SECURITY IN LARGE SCALE EMBEDDED SYSTEMS

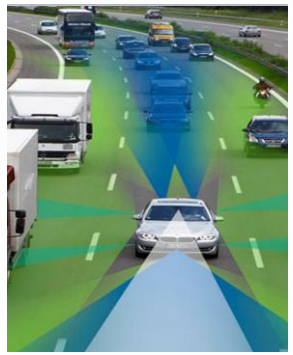
BMW Car IT GmbH

BMW CAR IT - WHO ARE WE?

Fully owned BMW subsidiary in Ulm (Germany)

Software Development, Integration and Testing for:

Infotainment Systems



Driver Assistance

Autonomous driving

WHAT DOES MY TEAM DO?

BMW Yocto Linux Platform (Node0) / Headunits (since 2015)

Agile Engineering Team
(6 Security Engineers, 1 Product Owner Security Features,
3 Project Manager)

Security Software Development

- Cryptographic Support/Protocol Libraries
- Privileged Execution Wrappers
- IPC Policy mechanism

...

System Configuration/Integration

- Systemd configuration (Users, Capabilities, ...)
- SELinux
- IPC ACLs
- Secure Boot
- Dm-Verity

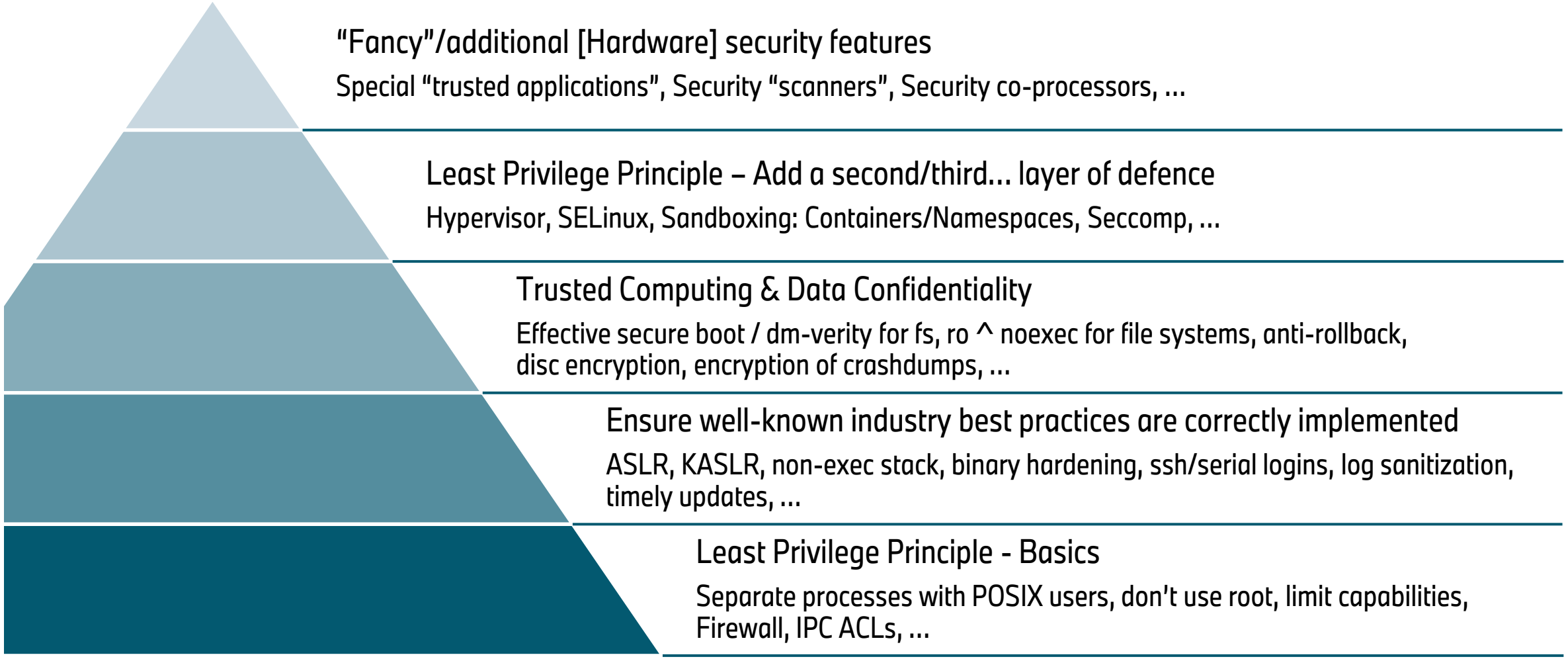
...

In-House Security Consulting

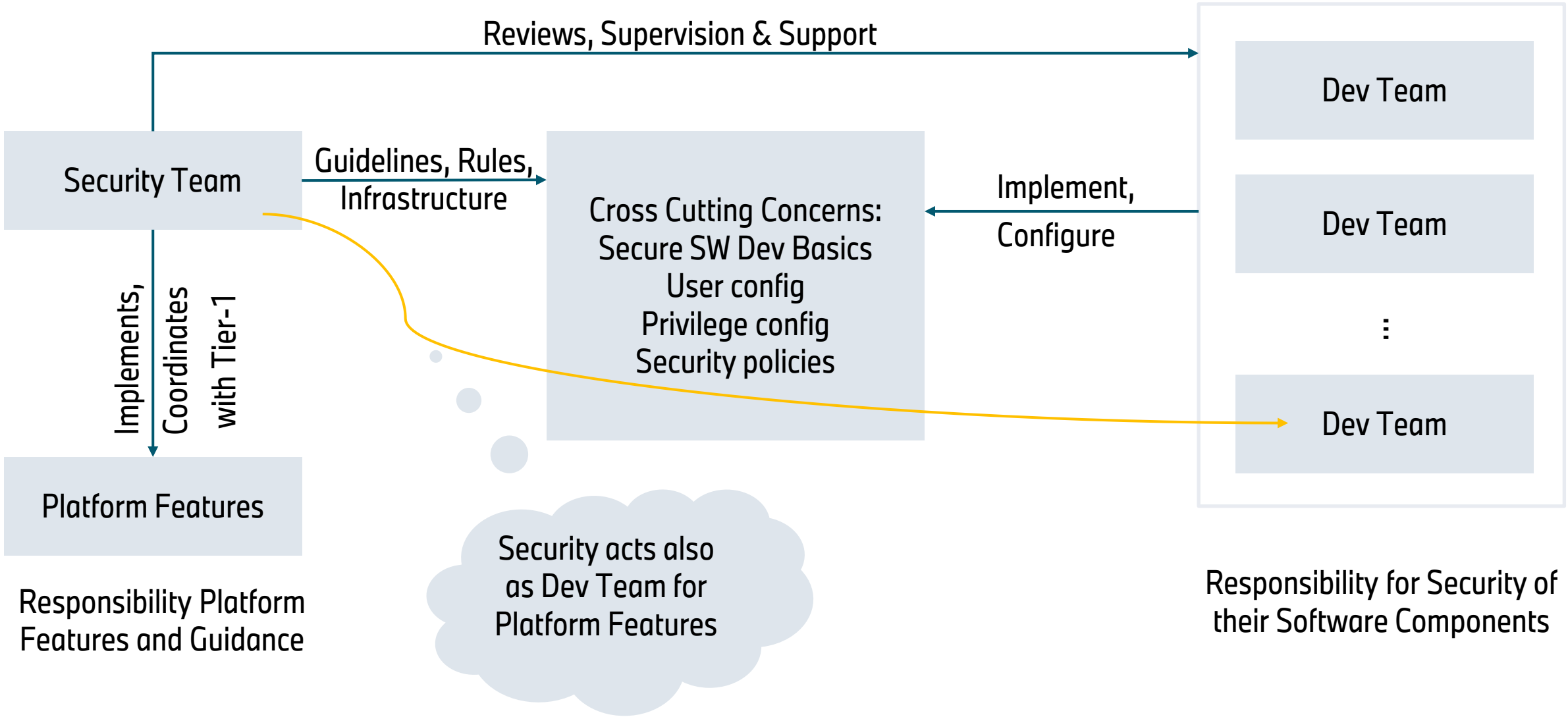
- Support Feature Architects/Developers in creating/implementing security concepts/a secure implementation
- UNECE R-155 „Automotive Cybersecurity“ compliance

...

SECURITY NEEDS PYRAMID – A PERSONAL VIEW FROM SYS ENG PERSPECTIVE



RESPONSIBILITY MODEL



FEATURES WE PROVIDE OUT OF BMW LINUX BASE PLATFORM (NODE0)

Permission Management

- User & Permission Management
- SELinux policy (refpolicy)
- Debug Mode

Networking integration

- Firewall (nftables)
- IPSec (strongswan)
- MACSec (MKADaemon)

Security software

- IPC ACLs (dbus & vsomeip)
- Smolkit (small polkit replacement)
- Cryptographic support (MoCoCrW)
- Privileged Execution Wrappers

Build system infrastructure

- Secure Boot signing backend utilities (PKCS#11)
- Dm-verity utilities
- Hardening Guidelines, Checkers and Templates

USER & GROUP MANAGEMENT

- Central usermanagement.bb recipe with includes for better readability and structure
- Configure users for services

```
# User/Group ID for vsomeipsecd
MGU_USERS:${PN} += "sec_vsomeipsecd"
MGU_UID:sec_vsomeipsecd = "4014005"
MGU_SUP:sec_vsomeipsecd = "mgu_dlt,mgu_someip"
```

- Configure shared groups for platform features

```
# Access to logging infrastructure
MGU_GROUPS:${PN} += "mgu_dlt"
MGU_GROUP_GID:mgu_dlt = "5001"
```

- Pre-alloctated uid ranges for self-service of dev teams
- Review to ensure overall „secure“ user config, shared group config and assiginment of supplementary groups

WE HAPPYLY USE WHAT SYSTEMD PROVIDES

Permission Management

Basics:
```
User=
Group=
SupplementaryGroup=
```

Capabilities:
```
CapabilityBoundingSet=
AmbientCapabilities=
```

Environment Setup

Privileged Startup-Actions
```
+ExecStartPre=
!ExecStartPre=
```

Environment organization
```
RuntimeDirectory=
RuntimeDirectoryMode=
```

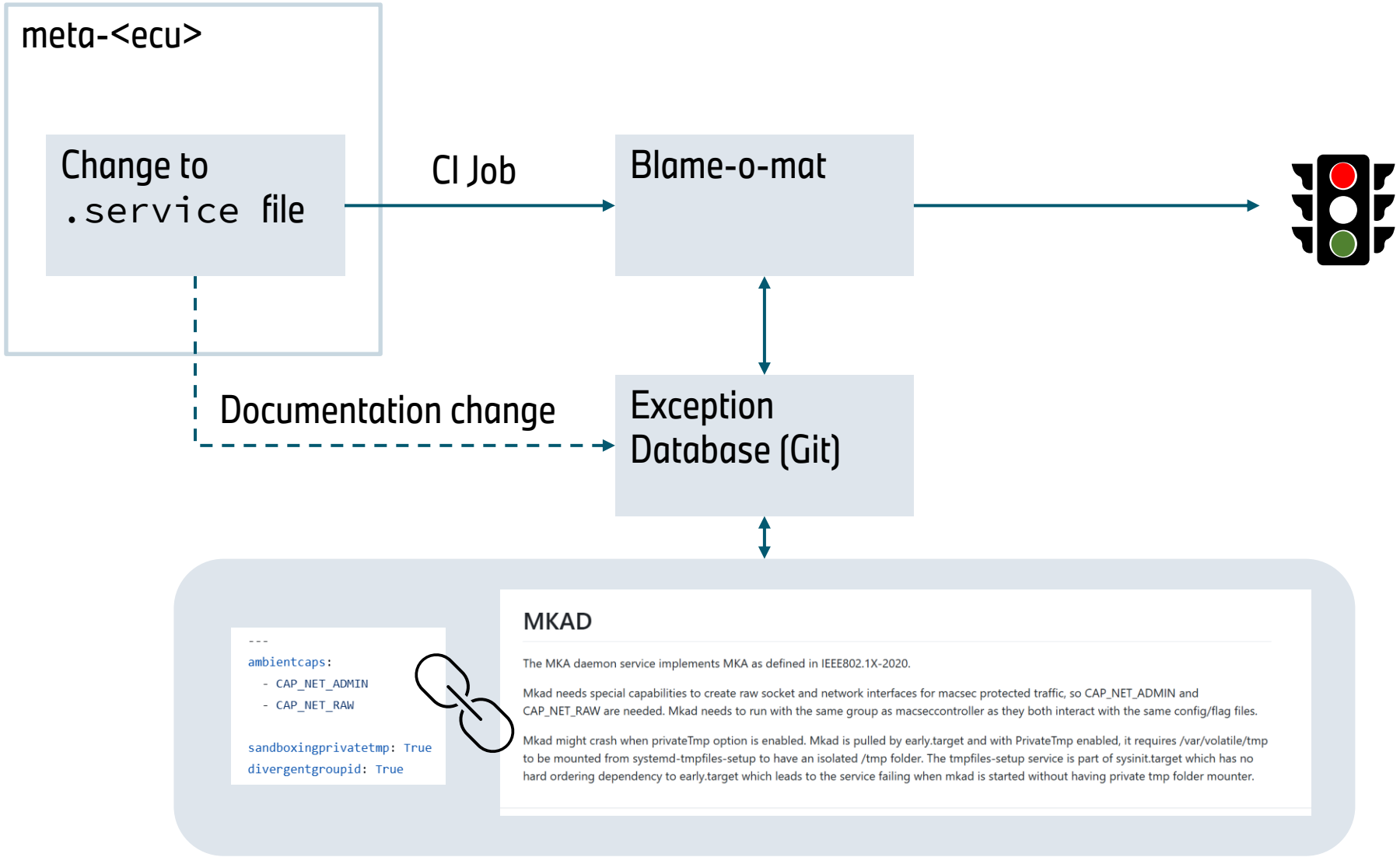
Sandboxing

Namespaces:
```
PrivateTmp=
ProtectKernelTunables=
ProtectControlGroups=
ReadOnlyPaths=
…
```

Seccomp:
```
SystemCallFilter=
SystemCallArchitectures=
MemoryDenyWriteExecute=
ProtectKernelModules=
…
```

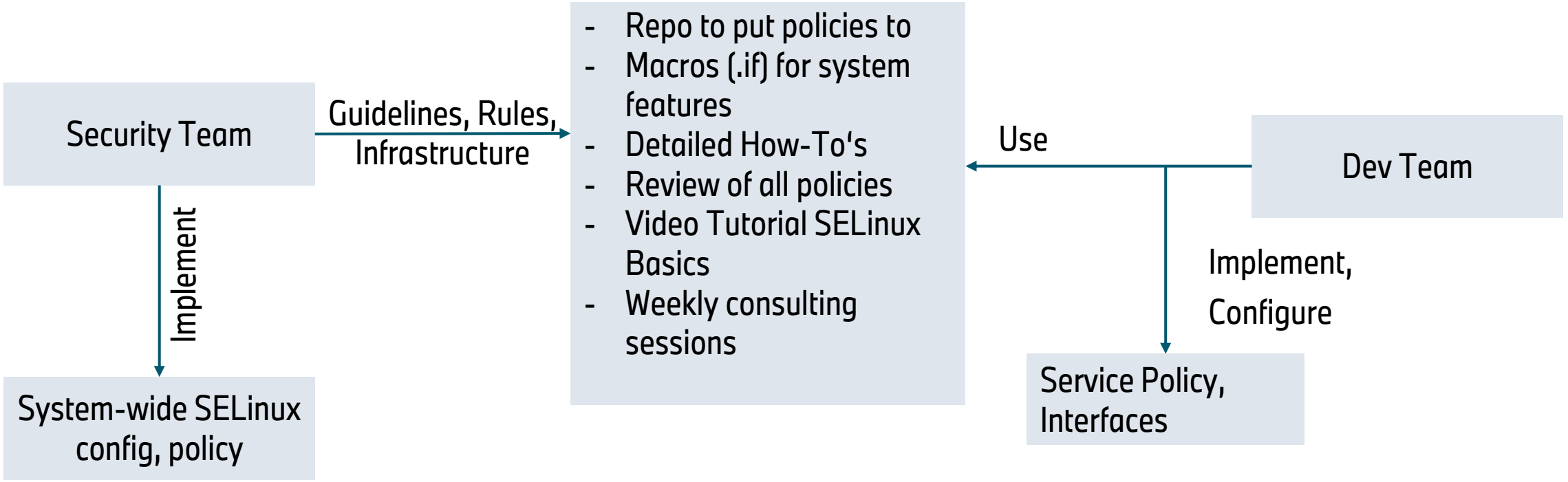
BLAME-O-MAT – EXCEPTIONS & DOCS

Interest to have this open source?



LESSONS LEARNED FROM SELINUX ROLL-OUT

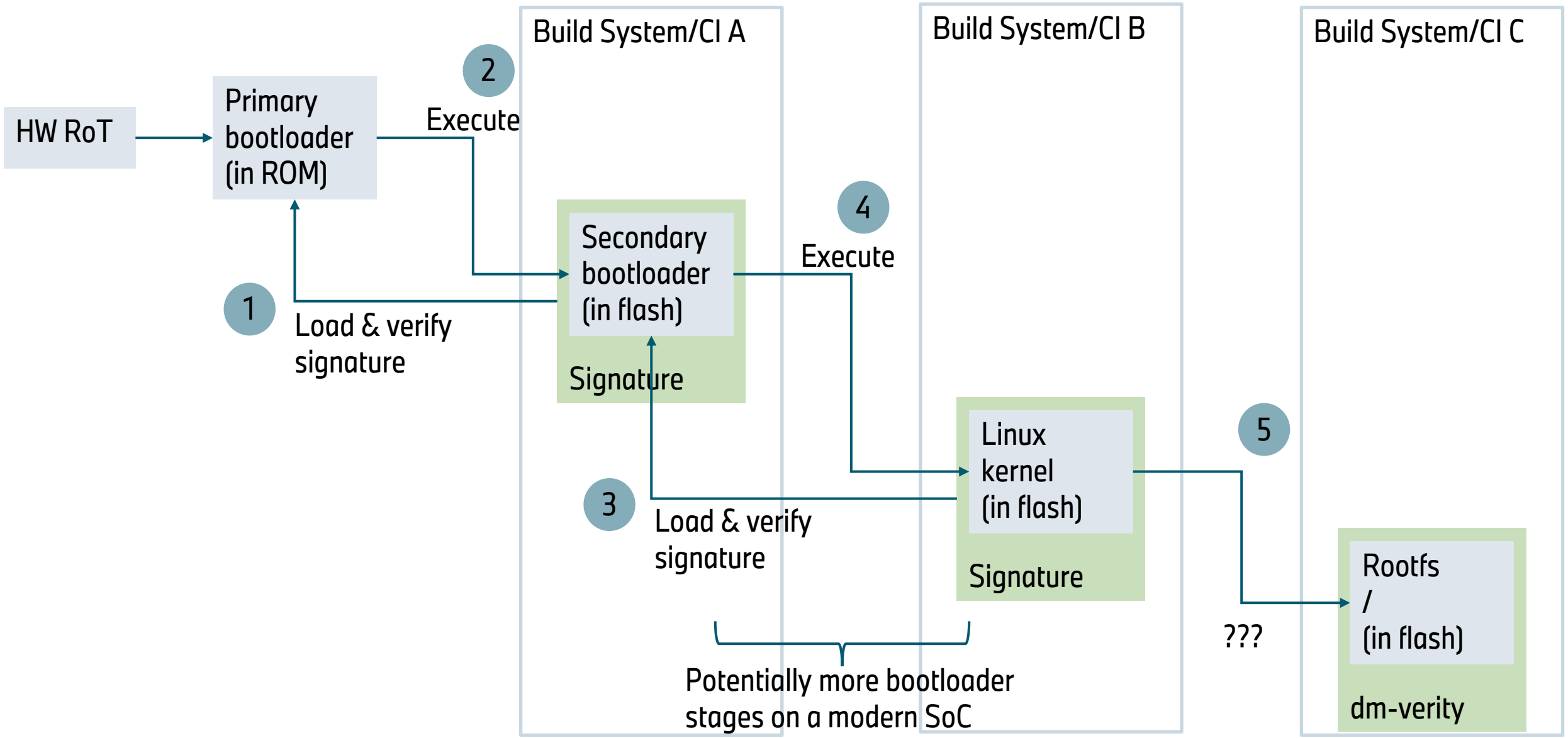
- SELinux is a complex beast → Why would we do this to ourselves?
- It's a good fit for a (fully configured) embedded system <> Usage in Android
- We use it based on the `refpolicy`
- Example how we employ our general responsibility model:



LESSONS LEARNED FROM SELINUX ROLL-OUT II

- Rollout procedure:
  - We created barebone policies with individual (permissive) types for all services, so that all services are executed with their own type
  - Security team worked on policy so that SELinux can be (globally) switched to enforcing
  - Monitoring of violation logs from automated test setuos and test drives in cars
- Project management
  - Define 6 month timeline to create „real“ policy
  - Clear interface management for interaction between types (provide in .if and early-on)
  - Manage the integration madness → Put policy files into meta layer repo
  - Took in total 1 year then but completed for head unit that launched in 2022 in BMW 7 series
- Lessons Learned
  - SELinux is a complex beast (even more than we already knew)
  - Collaboration & Support security team ←→ software development teams is key
  - A lot of patience and willingness to explain things over and over again, create awareness to personally convince developers that „it makes sense“ and is not just „extra work that breaks their already working features“
  - Start early to get it done parallel to regular development
  - It does get less painful over time

TRUSTED COMPUTING / SOFTWARE DELIVERY CONSTRAINTS



DM-VERITY + ROOTFS SETUP

Initrd

- Veritysetup called in userspace
- Flexible options for getting/verifying root hash
  → rootfs build indepdentent from bootloader
- Extra load time for initrd

Dm-init

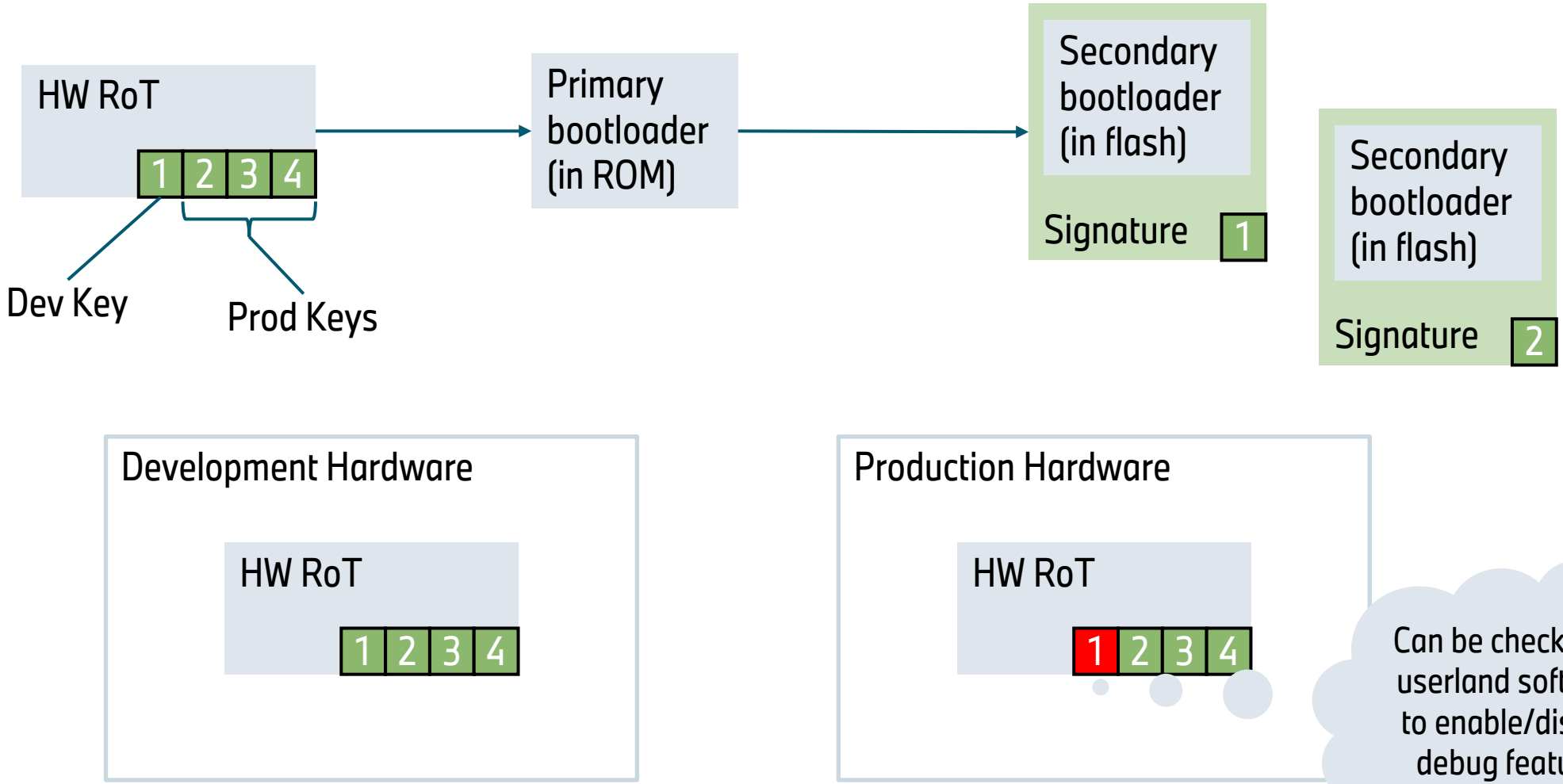
- All setup parameters in kernel commandline
- Bootloader needs to provide root hash
  → Special bootloader extension needed or dependency from rootfs to build system of bootloader.
- Potentially fast direct boot into main rootfs

?

Signature support in kernel?

- Does exist (via `root_hash_sig_key_desc`)
- Signature cannot be read from disc but needs to be provided by userspace ☹
- Root hash and some per-tree config can only be provided in kernel cmdline ☹
- If you worked on this → Would be really great

DEVELOPMENT: LEVERAGE MULTI-ROOT-OF-TRUST FEATURES



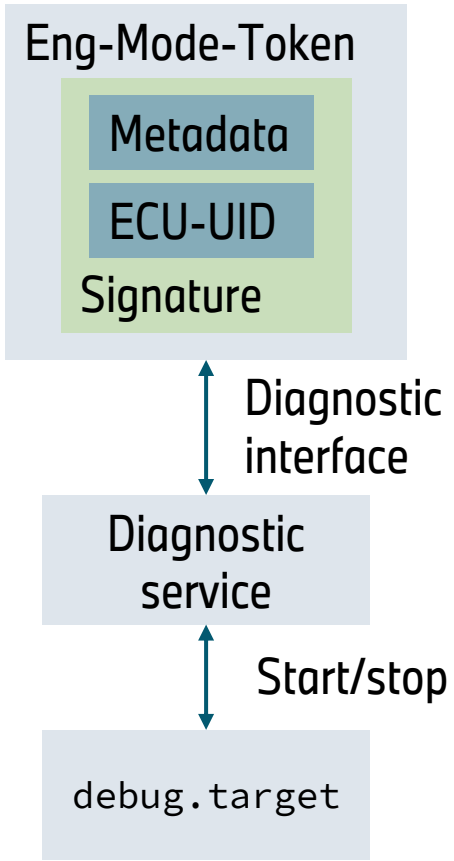
DEVELOPMENT: TOKEN-BASED DEBUG MODE

- How to debug production software?
- Special debug hardware vs. Validate the „final product“

- Our approach: `debug.target`
  - Serial shell
  - SSH
  - Relaxed firewall
  - ...

- Cryptographic token based on unique hardware device identifier

- Special deactivation mechanisms that require software changes for: SELinux, dm-verity, IPC ACLs



OUR KEY INSIGHTS

- Security is a system engineering task → System overview and feature understanding is strictly required
- KISS! – the dummer the approach is, the higher its probablility to work
- Try to reduce complexity for the projects (provide easy-to-use/hard-to-misuse APIs and scripts)
- Create awareness and trust (but verify)
  → Blame-o-mat / Self-Service approach but review for security-critical config
- Self-Understanding as „support team“ and not „police team“
  → Establish good working relationship with other development teams
- Open Source has (almost) everything you need
  - No need to reinvent the wheel → Just sometimes one requires a new suspension

QUESTIONS?