

TOSHIBA

Leading Innovation >>>

How do you update your embedded Linux devices?



**Daniel Sangorrin
Keijiro Yano**

BoF session
LinuxCon Japan
July 14th, 2016

Introduction

- **Goal of this BoF session**

- Discuss the availability and properties of **generic tools** for updating arbitrary embedded Linux devices.
- Share **know-how** on updating techniques.
- Get an idea of what needs to be done **next**.

- **Schedule**

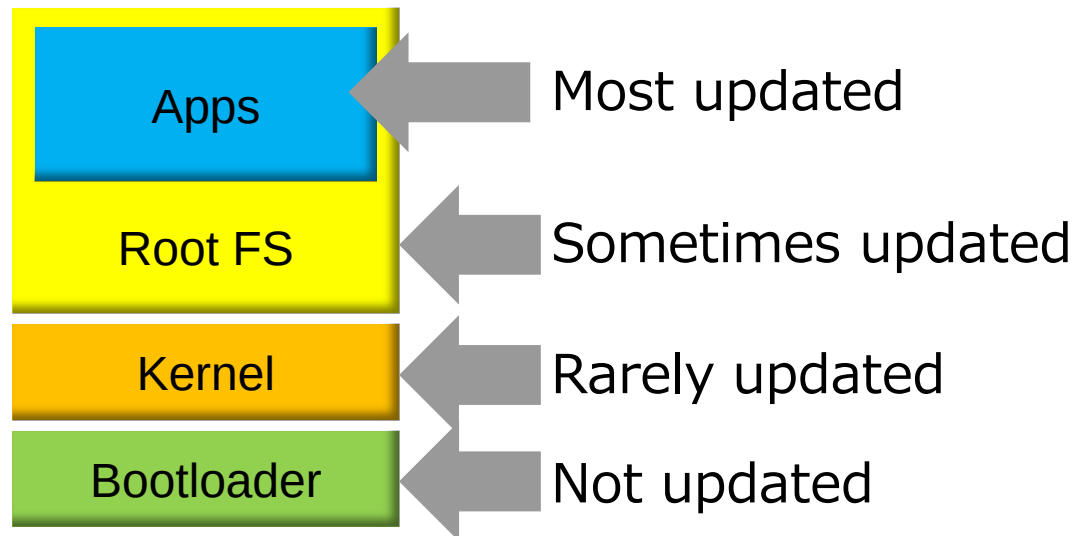
- 10' of presentation
- 30' of interactive discussion



Get involved!

Survey results

- **No single solution for all**
 - Local vs remote, manual vs automated, 10's vs 1000's devices, criticality, allowed downtime, storage limitations, air-gap vs network
- **Used tools: custom scripts**
 - A lack of quality open source tools or easy to follow guidelines?
- **Main purpose: application related updates**
 - Security fixes added by inertia (especially on stand-alone systems)
 - Bug fixes have risks (busybox: [6bd3fff51aa74e2ee2d87887b12182a3b09792ef](#))



Typical requirements

- **Easy**

- Well documented (for various update patterns)
- Autogeneration of config files, crypt/hash/signs, deltas..
- Platform integration (OpenEmbedded/Yocto, buildroot..)
- Small runtime dependencies (busybox, shell scripts..)

- **Extensible**

- Custom hooks/plugins for each update step

- **Robust**

- Fail-safe: version rollback, watchdog, sanity test, factory reset, atomic (all or nothing), power cuts
- Good error messages on failure

- **Secure**

- CIA triad (confidentiality, integrity, authenticity)
- Compatible with secure boot

Typical update patterns

- **Manual install (CD/USB/LAN..)**
 - Gets harder as the number of devices increases
- **Dual partition/copy**
 - Chrome OS, CoreOS..
- **Overlay**
 - Ostree (git-like), Snappy, overlayfs, docker, lxc, chroot..
- **Network boot**
 - PXE, tftpboot..

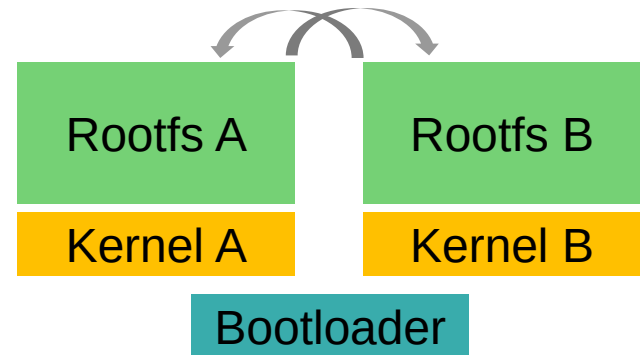


Fig 1.- Dual-partition update pattern

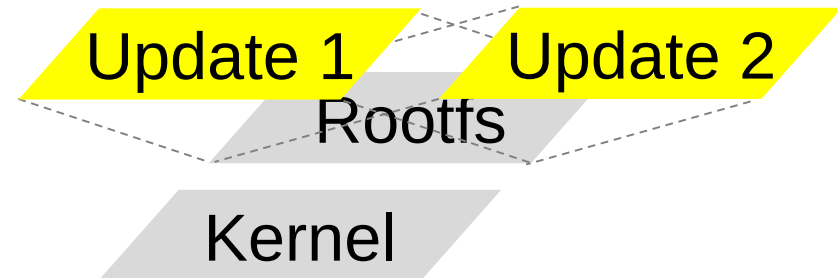
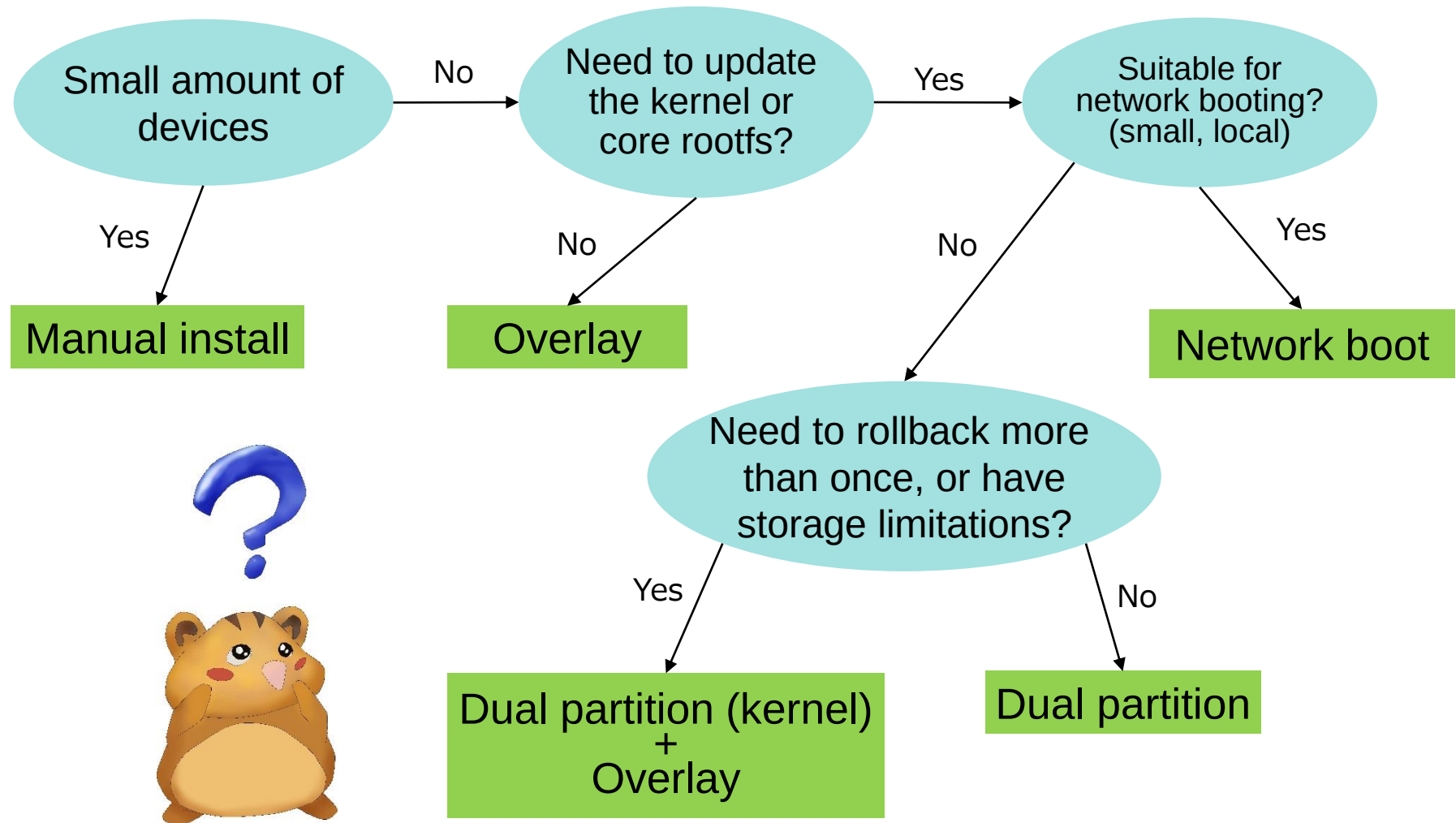


Fig 2.- Overlay update pattern

Note: package management tools (apt-get..) are orthogonal to these patterns.

Decision flow diagram (simplified)



Dual partition example: Chromium OS

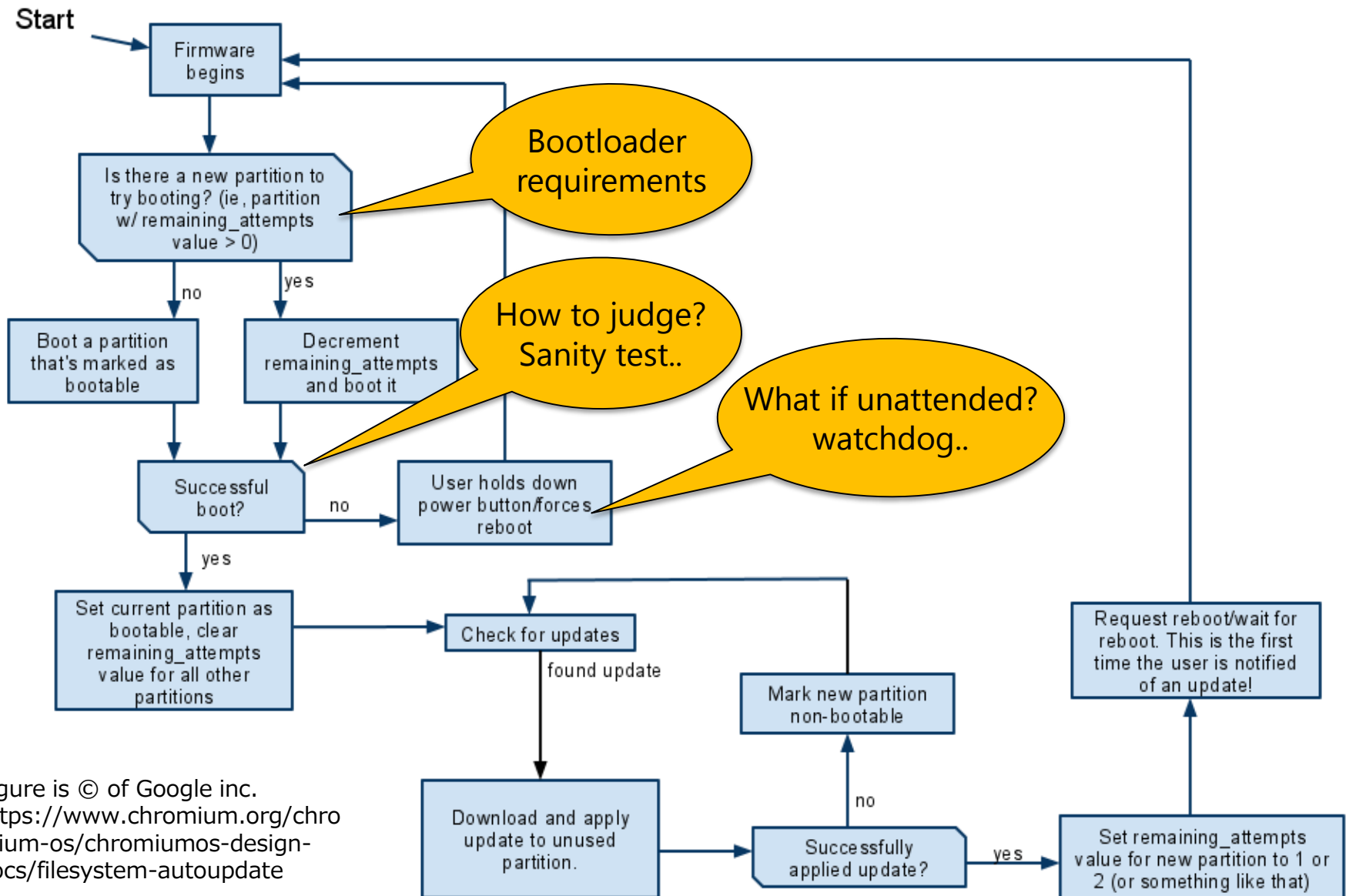


Figure is © of Google inc.
[https://www.chromium.org/chromiumos-design-docs/filesystem-autoupdate](https://www.chromium.org/chromium-os/chromiumos-design-docs/filesystem-autoupdate)

Generic tools

- **Bootloader support**

- U-boot: [bootlimit/bootcount/altbootcmd](#), redundant env, SPL
- GRUB2: [fallback](#) (no redundant env?)
- [Chromium OS](#): coreboot + u-boot + [GPT flags](#)

- **Update tools**

- [Swupdate](#): nice and extensible software update framework
- [Ostree](#): git-like software update framework
- [Mender](#): dual-partition pattern oriented framework
- [Swupd](#): revisioned software update mechanism (clearlinux)
- [Fwup](#): configurable image-based firmware update tool
- [Sysup](#): update rootfs from the initramdisk
- [Resin](#): docker/nodejs-based framework
- Dynamic update tools: [kpatch](#), [kgraft](#)..

Ref: <https://lists.linuxfoundation.org/pipermail/automotive-discussions/2016-May/002061.html>

Related projects

- **Projects for reference on software updates**
 - [Chromium OS autoupdates](#) (dual-partition)
 - [Android OTA updates](#) (manual recovery mode)
 - [Android 'N' will follow Chromium OS](#) (dual-partition)
 - [OpenWRT sysupgrade/LuCI](#) (manual recovery)
 - [overlay failsafe mode](#), [tftpboot flashing](#), [JTAG debriking](#)
 - [VyOS](#) (network OS, manual rollback through grub)
 - [CoreOS](#) ([dual-partition](#)), [Project Atomic](#) ([rpm-ostree](#)), [Snappy Ubuntu](#) ([transactional delta updates](#)), [ClearLinux](#) ([swupd](#))

Related presentations during this event

- **Automotive Grade Linux**

- Yannick: “[Secure boot and Secure software updates](#)”
- Arthur: “[Secure updates for Linux-based IVI systems](#)”
 - http://advancedtelematic.github.io/rvi_sota_server/
- Eystein “[Securing the Connected Car](#)”

- **Civil Infrastructure Platform (CIP)**

- Kobayashi&Jan: “[Introducing the CIP project](#)”
- Hayashi: “[Generating a reproducible and maintainable Embedded Linux Environment with Poky and Deby](#)”

- **LTSI/Fuego: automated testing (contribute!)**

- Tim bird: “[Introduction to the Fuego test framework](#)”

Extra references

- [Deviceside Software Update Strategies for Automotive Grade Linux](#) (Konsulko Group, sponsored by Advanced Telematics Systems GmbH)
- [Software Update on Embedded Systems](#) (Stefano Babic, DENX GmbH, ELCE 2014)
- [Building a robust Embedded Linux platform](#) (Thilo Fromm, FrOSCon 2012, [video](#))
- [Updating Embedded Linux devices in the field](#) (Chris Simmonds, 2net Ltd)
- [Building Murphy-compatible embedded Linux systems](#) (Gilad Ben-Yossef, Codefidence Ltd)
- [Safe upgrade of embedded systems; Upgrade without Bricking](#) (Arnout Vandecappelle, Essensium N.V.)
- [Redundant Booting with U-Boot](#) (Thomas Rini, TI)

Discussion time!

- What software **tools** do you use for updating (swupdate, custom, ...)?
- Do **bootloaders** require extra functionality (watchdog..)?
- What do you think about **rollbacks** with chroot, overlayfs, docker, brtfs..?
- What characteristics must a **deployment** framework have? (version synchronization across devices..)
- What **other** problems do you have in software updating?



Possible issues (1/3)

- **Disconnection between projects**
 - Don't reinvent the wheel.
- **x86 support**
 - Most tools and documentation focus on ARM (or PowerPC).
- **Bootloader fail-safe support**
 - GRUB2: no redundant environment.
 - Watchdog support (kernel panic timeout is not enough).
- **Distributed systems**
 - All units may need to synchronize to the same version.
- **Bugs in the server/client update tools**
 - Write clean code, avoid rarely used libraries, fuzzy tests..
- **Documentation**
 - End-to-end guidelines, patterns and tools comparison.

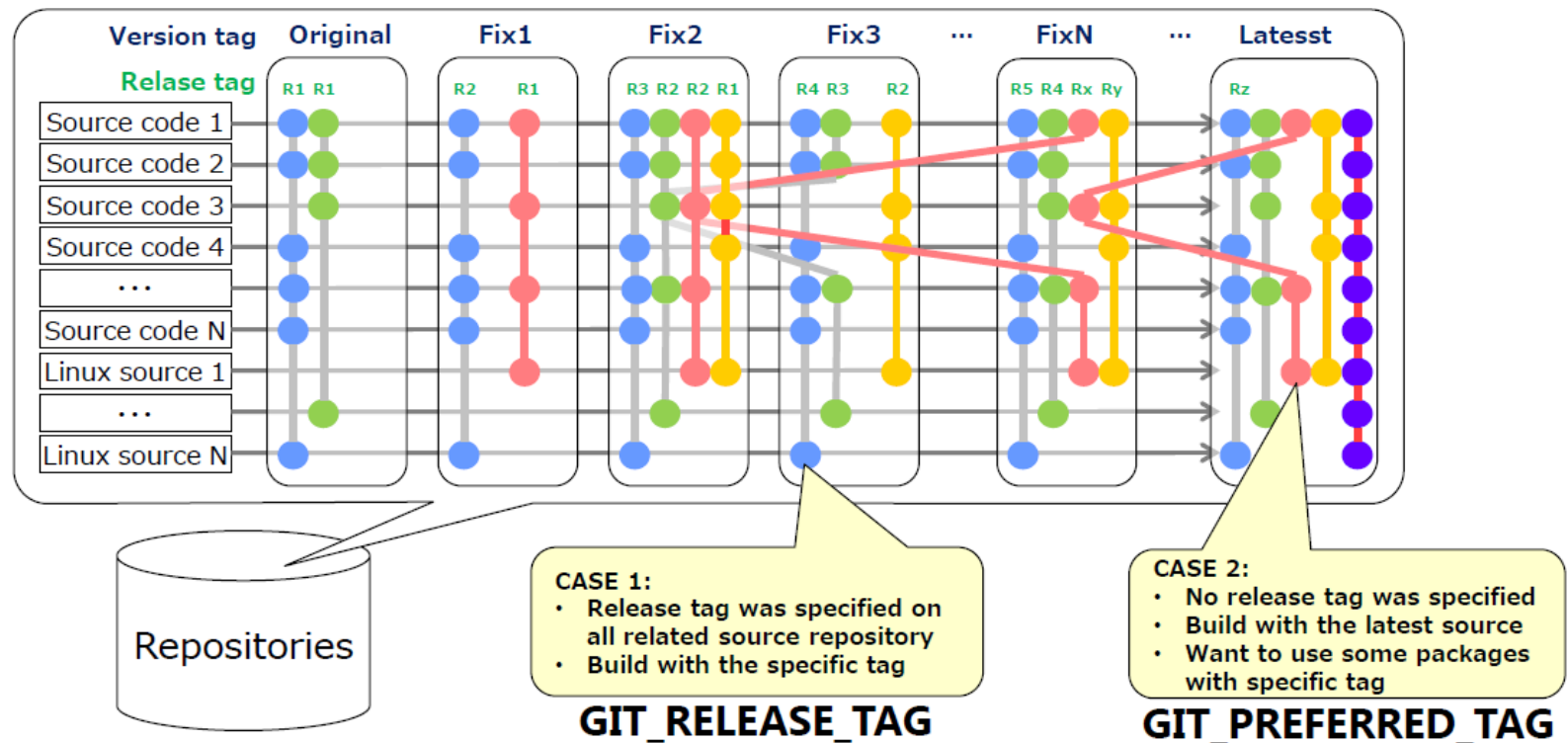
Possible issues (2/3)

- **Sanity testing after an update**
 - Checksum files after installation.
 - Automated tests on the board.
 - Check that you can install future updates.
- **Compatibility with security tools**
 - Secure boot, Linux IMA/EVM.
 - LSM: tomoyo, smack, selinux, apparmor.
- **Target update tool's dependencies**
 - Favor stable maintained long-term library versions.
 - Eg: Same versions as stable long-term distros (Jessie..).
 - Embedded systems have many constraints.
 - Minimize language dependencies, dependencies on boot scripts, size, version conflicts and recipe backporting.

Possible issues (3/3)

- **Source code management**

- Each update may add/remove/update parts of the rootfs
 - Git-tag the source code for each rootfs (see Deby)



TOSHIBA

Leading Innovation >>>