automated testing infrastructure

Apertis

- based mostly on Ubuntu
- takes several packages directly from Debian
- provides its own software packages, frameworks, APIs

Apertis

- systemd for application lifecycle tracking
- AppArmor for policy enforcement
- Wayland for graphics
- GStreamer for multimedia playback

Apertis + OSTree + Flatpak = ♥

- Apertis app bundle format based on Flatpak
- original system updater used btrfs to ensure atomicity and enable recovery mode
- OSTree-based solution replaces an older btrfs-based updater

Apertis + OSTree + Flatpak = ♥

Drawbacks of the btrfs updater:

- significant maintenance effort
- bootloader doesn't support btrfs
  can't use on `/boot`
- requires initramfs
- directly manipulates btrfs volumes
  unsuitable for e.g. unprivileged containers
- testing is difficult
- btrfs itself has a lot of issues

Apertis + OSTree + Flatpak = ♥

OSTree:

- works with any filesystem
- stores multiple trees in the same repo
- no need for extra partitions for safe upgrades — saves space
- less custom code
- works better with containers
- a full solution on its own, not just a building block

Why Ubuntu+Debian, not just Debian?

- the Universal operating system
- composed entirely of free software
- developed by a community of individuals

Why Ubuntu+Debian, not just Debian?

- `stable` moves too slowly, changes between releases are quite significant
- `unstable` breaks things a bit way too often
- until recently, releases had unpredictable timing, there wasn't an LTS release

Why Ubuntu+Debian, not just Debian?

Advantages:

- large install base, so despite more frequent releases there's still a lot of testing
- Ubuntu is an upstream for AppArmor, on which we rely
- regularly timed releases — and LTS ones too

Why Ubuntu+Debian, not just Debian?

Downsides:

- being derivative of a derivative complicates upgrades/rebases
- we don't **really** need all patches Ubuntu applies (Mir, anyone?)

COLLABORA

Presentation Outline

Who am I

Enter Apertis

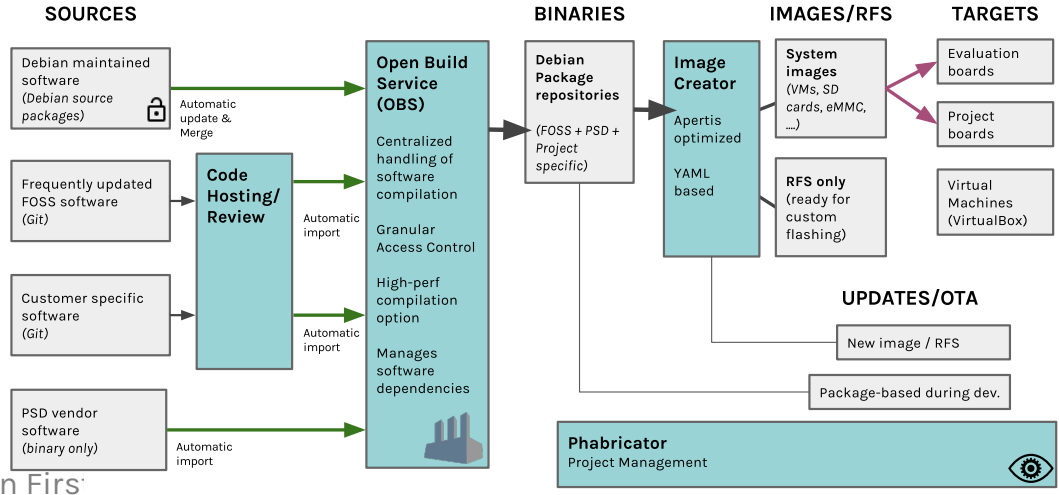
Build infrastructure

Packaging workflows

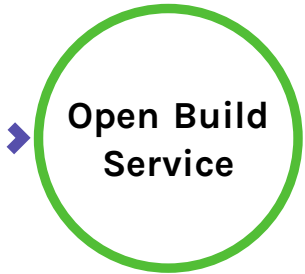
Image builds

COLLABORA

Overview of the Apertis infrastructure



OBS



- builds packages in a fresh chroot every time
- all source packages are revision controlled
- provides fine-grained access control
- provides Subversion-style branching and merge review system

For more information on OBS, please watch a talk by Andrew Lee at FOSDEM 2018: https://col.la/fosdem18obs

OBS: components split

Component	Description
apertis:18.03:target	Apertis target
apertis:18.03:development	Apertis development
apertis:18.03:helper-libs	Apertis helper libraries
apertis:18.03:hmi	Apertis human machine interface packages
apertis:18.03:sdk	Apertis SDK
apertis:18.03:snapshots	Apertis git snapshots

Supported architechtures: amd64, armhf, arm64.

OBS: components split

Component	Description
apertis:18.03:target	Apertis target
apertis:18.03:development	Apertis development
apertis:18.03:helper-libs	Apertis helper libraries
apertis:18.03:hmi	Apertis human machine interface packages
apertis:18.03:sdk	Apertis SDK
apertis:18.03:snapshots	Apertis git snapshots

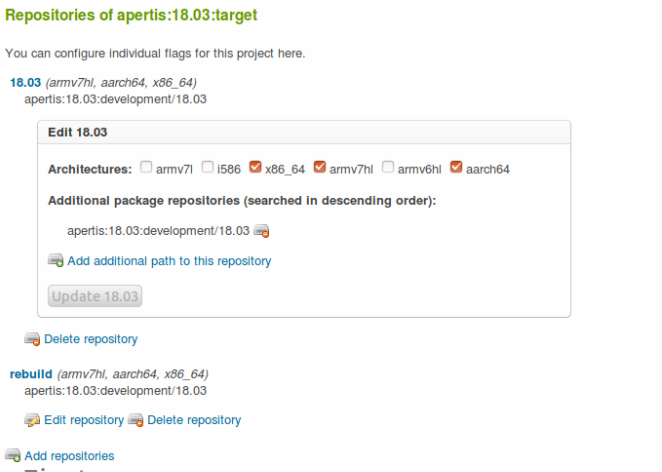
Supported architechtures: amd64, armhf, arm64.

OBS: how the packages are published

- OBS maintains internal package repositories, usually one per project
- internal repositories aren't in APT format, so `reprepro` is used to make packages available to APT
- it easy to do full rebuilds using more than one repository per project:
  - add one more repository per project, `rebuild`
  - depends on the main repository, non-publishing

COLLABORA

OBS: how the packages are published



Packaging workflows: Ubuntu/Debian packages

- packages with no changes are imported unchanged directly to OBS
- packages with minimal changes are maintained in OBS
  - mostly custom DEP-3 patches applied on top of the sources from Ubuntu
  - local changes get `co1` version suffixes
- a fork of Ubuntu's Merge-o-Matic called Merge-our-Misc, is used to pull new updates from Ubuntu LTS
- select packages are kept in Git

Packaging workflows: Git

Non-Apertis packages:

- DEP-14:
  - `upstream/{latest,$version}` branches with the upstream project code
  - `apertis/{master,$distro}` branches for Apertis packaging
  - `apertis/$version` tags (eg. `apertis/2.48.0-1ubuntu4co1`)
- `git-buildpackage` and `gbp-pq` to manage patches.

Packaging workflows: Git

Native Apertis packages:

- `master` branch contains the sources for the current development release
- packaging metadata is also kept in `master`, not on a separate branch
- `$distro` branches (for instance, `17.12`) contain the sources for past distro releases
  - `$version` tags for 'upstream' source releases (eg. `0.1803.1`)
  - `apertis/$version` tags for packaging releases (eg. `apertis/0.1803.1co1`)

Packaging workflows: Git

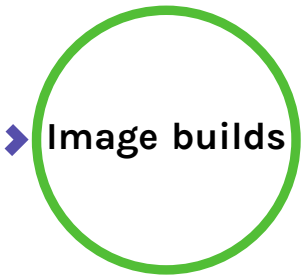
Apertis Jenkins instance:

- builds every new commit in a controlled environment
- if the build succeeds, the source package is submitted to OBS for a clean rebuild in the `:snapshots` component
- for commits tagged as releases, Jenkins creates merge requests for the main component (`:target`, `:development` etc).

Packaging workflows: Git

- `build-snapshot` script by Simon McVittie is used by Jenkins to build packages and create source packages to be uploaded to OBS
- new patches submitted for review at Phabricator get built on top of the branch they apply to

Image builds

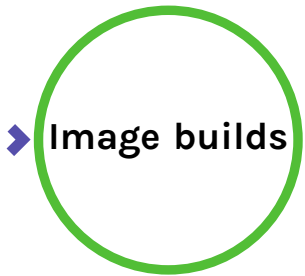


- Jenkins
- Linaro image tools

- multi-stage process
- separation of hard-ware-dependent and hardware-indepen-dent components

COLLABORA

Image builds



- Jenkins
- ~~Linaro image tools~~
  debos!
- multi-stage process
- separation of hardware-dependent and hardware-independent components

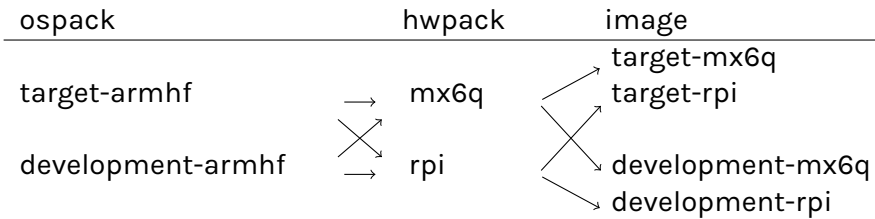
COLLABORA

Image builds

ospack	hwpack	image
target-armhf	mx6q	target-mx6q target-rpi
development-armhf	rpi	development-mx6q development-rpi

COLLABORA

Image builds



debos: Debian OS builder

```
- action: debootstrap
  suite: "stretch"
  components:
    - main
  mirror: https://deb.debian.org/debian
  variant: minbase
  merged-usr: true
```

debos: Debian OS builder

```
{{- if eq $type "target" "development" "sdk" }}
- action: apt
  description: "Target packages"
  packages:
    - apertis-target
    - apertis-hmi
    - libegl1-mesa-drivers
{{- end -}}
```

COLLABORA

debos: Debian OS builder

```
  # add firmware
  - action: overlay
    origin: firmware
    source: firmware-{{ $firmware_version }}/boot
    destination: /boot/firmware
```

https://github.com/go-debos/debos/

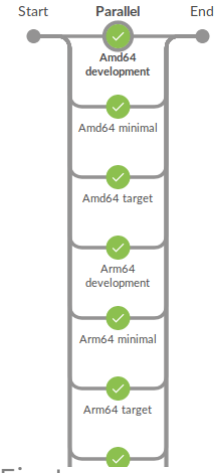
Sjoerd Simons <sjoerd@collabora.com>
Denis Pynkin <d4s@collabora.com>

Image builds

- build ospacks
- combine ospacks and h/w-specific packages and data into images
- generate sysroots for the SDK
- trigger tests on LAVA
- scan the package changelog and close bugs fixed in the packages

COLLABORA

Image builds



Amd64 development - 13s

- General SCM
- docker pull docker-registry.apertis.org/apertis/apertis-18.03-image-builder
- Shell Script
- ${BUILD_DATE_FORMATTED,"yyyyMMdd"}.${BUILDS_TODAY_Z} — Determ
- Shell Script
- Shell Script
- echo docker:x:$(id -u):$(id -g):docker gecos:/tmp:/bin/false > ${NSS_WRAPP
- Shell Script
- Shell Script
- sysroot/18.03/sysroot-apertis-18.03-amd64 — Write file to workspace
- echo docker:x:$(id -u):$(id -g):docker gecos:/tmp:/bin/false > ${NSS_WRAPP
- LD_PRELOAD=libnss_wrapper.so rsync -e "ssh -oStrictHostKeyChecking=no"

Challenges

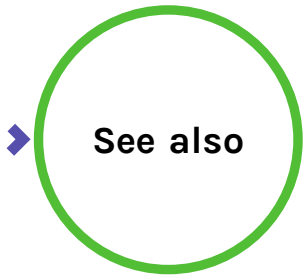
- MoM can handle simple package merges, fails on conflicts
- Git workflows help, but you can’t put a whole distro in Git
- removing old and obsolete packages
- OBS isn’t sbuild, have to deal with occasional FTBFS
- OBS ignores `Essential`, needs manual overrides

Future plans

- use GitLab for Git hosting
- focus shift to become a common platform for automotive systems, not just infotainment

COLLABORA

Interested?



https://apertis.org

https://collabora.co.uk

FOSDEM 18

Managing build infrastructure of a Debian derivative

Thanks!

Q & A