Software updates with OSTree – Why and how?

Anton Gerasimov

Advanced Telematic SYSTEMS

Advanced Telematic SYSTEMS

ATS Advanced Telematic Systems.

Open source and open standard
for connected mobility.

ATS Garage.

Web service for deploying software to embedded Linux devices.

https://app.atsgarage.com

Why do we need OTA updates?

Security

Minimize time between vulnerability detection and fix

CI

Get rid of tiresome reflashing process

Market

Our competitors already have it

Update methods.

Package-based (rpm, dpkg etc.)

+ Low bandwidth consumption

— Unreliable in absence of human operator (number of possible system configurations is exponent of number of installable packages)

Full file system update

+ Robust; can be tested exhaustively before distribution

— Consumes a lot of network bandwidth and decreases memory lifetime

Atomic differential (OSTree)

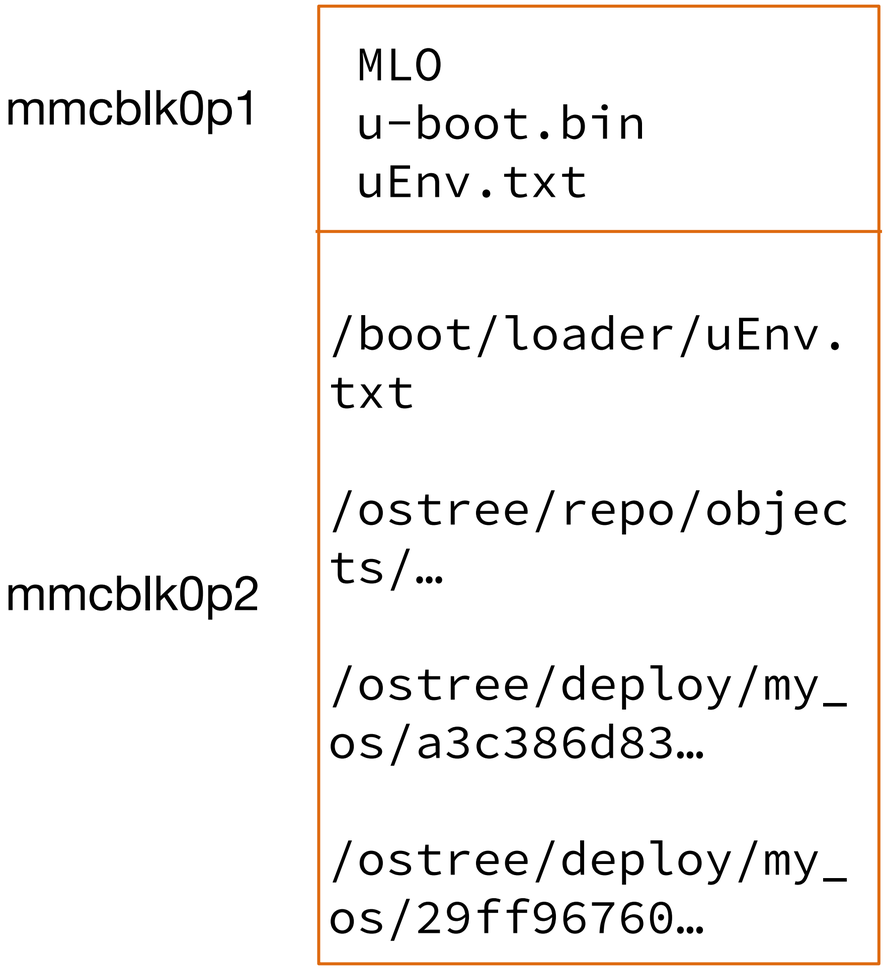
+ Combines robustness with minimal bandwidth consumption

— Requires some effort to adopt (improved recently)

OSTree.

- Git-like tool for bootable filesystems. Designed and maintained by GNOME/Red Hat developer Colin Walters.
- Original purpose: continuous integration for GNOME team.
- Target platform: PC running Linux. Not designed for embedded systems, limited support for other POSIX-compliant OSes.
- More info on ostree.readthedocs.io

OSTree basics.

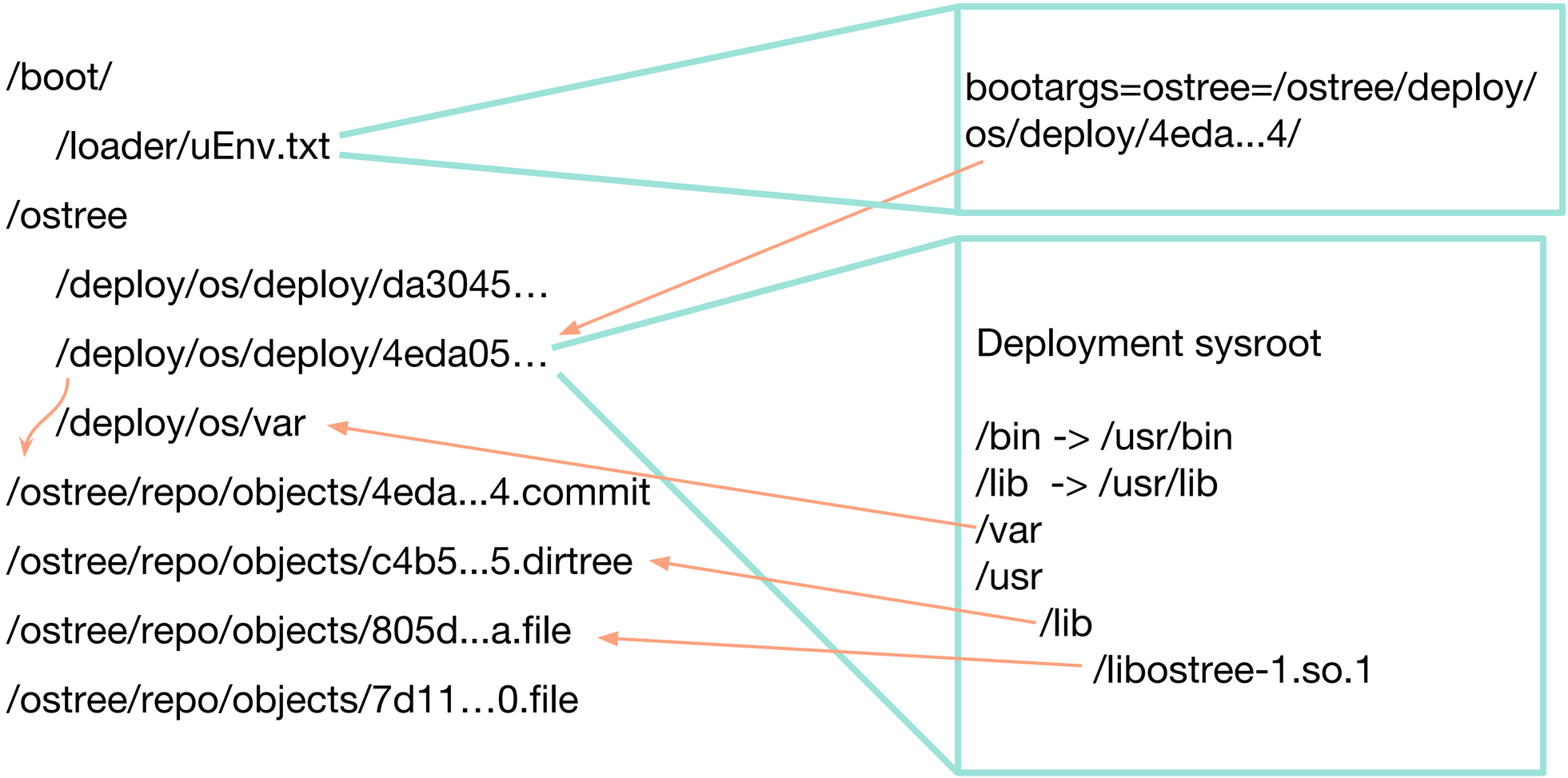


- Physical sysroot - just one per device. Contains OSTree repo, OSTree deployments and /boot directory with information about current deployment sysroot. Device never boots into physical sysroot.
- Deployment sysroots - one device can contain multiple deployments (two by default). They are stored in /ostree/deploy under physical sysroot. Physical sysroot is mounted to /sysroot mountpoint of deployment sysroot so that OSTree can access its repository.

OSTree basics: boot procedure.

- Bootloader reads kernel, initramfs and deployment sysroot location from /boot/loader/uEnv.txt and boots into initramfs.
- Initramfs prepares deployment sysroot: mounts /var, /home and /sysroot, remounts /usr as read only.
- After the sysroot is prepared, initramfs boots into it.

OSTree basics: sysroot



OSTree integration.

Already done in meta-updater

1. Prepare physical sysroot.
2. Prepare deployment sysroot.
3. Make bootloader and initramfs work together to boot the deployment.

4. Make sure you control mutable state in your system.

What if I just commit my rootfs to OSTree?

Deployed files are hardlinks to objects in OSTree repo and are shared between deployments. Therefore they can’t be modified by running system.

- All files managed by OSTree should reside in /usr that is mounted read-only.
- Writable files should reside in /var, but software should be aware of how to populate it with initial data.
- OSTree already manages /etc. Not really fit for embedded systems.

Case of AGL Application Framework (1).

Two update domains.

1. Full file system updates with OSTree.

2. Application updates with Application Framework.

Application database is located in **/var/lib/afm.** Some applications come pre-installed in the file system, while other can be installed in runtime.

How do we manage **/var/lib/afm**?

Case of AGL Application Framework (2).

Just ignore initial database.

\+ Almost zero integration effort

— No pre-installed apps

Merge initial database in /usr/afm with the one generated runtime.

\+ Applications can be updated both with OSTree and AppFW

— A lot of integration effort, merger can fail or give unexpected results.

Populate /var/lib/afm from /usr/afm just once.

\+ Moderate integration effort, very robust.

— Pre-installed apps are populated just once, can’t update apps with OSTree.

Meta-updater: Yocto/OE layer for OSTree updates.

Implements

- Seamless integration into Yocto build process.
- Deployment sysroot as an OSTree commit.
- Physical sysroot and bootable images for supported platforms.
- Pushing OSTree commits to a server through a well-documented protocol.

Does not implement

- Population of /var. It is application-dependent.
- Support for arbitrary board. Currently Raspberry Pi 2/3, Minnowboard Turbot, Renesas RCar Porter board and qemux86-64 are supported.

Open issues.

- /etc merger. The way it is implemented in OSTree doesn’t work well for embedded systems.
- File system stability. Physically there is only one file system, and if it gets corrupted due to hardware bugs, driver bugs etc. the system becomes unbootable.
- OSTree itself is a part of deployment sysroot => system can be bricked.
- Rollback logic is not a responsibility of OSTree. Ideally it should be implemented in the bootloader.

Links.

- OSTree: https://github.com/ostreedev/ostree
- AGL: https://www.automotivelinux.org/
- Meta-updater: https://github.com/advancedtelematic/meta-updater
- Quickstart with meta-updater and Raspberry Pi: https://github.com/advancedtelematic/garage-quickstart-rpi

Contacts

ATS Advanced Telematic Systems GmbH
advancedtelematic.com

Anton Gerasimov
Embedded Software Engineer

anton@advancedtelematic.com