DIGLIM eBPF

Department name: Dresden Research Center (DRC)
Author’s name: Roberto Sassu
Date: 02/05/2022

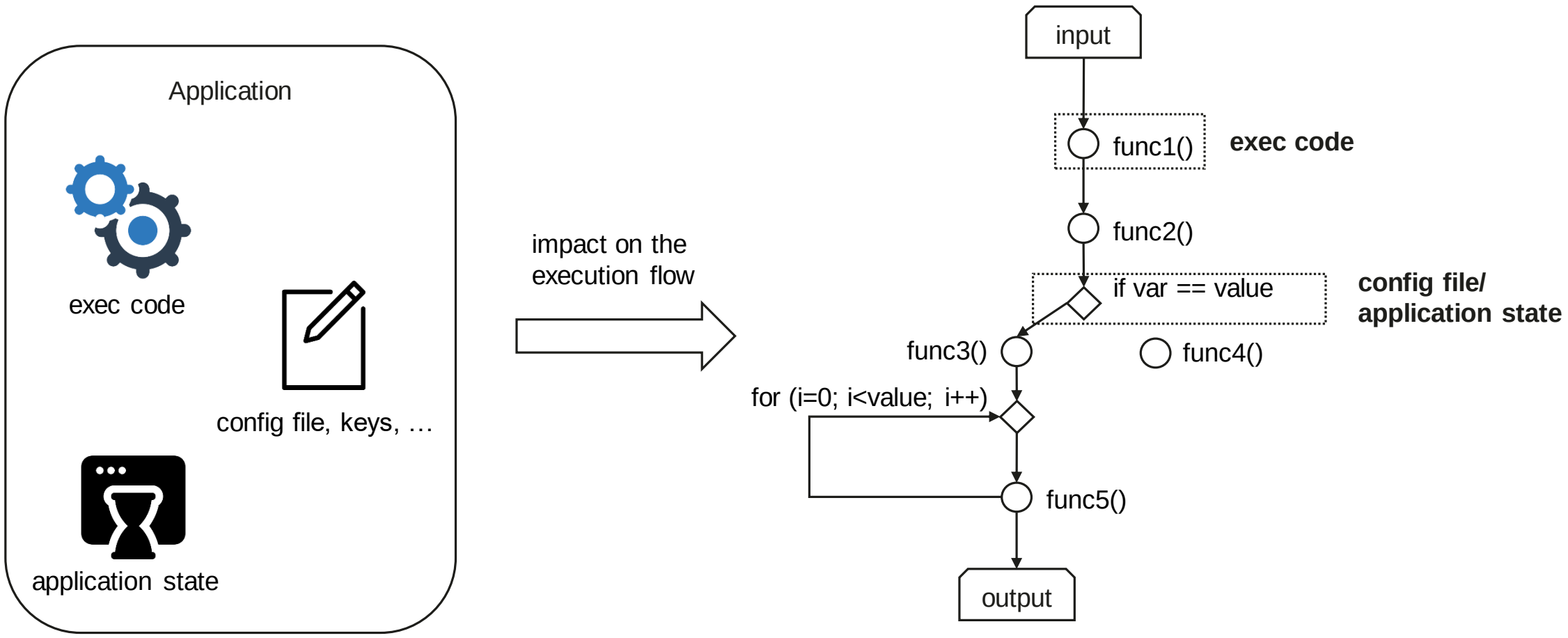
HUAWEI

Contents

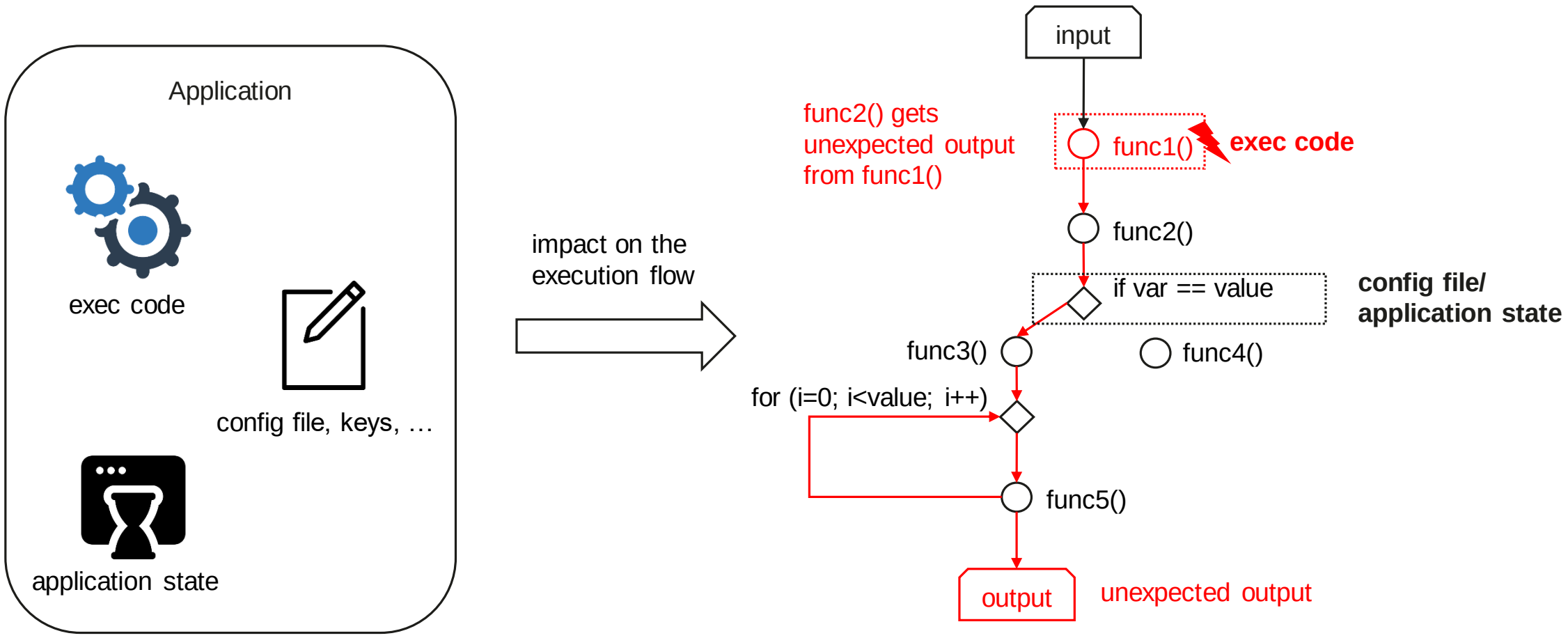
Goal of This Presentation

- Propose a flexible and scalable solution for integrity protection at OS-level
- Describe its implementation with eBPF
- Discuss how it can be integrated in the eBPF kernel subsystem

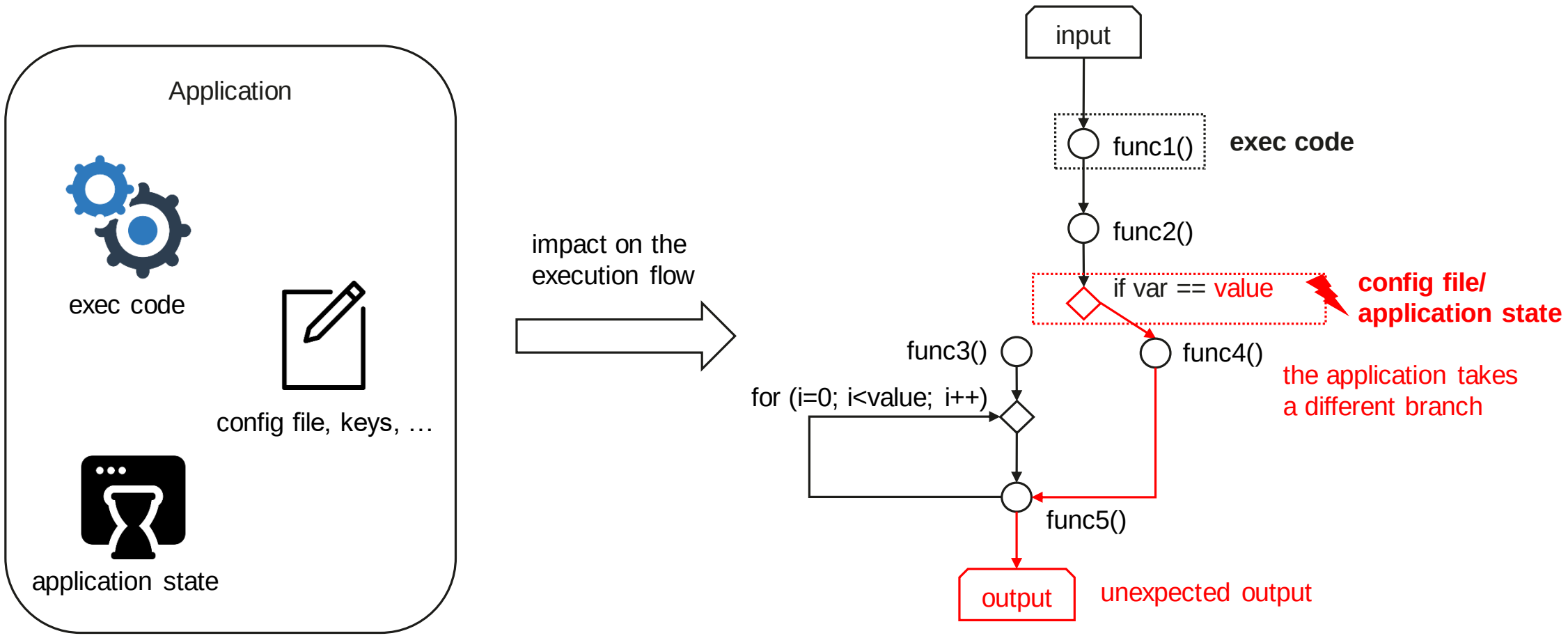
Integrity Protection – What Are We Protecting Against?



Integrity Protection – What Are We Protecting Against?

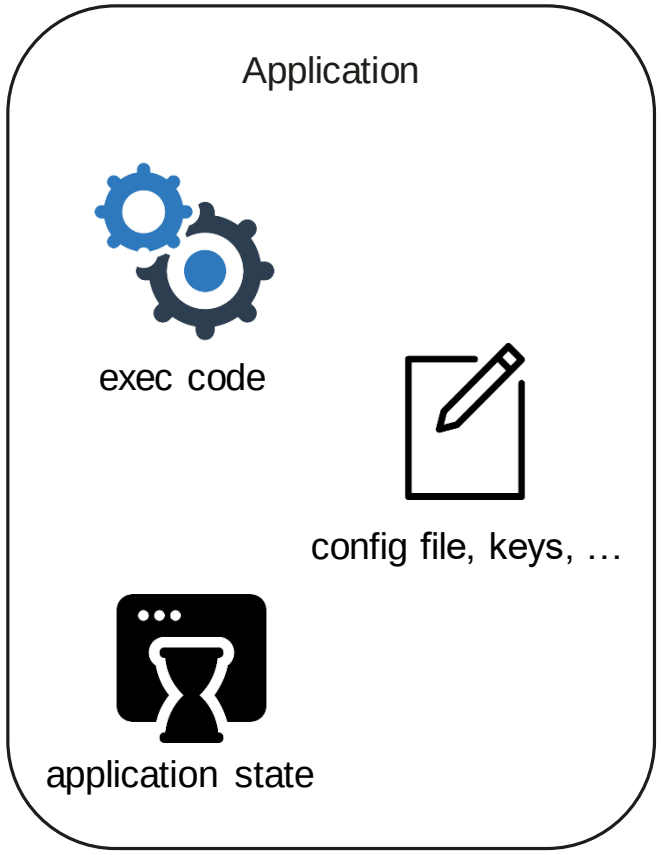


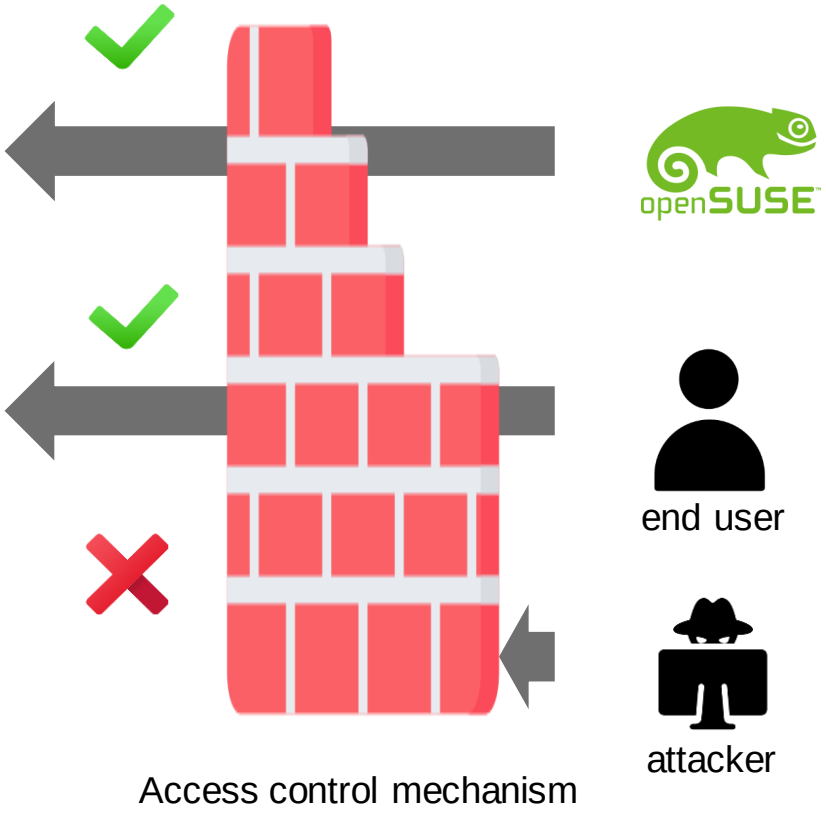
Integrity Protection – What Are We Protecting Against?



Integrity Protection – How Do We Protect the Application?

Let only approved sources be loaded by the application



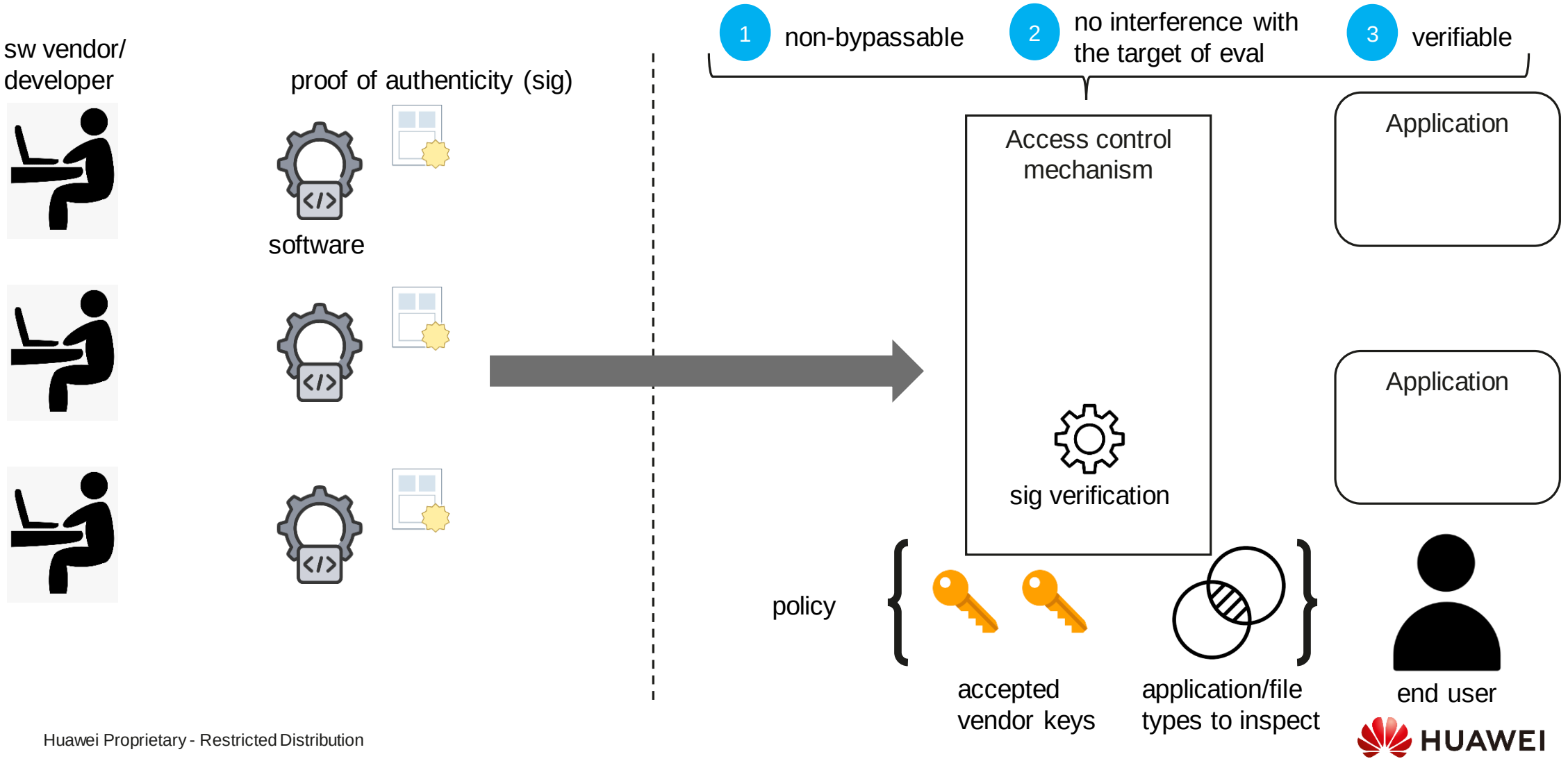


end user

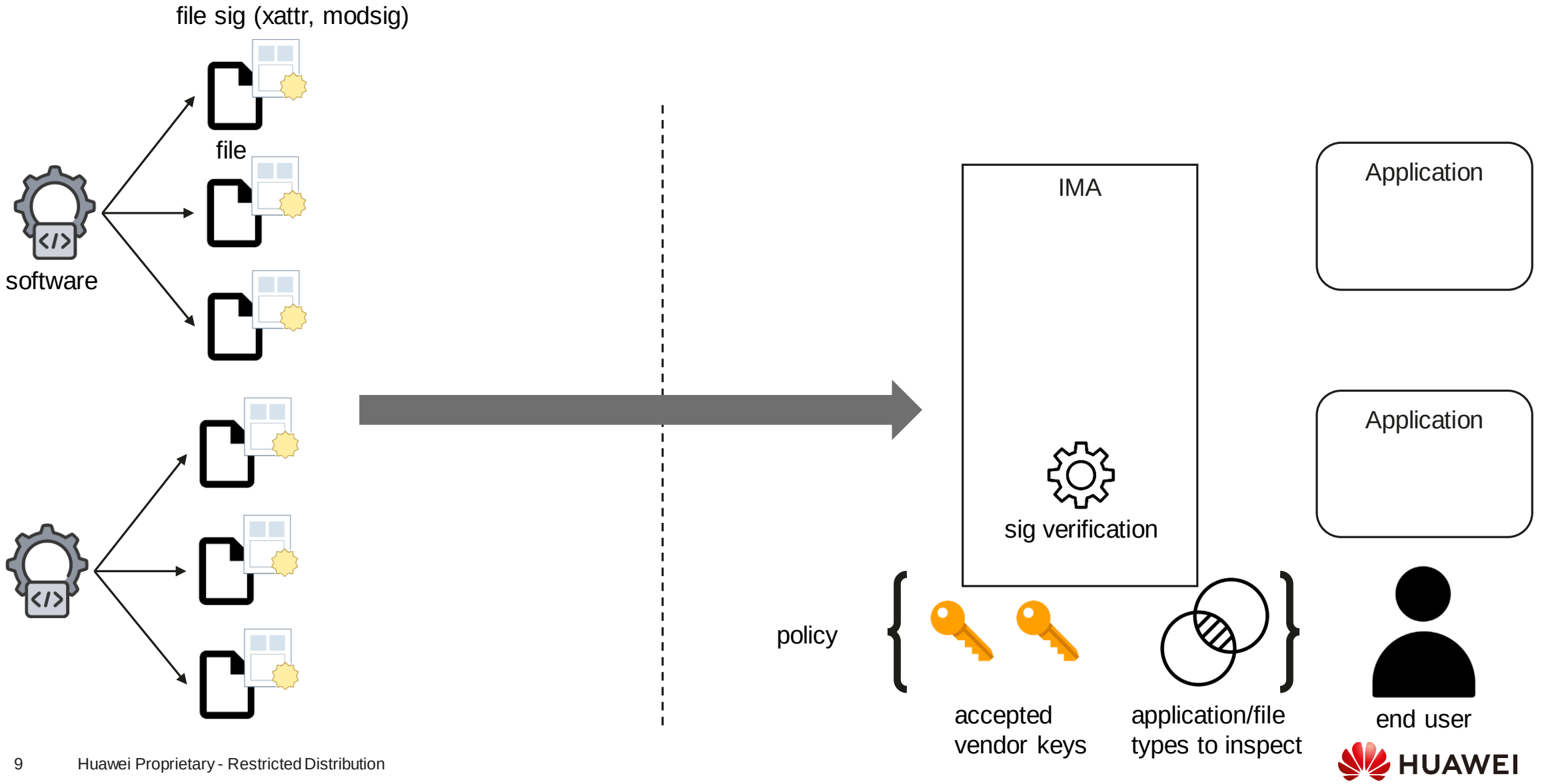
attacker


Does not work for application state!

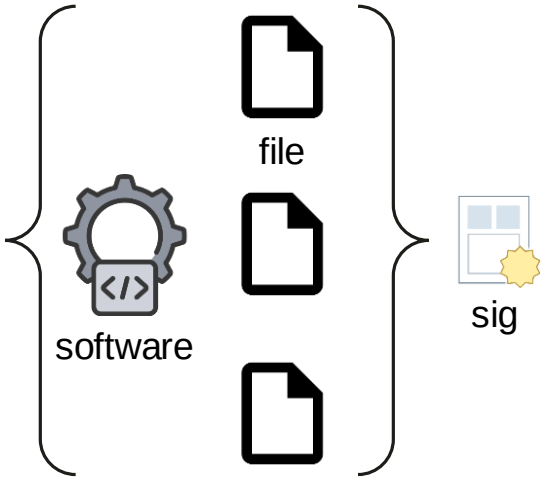
Integrity Protection – Generic Solution



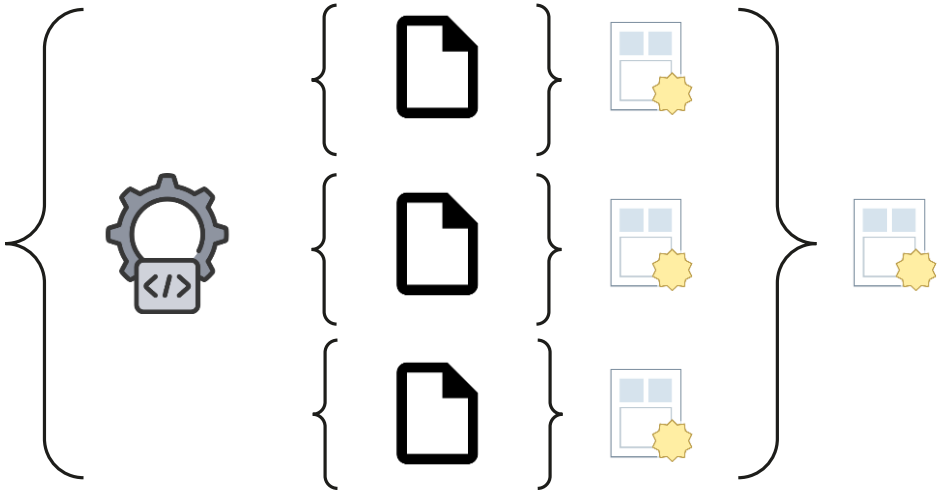
Integrity Measurement Architecture (IMA)



IMA Limitation



Current format of RPMs



RPM format required by IMA
(feature accepted for Fedora 37 [1])

- IMA cannot reuse existing information due to the different granularity
- Mass rebuild of a Linux distribution needed to support file signatures

[1] https://fedoraproject.org/wiki/Changes/Signed_RPM_Contents

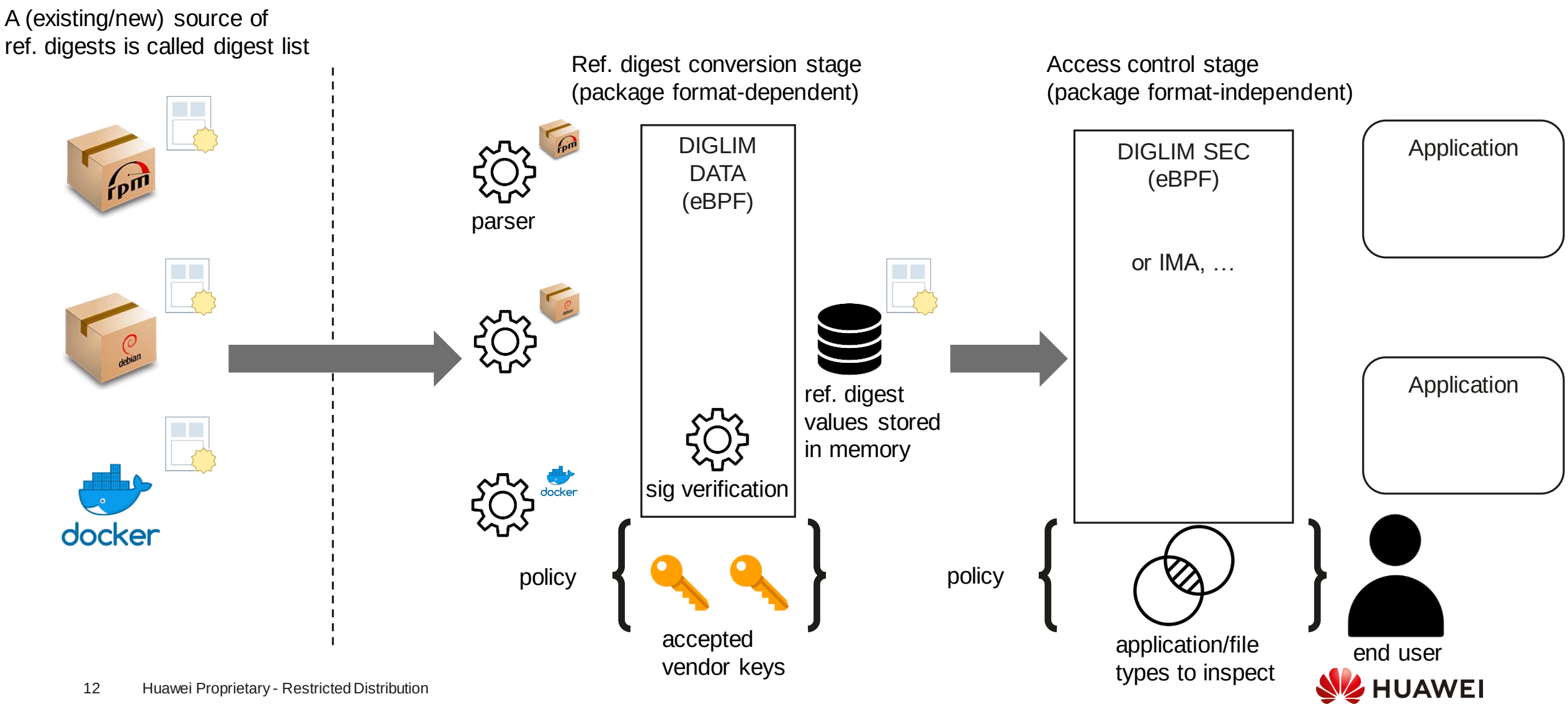
More IMA Limitations

- Files in the initial ram disk cannot be appraised due to the lack of xattr support
  - Huawei's openEuler is using a non-upstream patch since openEuler 20.09 (released in September 2020)
  - https://www.mail-archive.com/bug-cpio@gnu.org/msg00641.html

- Unusual default policy: appraise only files owned by root
  - Changing file owner is sufficient to circumvent this policy (if there is no metadata protection)

- File metadata are not protected, even with the Fedora 37 feature
  - Unclear how this problem will be addressed, another signature?
  - Metadata protection (experimental) supported since openEuler 20.09 with our IMA Digest Lists extension

Digest Lists Integrity Module (DIGLIM) eBPF



HUAWEI

DIGLIM eBPF Tradeoff

Pros

+ Not constrained to a specific data format

+ Extensible, new parsers can be implemented

+ Linux distributions can do appraisal verification with almost no modification

+ More efficient, one signature verification for all the files in a package

Cons

- Memory occupation
  - Currently eBPF map fully preallocated
  - Numbers not too bad with in-kernel implementation (208K of digests + 556K of indexes, Fedora minimal)

- Small delay at boot for parsing digest lists
  - Depends on the number of digest lists
  - Compensated by faster verification (lookup vs sig ver)
  - Overall good performance with in-kernel implementation (measurement+appraisal of 5896 files in ¼ of the time, compared to IMA)

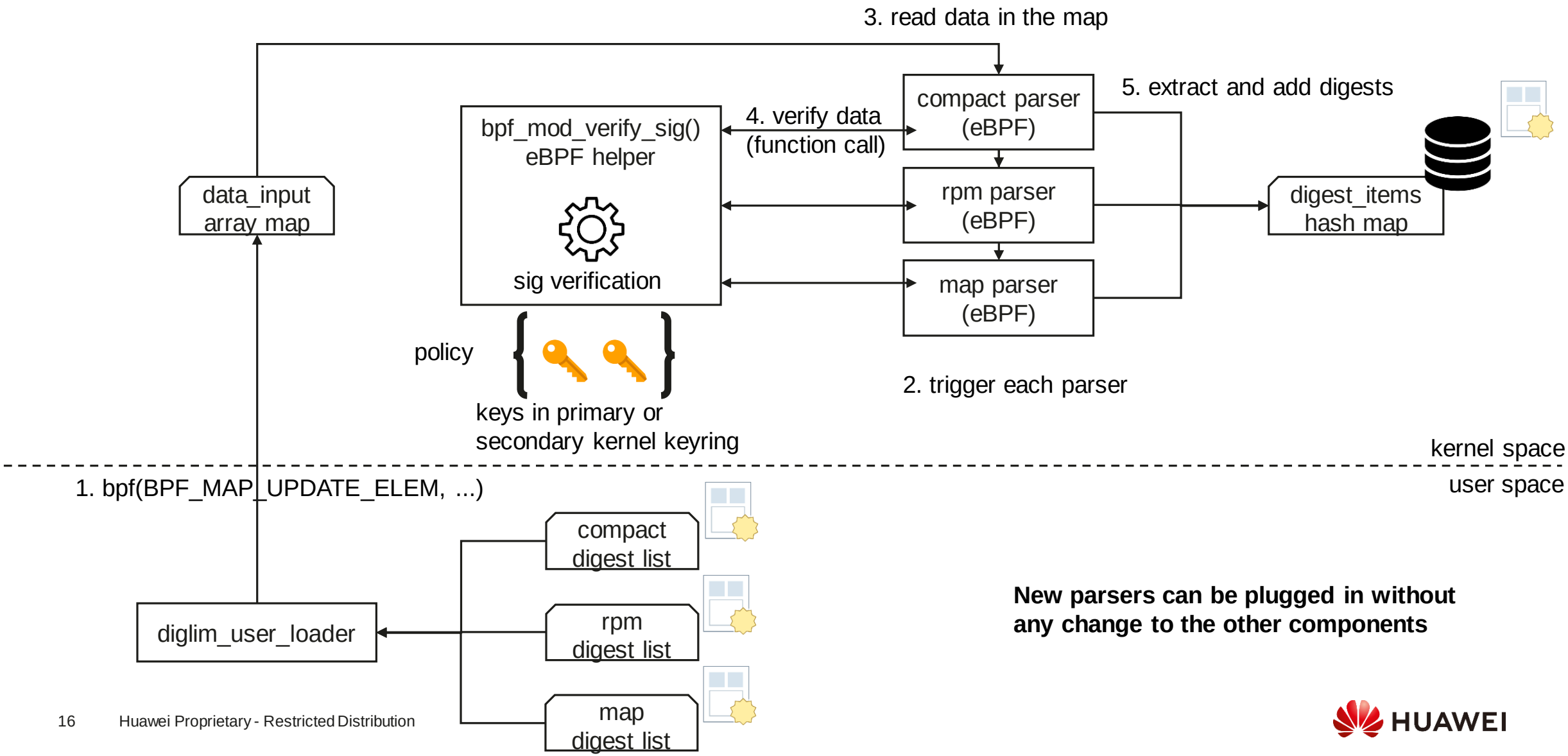
Why eBPF?

- IMA Digest Lists not upstreamed (too invasive)
- DIGLIM not upstreamed (minimal changes to IMA, however no answer at the third version of the patch set)
- To be honest, I didn’t consider eBPF as an option until I started studying it
- The idea of making DIGLIM available without the constraint of upstreaming it in the kernel made me study it

What is eBPF?

- It is a framework to add more functionality to the kernel without modifying the kernel
- It loads sandboxed programs, after statically checking their code
- Integrity assurance: no illegal memory accesses and finite program termination
- Confidentiality assurance: restricted access to memory
- bpf LSM: Linux Security Module to plug in sandboxed eBPF programs
  - > Lets eBPF programs implement defined security hooks (e.g. file open, execution, …)
  - > Lets eBPF programs make a decision on a requested operation based on information provided by the kernel (e.g. file digest calculated by IMA)
  - > Don’t require eBPF programs to be registered to the LSM framework (LSMs need to be compiled built-in in the kernel)

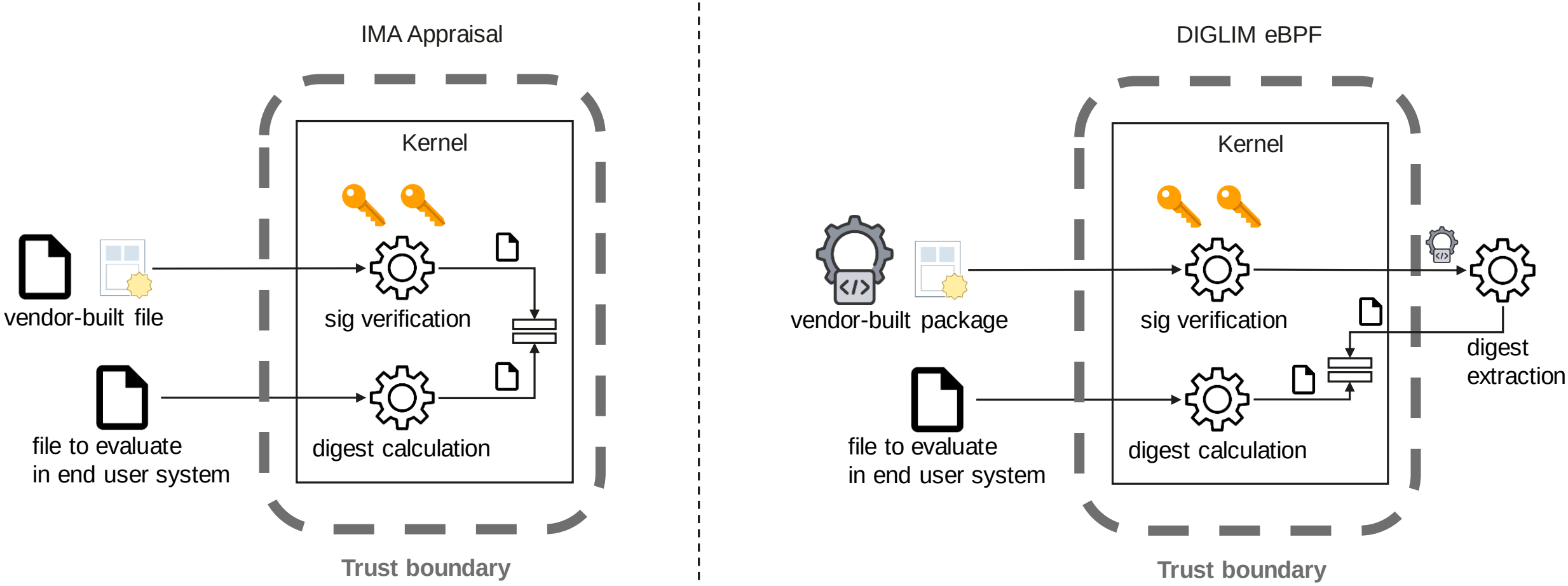
DIGLIM eBPF Architecture – Conversion Stage



New parsers can be plugged in without any change to the other components

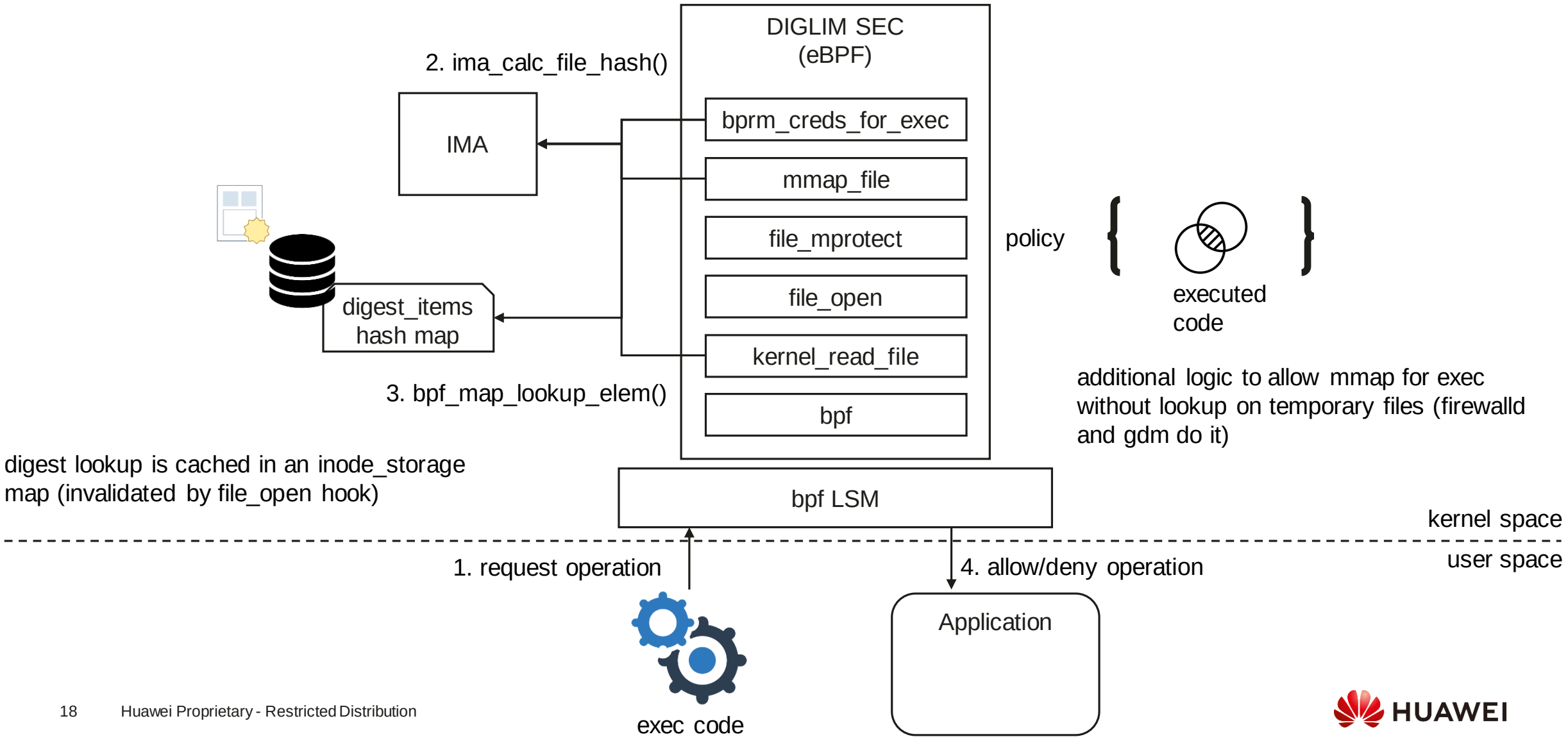
HUAWEI

Why Parsers Are Not in User Space?



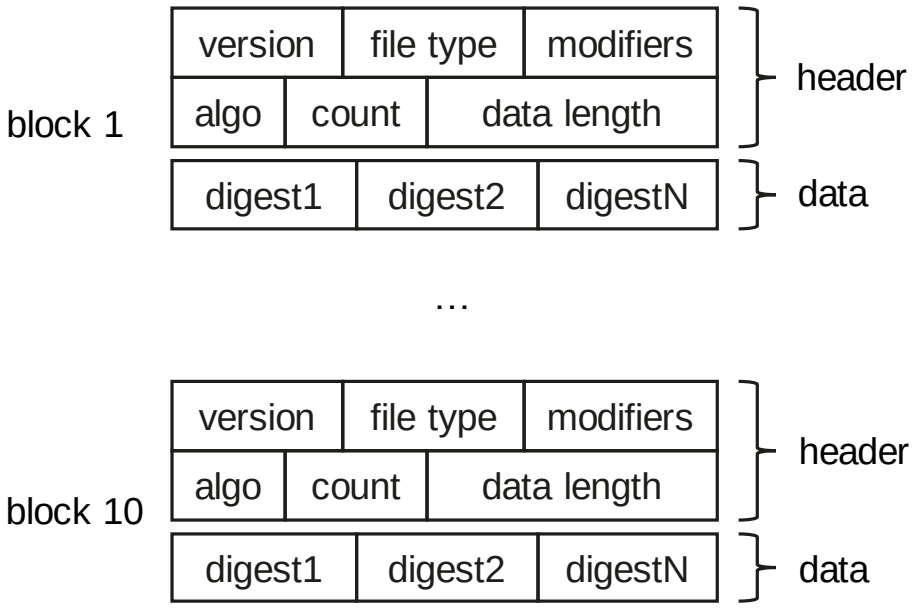
Question: how to extend the trust boundary to the additional component required by DIGLIM eBPF?

DIGLIM eBPF Architecture – Access Control Stage



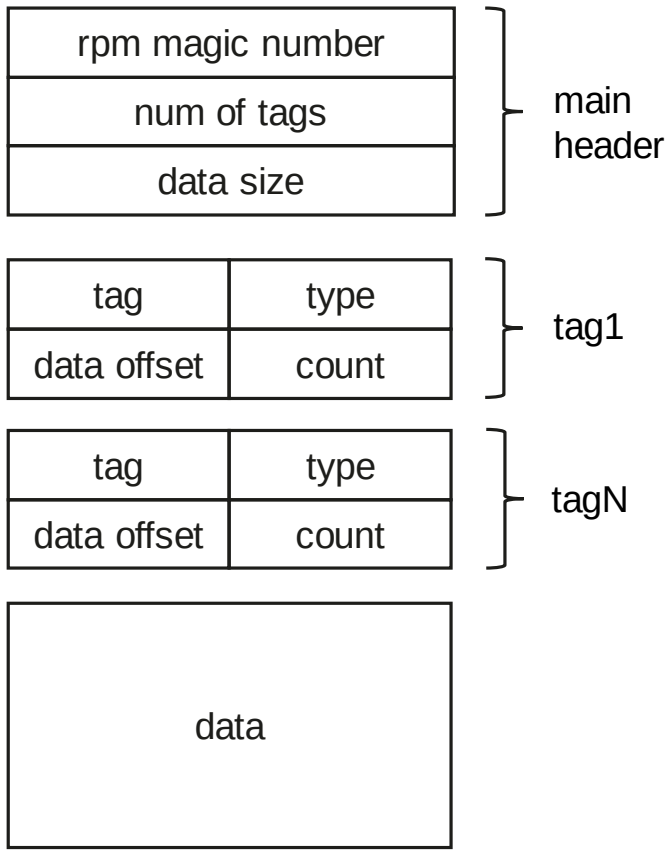
Parsing Packages with eBPF – Is It That Complex?

2 nested loops
- 0..9 blocks
- 0..count - 1 (for each block)



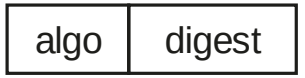
compact digest list (LOC: ~130)

3 sequential loops
- 0..num of tags - 1 (get tag count and data offset)
- 0..dirnames count - 1 (to get the dir of each file)
- 0..files count - 1



rpm digest list (LOC: ~400)

simple sequential read



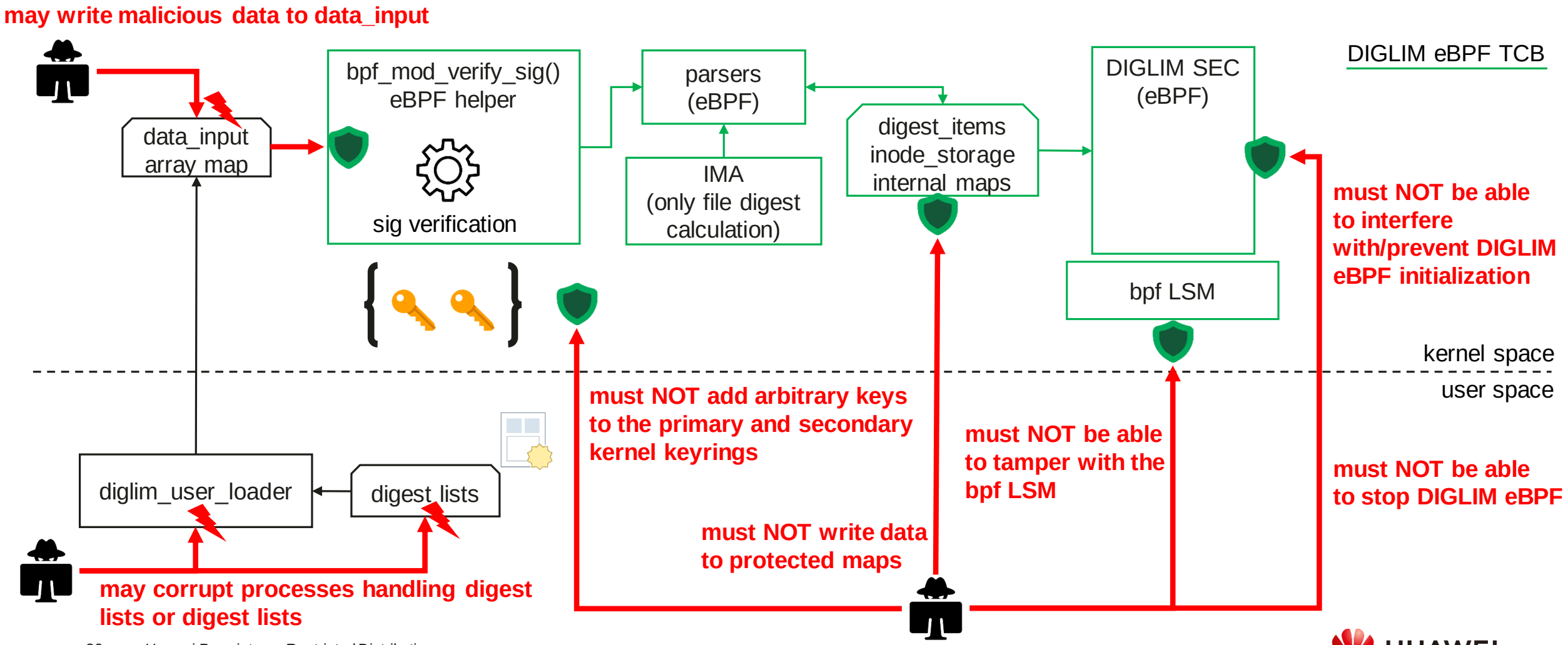
map digest list (LOC: ~40)

security module (LOC: ~250)

total code to verify/trust (LOC: ~820)

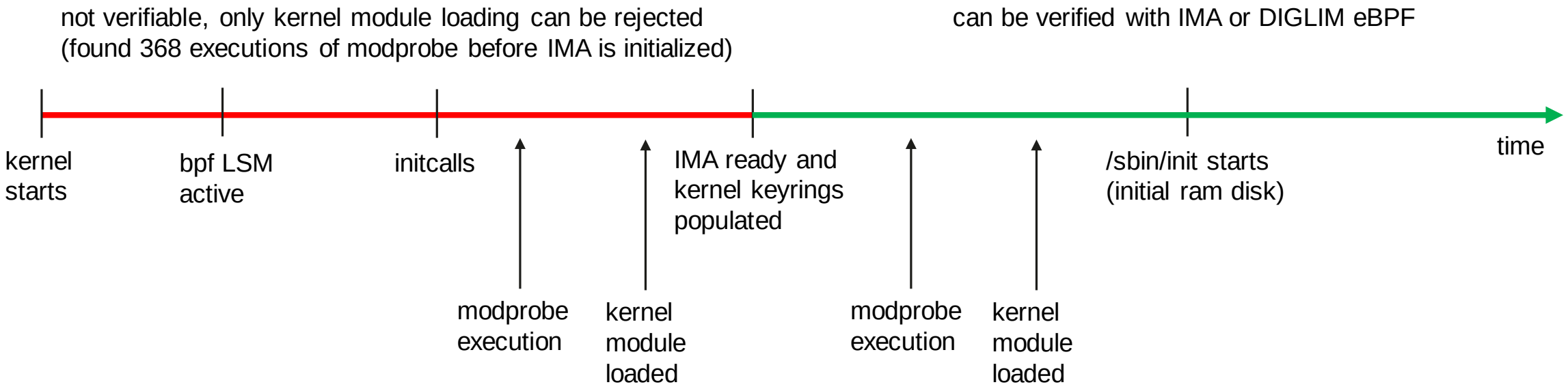
HUAWEI

DIGLIM eBPF Threat Model



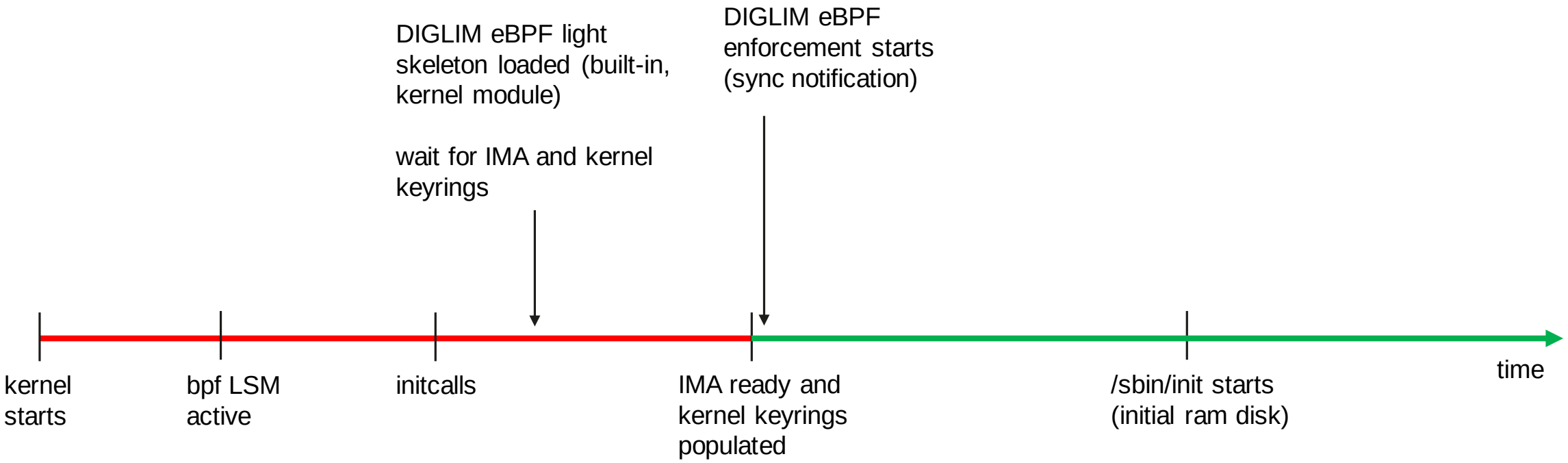
HUAWEI

DIGLIM eBPF Initialization – When?



If some operations that the policy would deny are allowed, the system security state is unknown

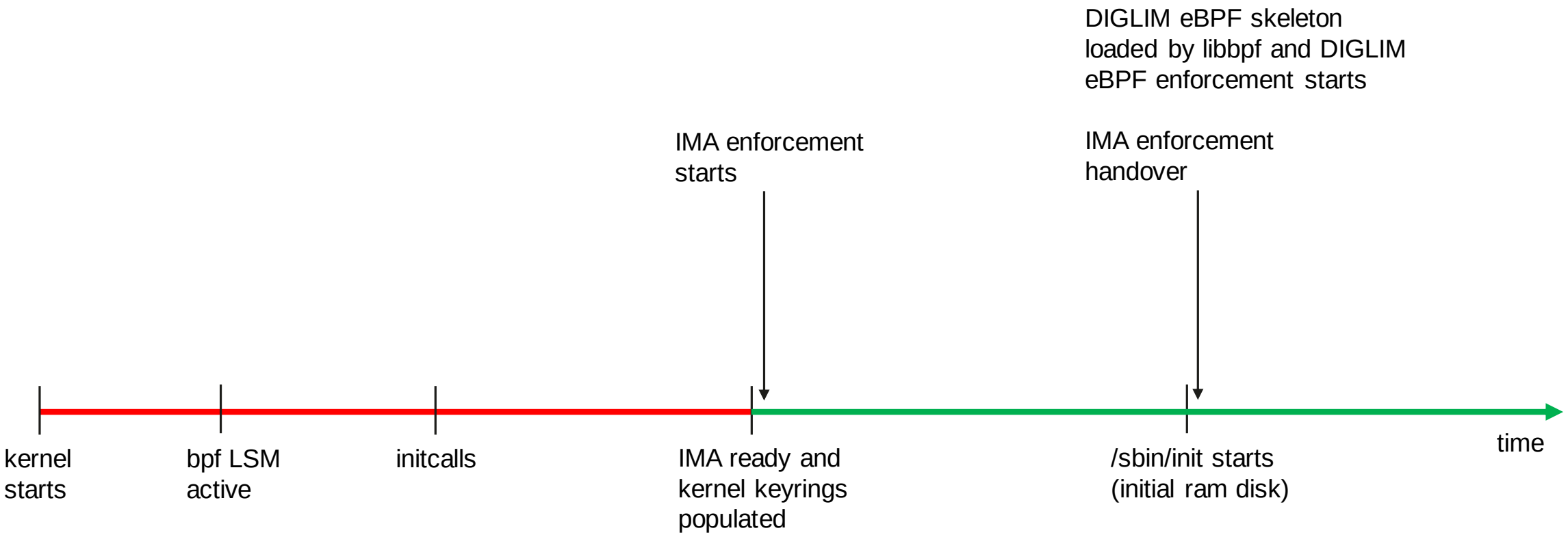
DIGLIM eBPF Initialization Proposal – Built-in/Kernel Module



If built-in, any change to DIGLIM eBPF must be accepted in the upstream kernel

If kernel module, DIGLIM eBPF development could be out-of-tree (requires eBPF to load it)

DIGLIM eBPF Initialization Proposal – As /sbin/init



Requires an additional enforcement mechanism (e.g. IMA) to verify executable code before /sbin/init is executed

DIGLIM eBPF Requirements for Linux Distribution Vendors

- Built-in
  - No code integration needed (follow upstream)
- Kernel module
  - Invocation code is in upstream
  - Take kernel module and eBPF programs from DIGLIM eBPF repository, and sign them
- User space
  - Provide IMA enforcement until DIGLIM eBPF starts
  - Take user space loader and eBPF programs from DIGLIM eBPF repository, and add file signatures (also to files in the initial ram disk)
- Common task for all cases
  - Sign digest of digest list loader (DIGLIM eBPF enforcement active)

No mass rebuild, other packages can be used as they are (even those already released)!

Demo

Conclusions

- Enforcing verification of executable code is not a very difficult challenge per se, but requires support in the ecosystem
- Fedora 37 feature introducing file signatures is a very good news, integrity feature available for users
- DIGLIM eBPF seems a more flexible and scalable option requiring much less effort for Linux distribution vendors
- Other LSMs, eBPF programs and IMA could benefit from DIGLIM's repository of authenticated reference values
- Some minor dependencies need to be added to the kernel (e.g. support for PGP keys and signatures)
- Hope to make some progress with you at the summit!

Links

- Kernel patches:
  - https://github.com/robertosassu/linux/tree/bpf-diglim-v1

- DIGLIM
  - https://github.com/robertosassu/diglim-ebpf/tree/devel-v0.1.3

Thank you.

Bring digital to every person, home and organization for a fully connected, intelligent world.