26/10/2018

A Simple Protocol for Remote Attestation of System Integrity

Roberto Sassu

Cyber Security and Privacy Lab (CSPL)

Linux Security Summit Europe 2018

Outline

- Problem
- Background
- Our Proposal
- Conclusions

Remote Attestation – Problem

Remote attestation (RA) definition: integrity evaluation done by a remote verifier to check whether a system can accomplish its tasks as expected

Evaluating the integrity of OS (kernel + applications + their state) is **very** complex

- Reference measurements and verification services are not available
- It is unclear what information must be supplied to verifiers and how to analyze them

RA cannot be easily integrated into existing products because

- A dedicated server must be added to the infrastructure
- Two separate protocols must be implemented for secure communication and attestation

Background

Integrity is the expectation that a system/application behaves as defined by the developer

Measurement from Core Root of Trust for Measurement (CRTM) up to the application

Measurement at OS-level (Linux) is done by Integrity Measurement Architecture (IMA)

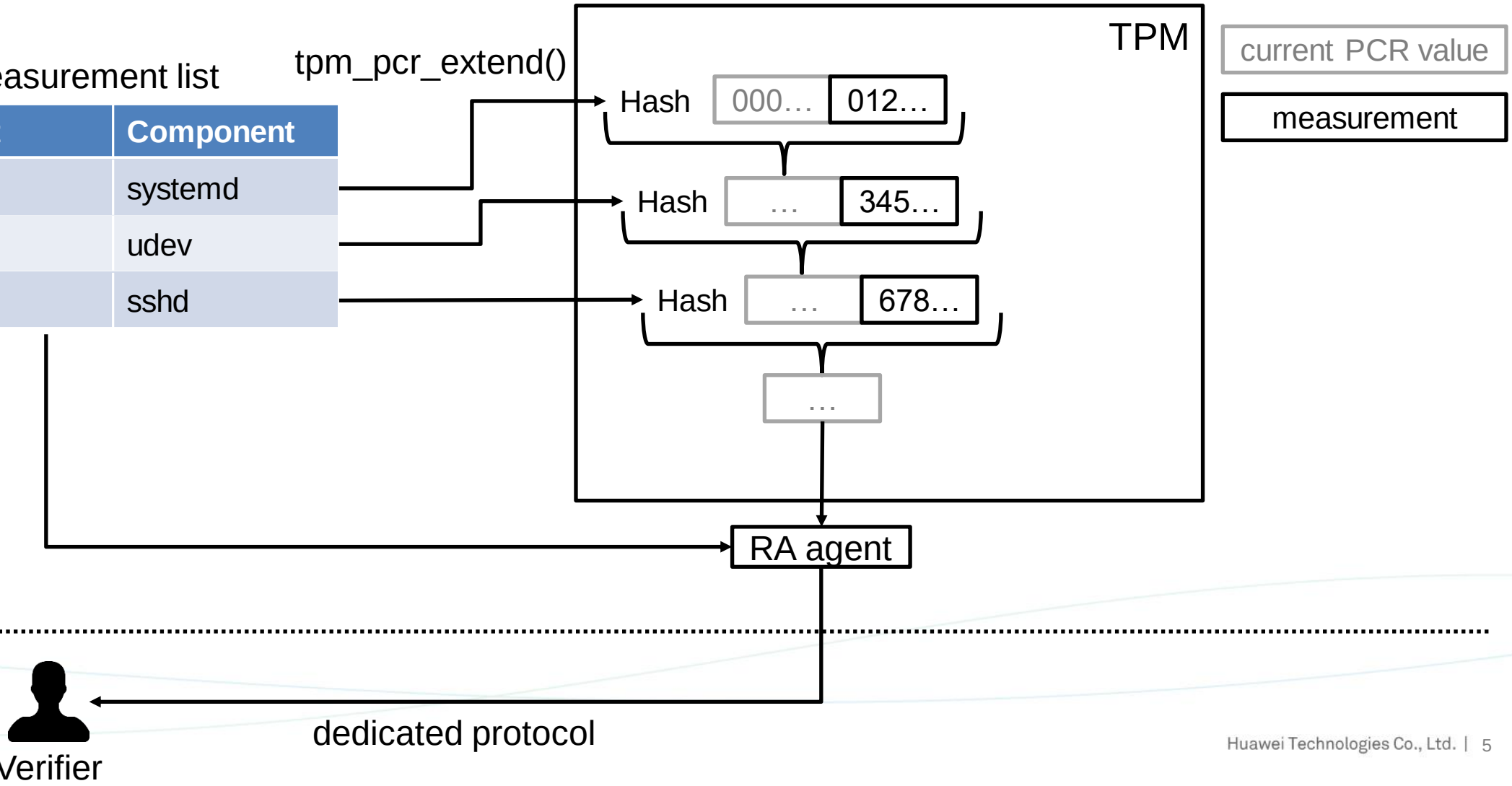
Evaluation done by comparing actual measurements with reference values

Background – Explicit RA

PCR: Platform Configuration Register

IMA measurement list

PCR	Digest	Component
10	012…	systemd
10	345…	udev
10	678…	sshd

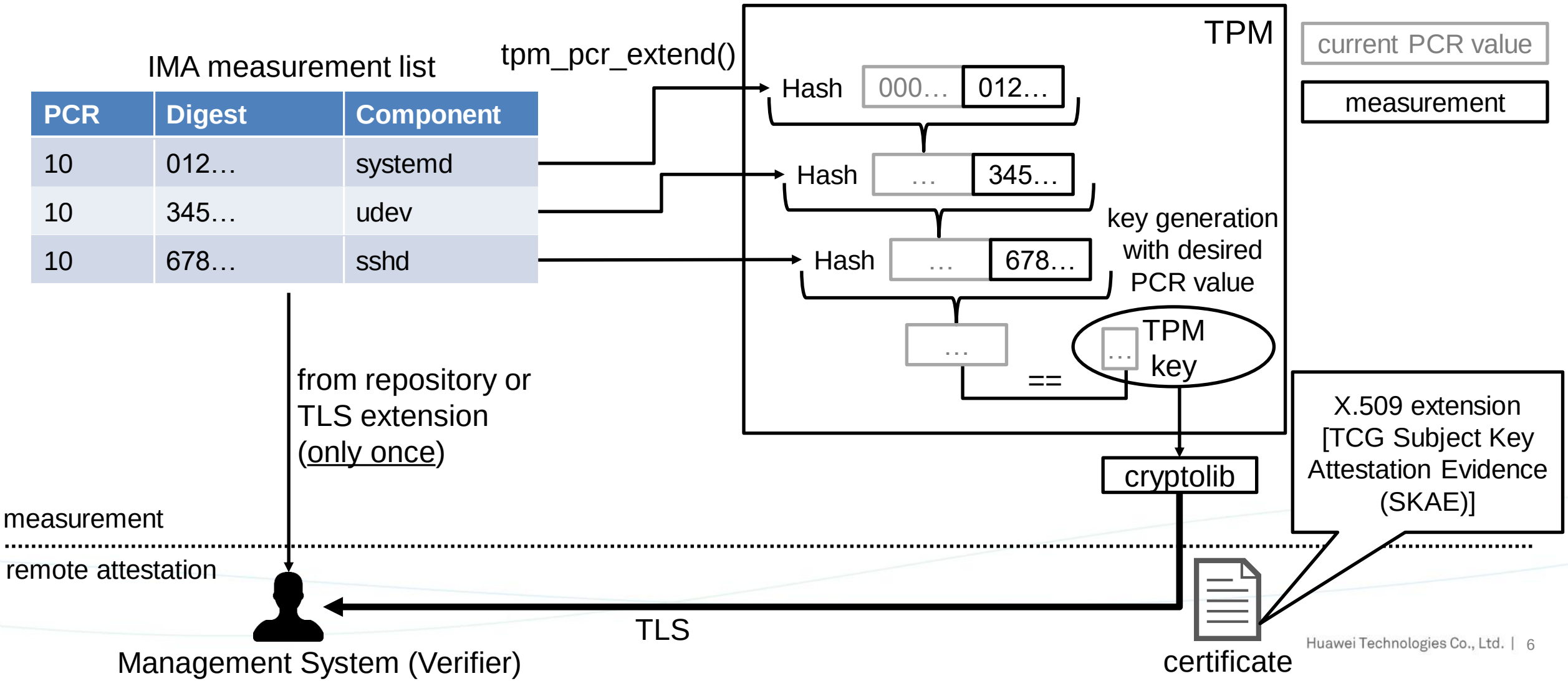


Background – Implicit RA

PCR: Platform Configuration Register

IMA measurement list

PCR	Digest	Component
10	012…	systemd
10	345…	udev
10	678…	sshd



Simple RA Protocol with Implicit RA

Implicit RA is more suitable for integration into existing products, as it only requires

- Switching from software keys to TPM keys
- An additional verification of an X.509 extension (SKAE)

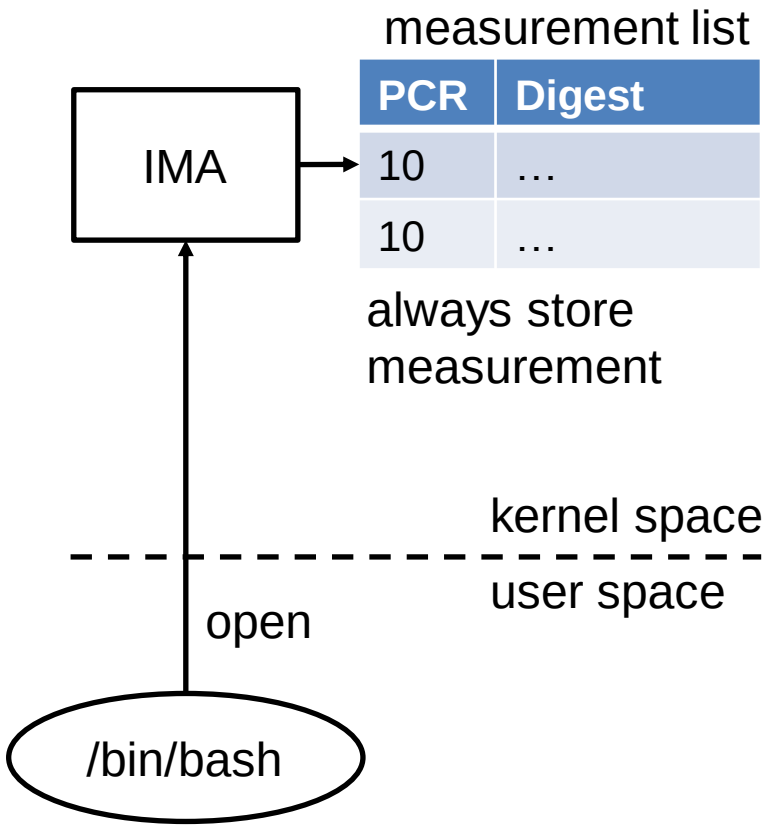
Problem: IMA PCR is not predictable

- Depends on which and when files are executed
- Depends on the content of mutable files

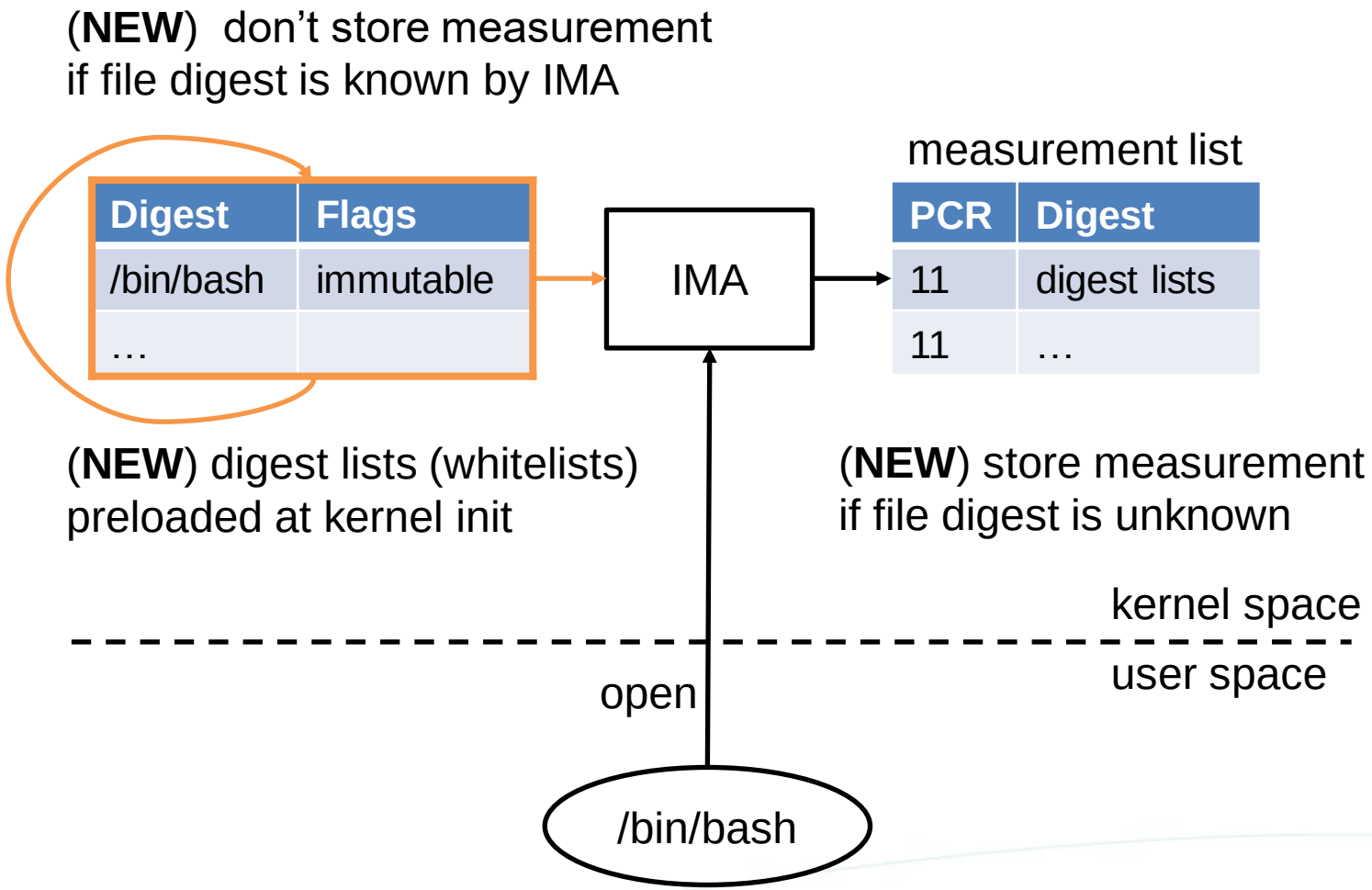
Solution: make IMA PCR predictable

- IMA Digest lists extension to address unpredictable file access
- Enhanced Policy-Reduced Integrity Measurement Architecture (PRIMA) to handle mutable files

IMA Digest Lists

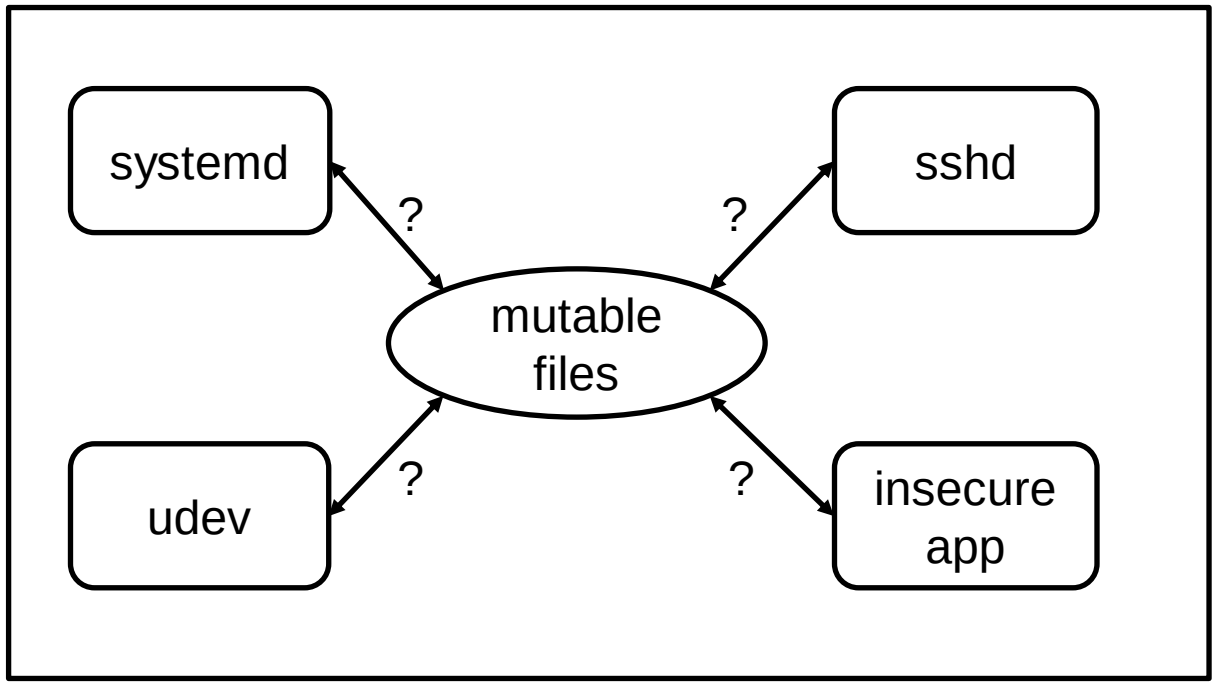


current behavior (PCR #10)



current behavior (PCR #10)
new behavior (PCR #11)

Mutable Files in the IMA Measurement List



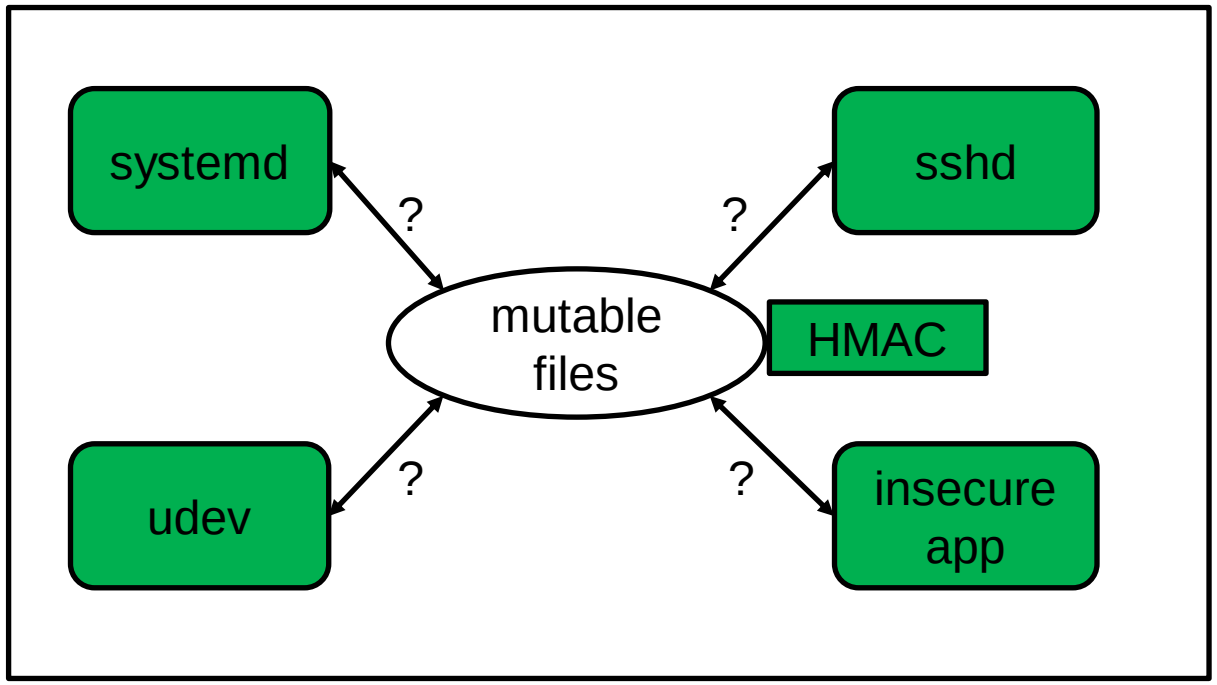
PCR	Digest	Component
11	known	digest lists
11	**unknown**	mutable file

IMA measurement list

Verification fails

How to deal with mutable files?

Alternative Solution for Evaluation of Mutable Files

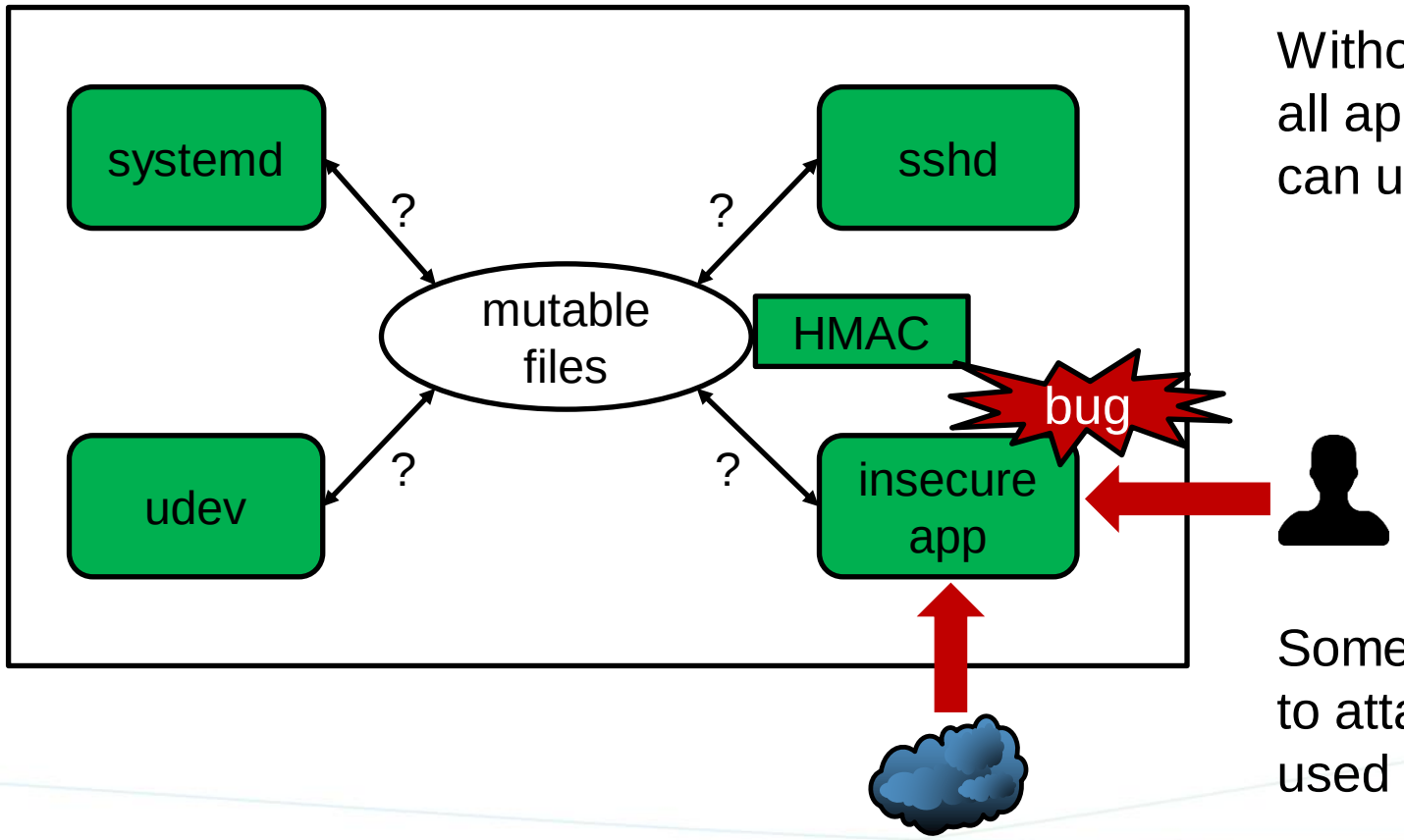


Evaluate the processes writing mutable files, instead of measuring them

If processes are not compromised, they must have updated mutable files correctly

Protect mutable files against offline attacks with an HMAC (key not sealed to OS!)

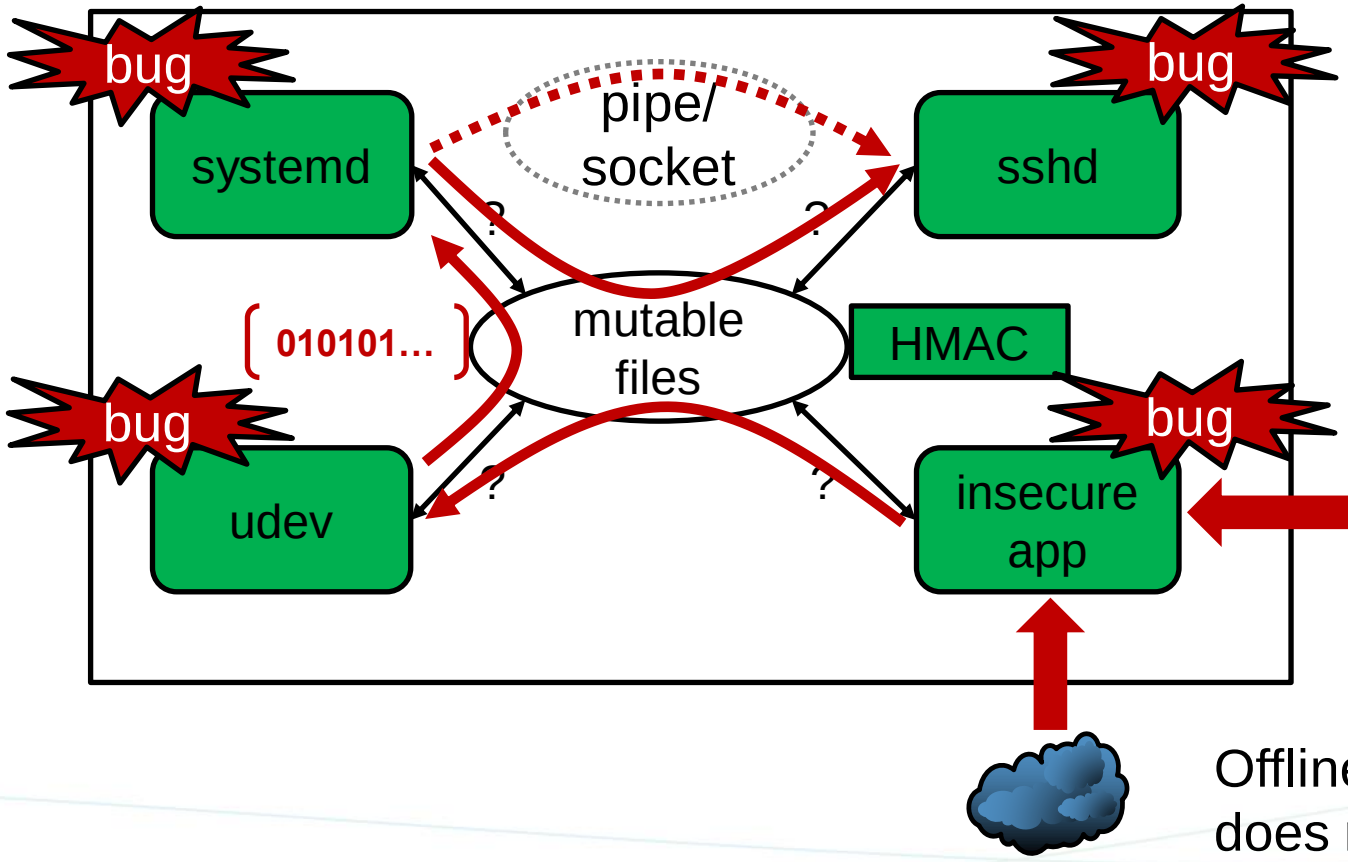
Unknown Impact of Process Actions without MAC



Without Mandatory Access Control
all applications with enough privileges
can update mutable files

Some applications are more susceptible to attacks during execution and can be used as an entry point

True System State Differs from Reported State

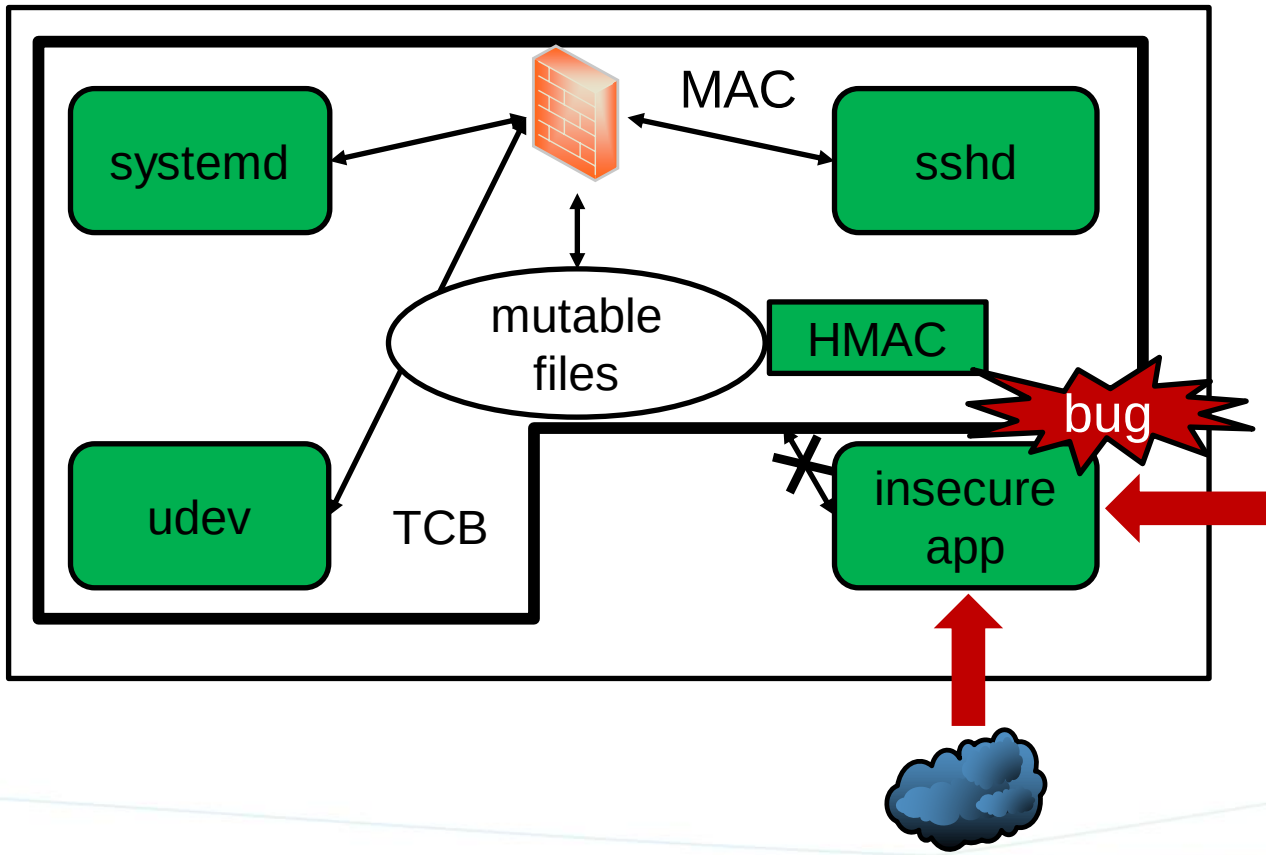


An attacker can exploit an insecure app and make it produce malicious byte sequences

These byte sequences can be used to exploit bugs in other apps, without being detected

Offline protection (HMAC) does not help to detect such malicious file modifications

Protect Mutable Files with Mandatory Access Control



MAC can protect the critical part of the system (TCB) by enforcing an integrity policy, such as Biba or Clark-Wilson

Mutable files inside the TCB can be written only by the TCB

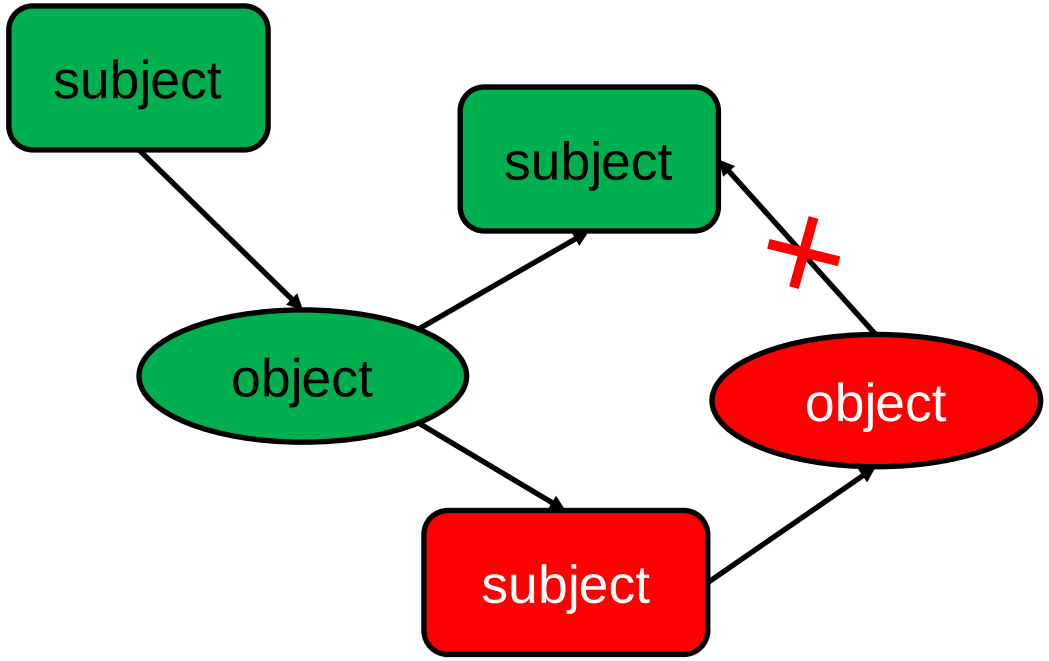
MAC and integrity policy become part of the evidence to be sent to verifiers [1]

[1] Policy-Reduced Integrity Measurement Architecture (PRIMA)
Trent Jaeger, Reiner Sailer, and Umesh Shankar

Integrity Models – Biba vs Clark-Wilson

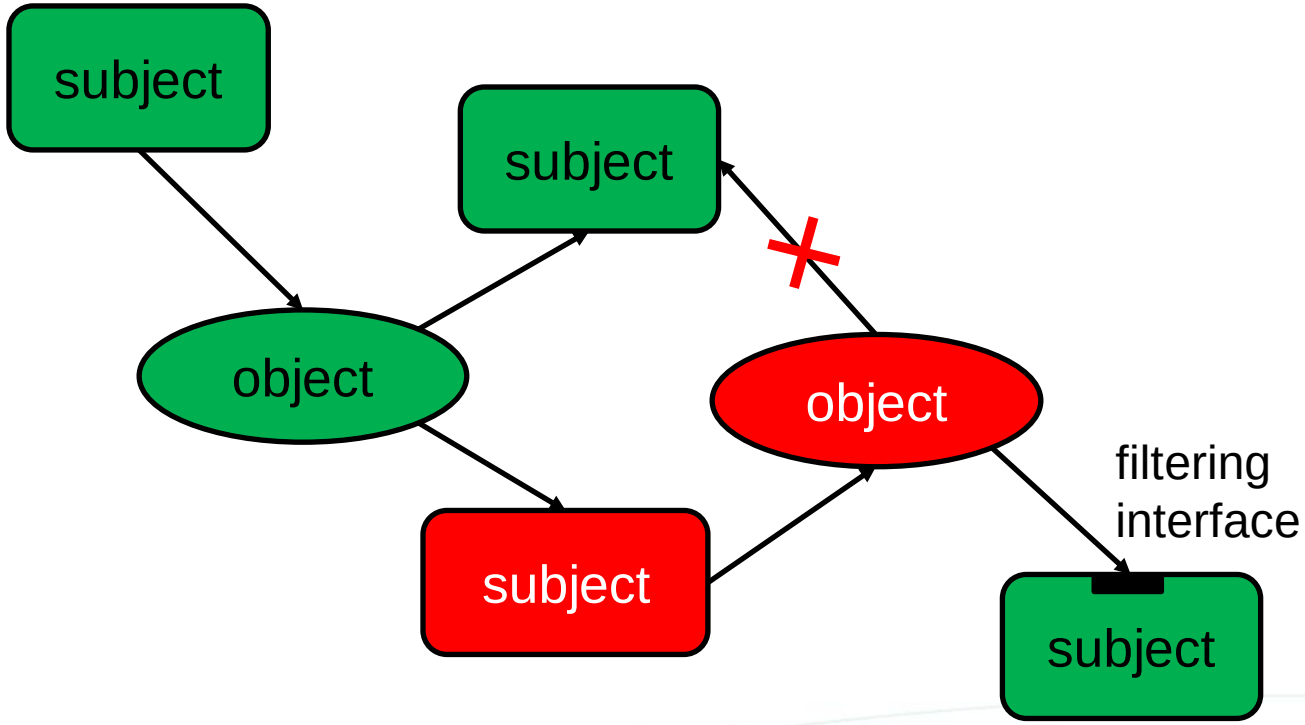
High integrity

Low integrity



Biba model

Read from filtering interface allowed if code can reliably handle malicious data



Clark-Wilson model

PRIMA Overview and Drawbacks

PRIMA finds from the SELinux policy a minimal TCB that satisfies Clark-Wilson requirements

Only code and immutable files that belong to the TCB must be measured

Drawbacks that limit application in the industry

- Unlikely to find a small TCB
  - A generic policy (e.g. in Fedora) considers all the possible application usage scenarios
  - ~100,000 allow rules, only ~2.5% of rules are requested by a running system
- TCB must be adapted for each specific use case
- Offline attacks are not taken into account

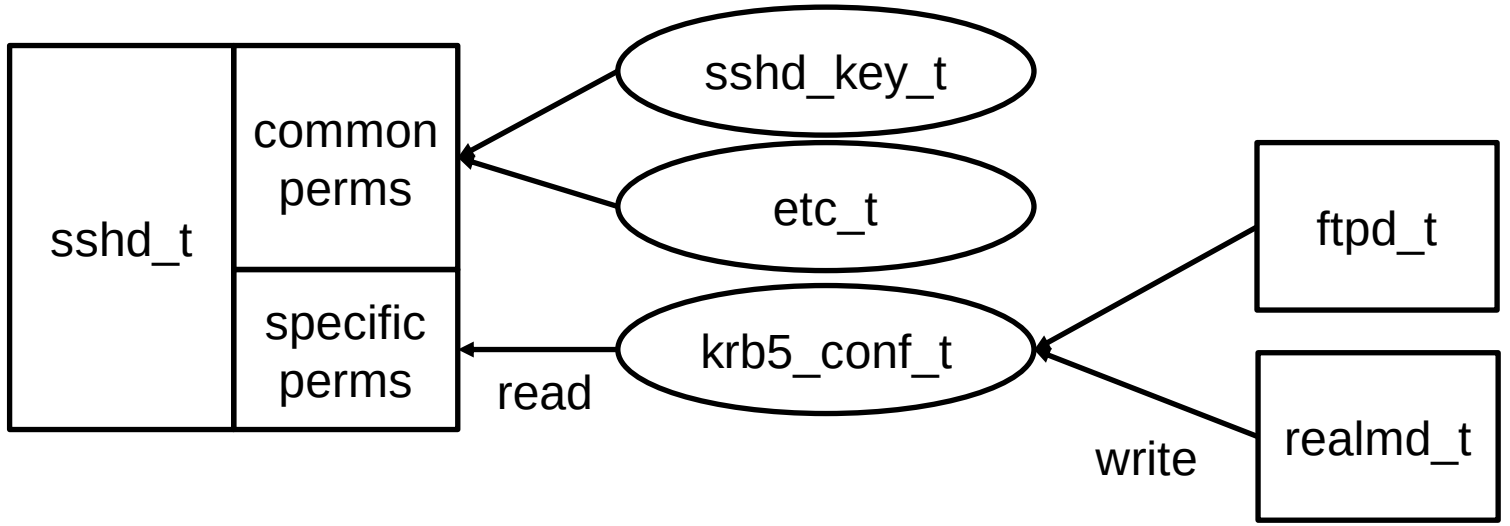
Our Proposal to Simplify and Complete PRIMA

- Reduce TCB size by considering processes interactions discovered on the target system
  - With a new Linux Security Module (LSM), called Infoflow LSM

- Detect malicious updates of mutable files throughout their entire lifetime
  - PRIMA does not guarantee that TCB protection was enabled before reboot

Reduce TCB size

Example: information flow analysis for sshd (included in the TCB)

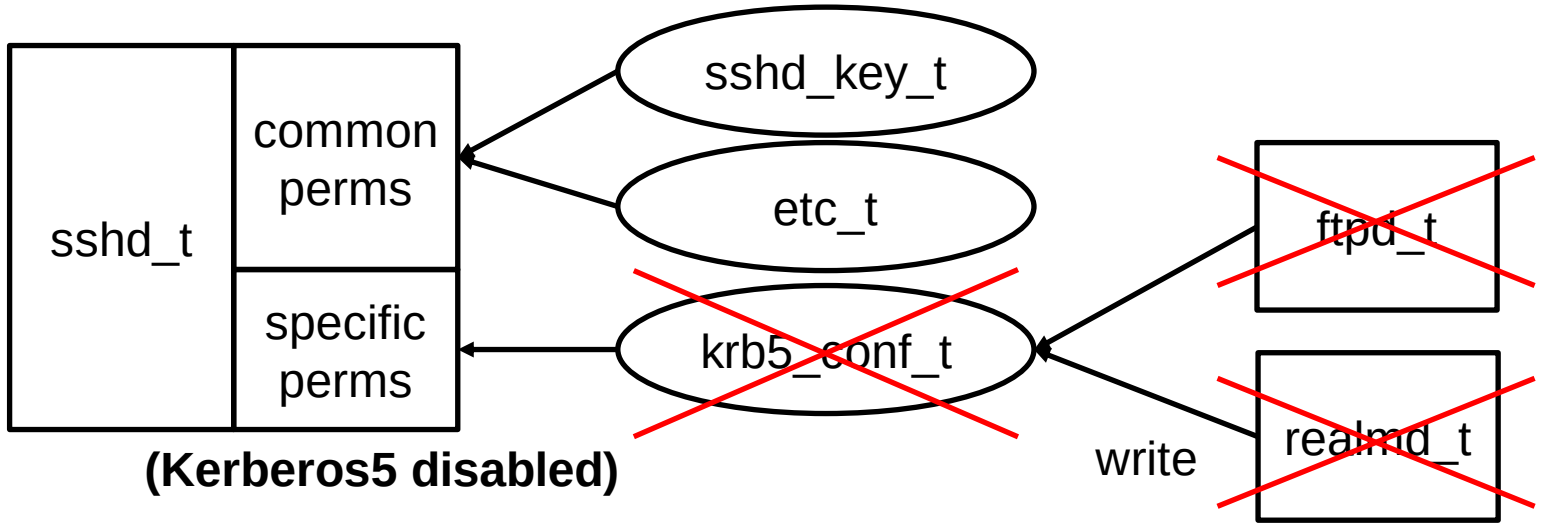


With PRIMA, Kerberos5 would be added to the TCB (high risk) or would have to be manually excluded (too much effort)

Permissions taken from SELinux policy of Fedora 27

Reduce TCB size

Example: information flow analysis for sshd (included in the TCB)



With our proposal, Kerberos5 is automatically excluded

Permissions taken from SELinux policy of Fedora 27

Detect Malicious Updates of Mutable Files

State of the art

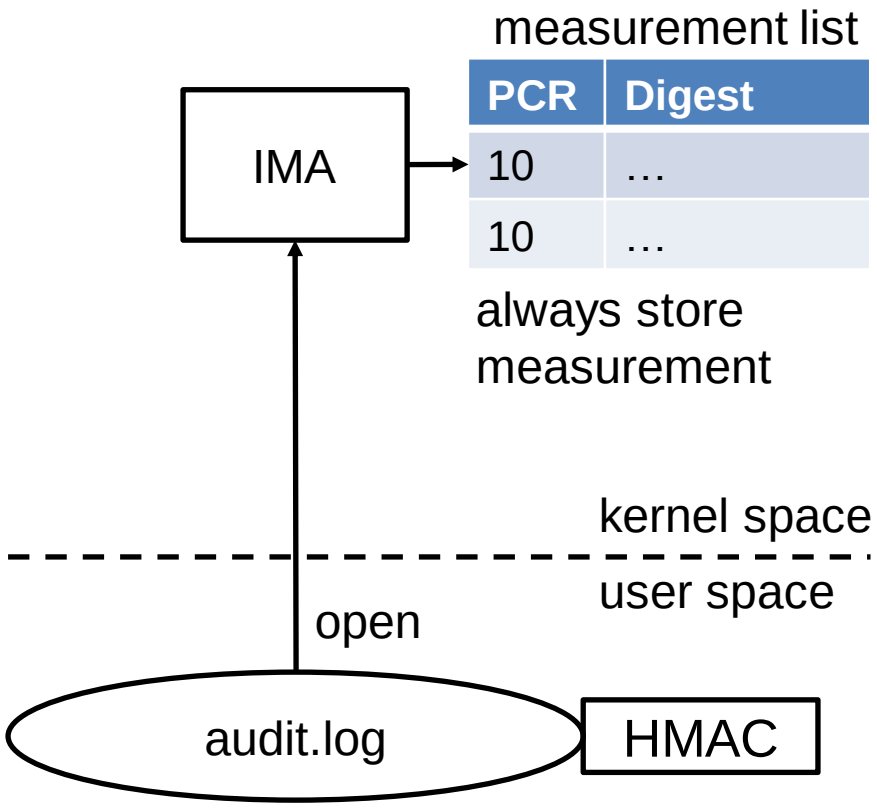
- IMA Appraisal/EVM protect the integrity of data/metadata against offline attacks

- EVM key is sealed with TPM, but not to OS (IMA PCR unpredictable)
  - EVM key can be used when TCB protection is disabled

Our proposal

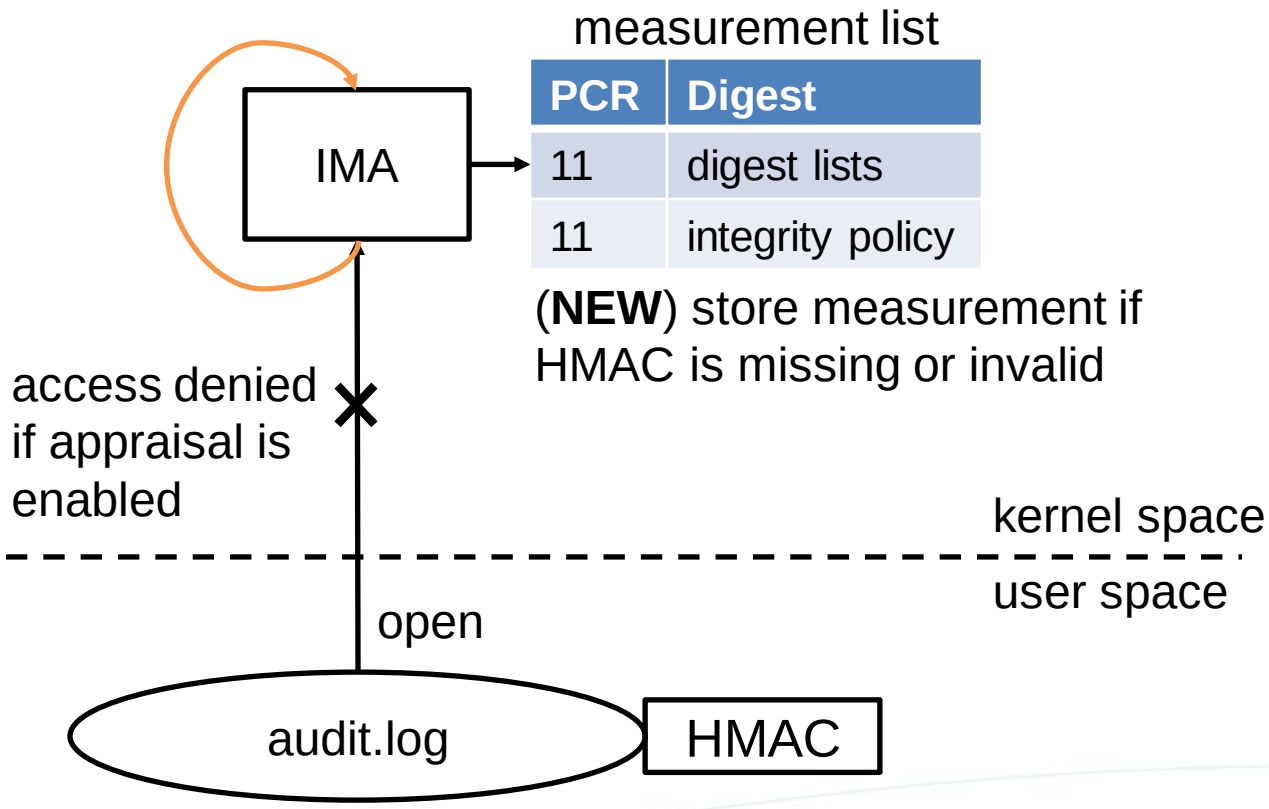
- Seal EVM key to predictable IMA PCR, extended with digest lists and integrity policy
  - EVM key can be used **only** when TCB protection is **enabled**
  - If TCB protection is disabled, the unsealed EVM key is **erased** before integrity violations
- A valid HMAC implies that the mutable file was updated by the TCB and code/data was known
  - With this guarantee, mutable files can be excluded from measurement

Exclude Mutable Files from Measurement



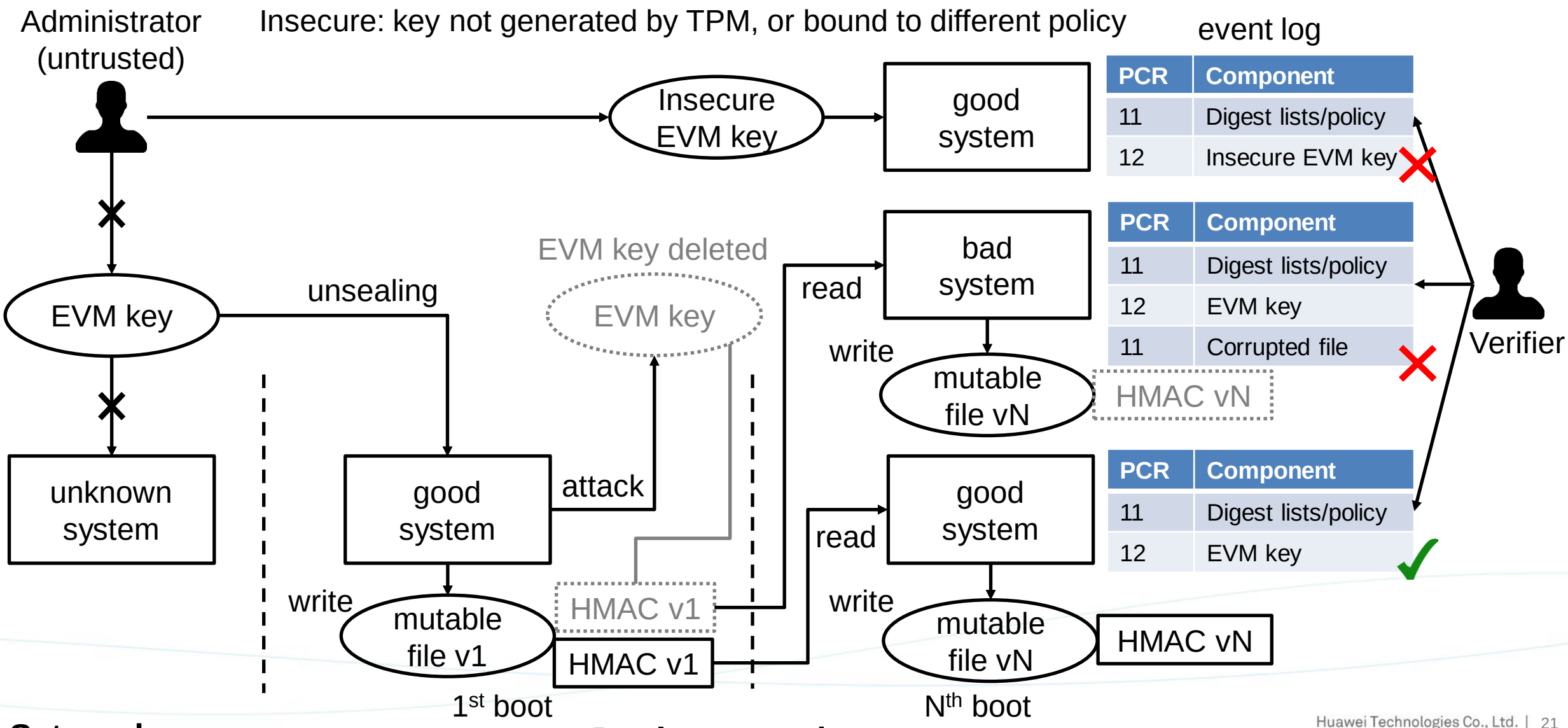
current behavior (PCR #10)

(**NEW**) don't store measurement if HMAC is valid and EVM key is sealed to OS

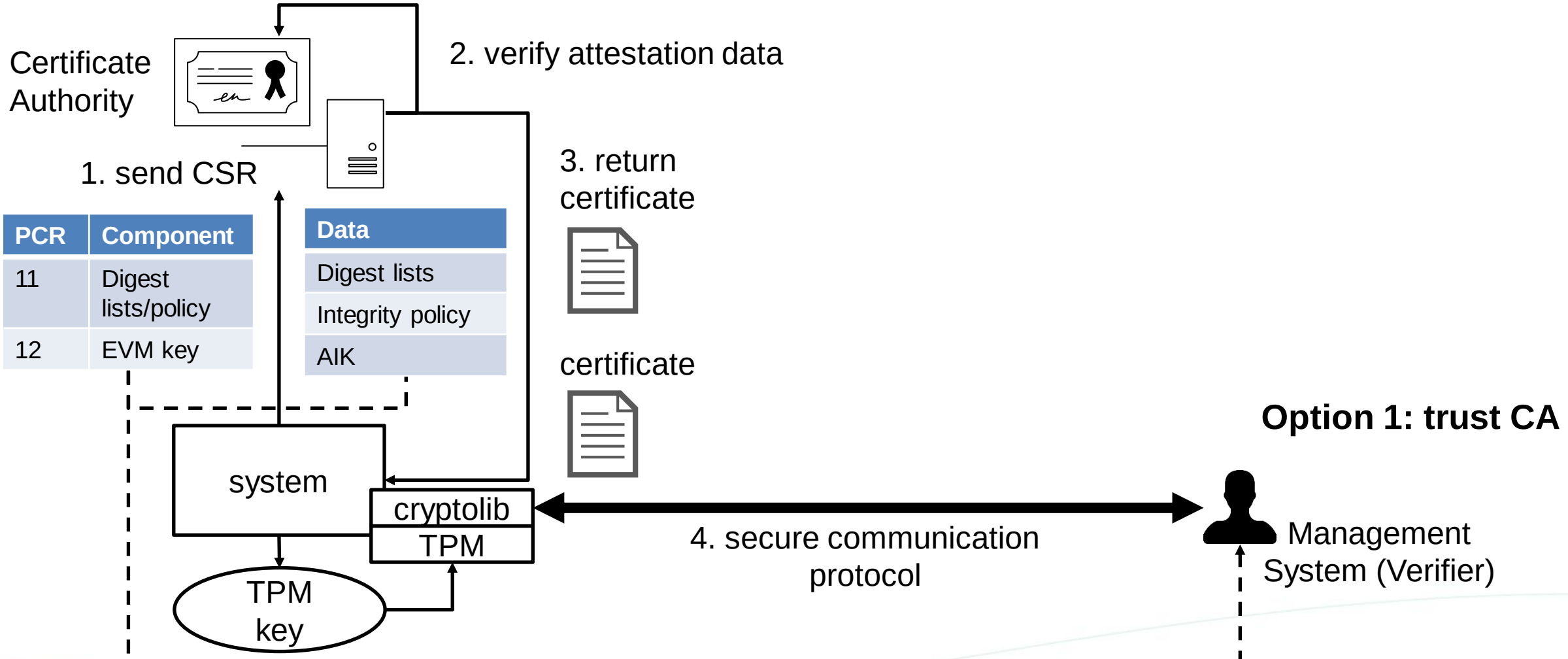


current behavior (PCR #10)
new behavior (PCR #11)

Chained Integrity Verification across Reboots

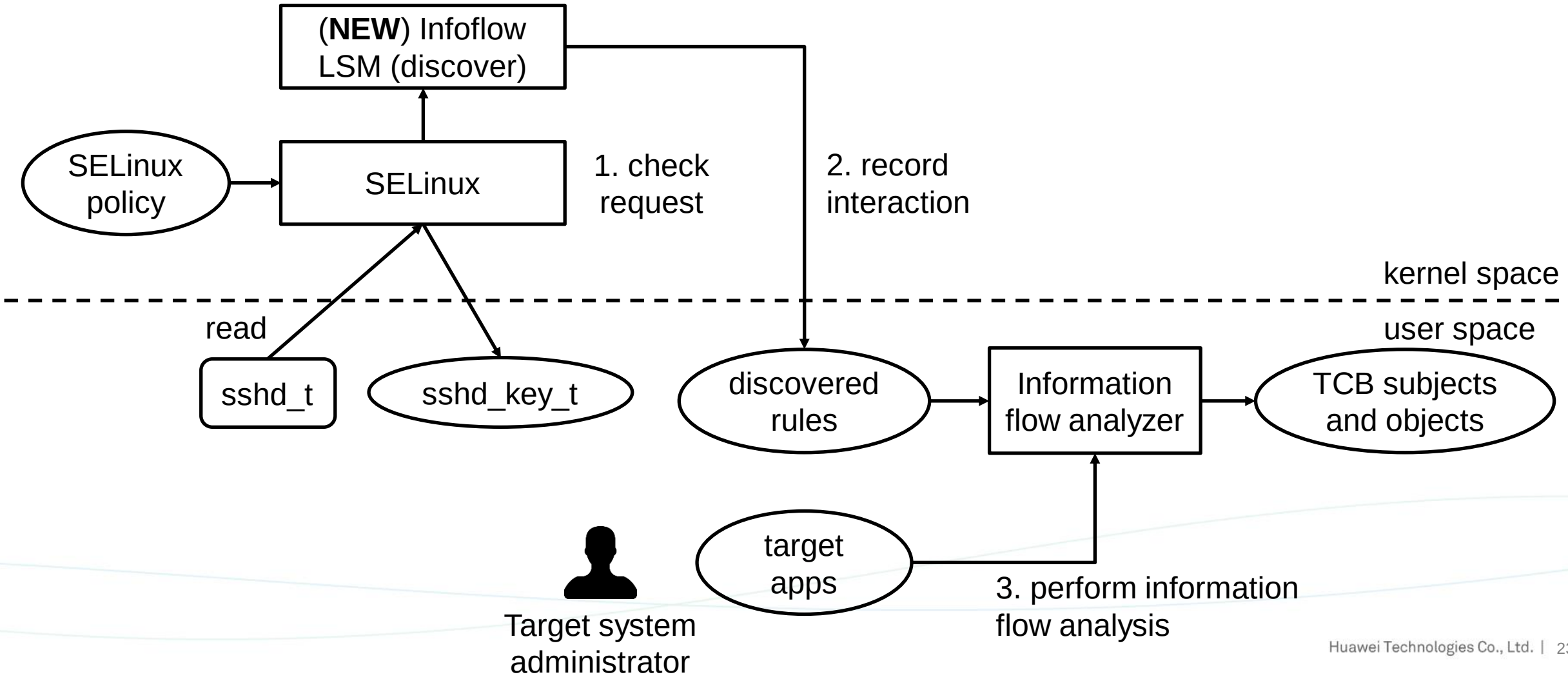


Implicit RA – Verification Options

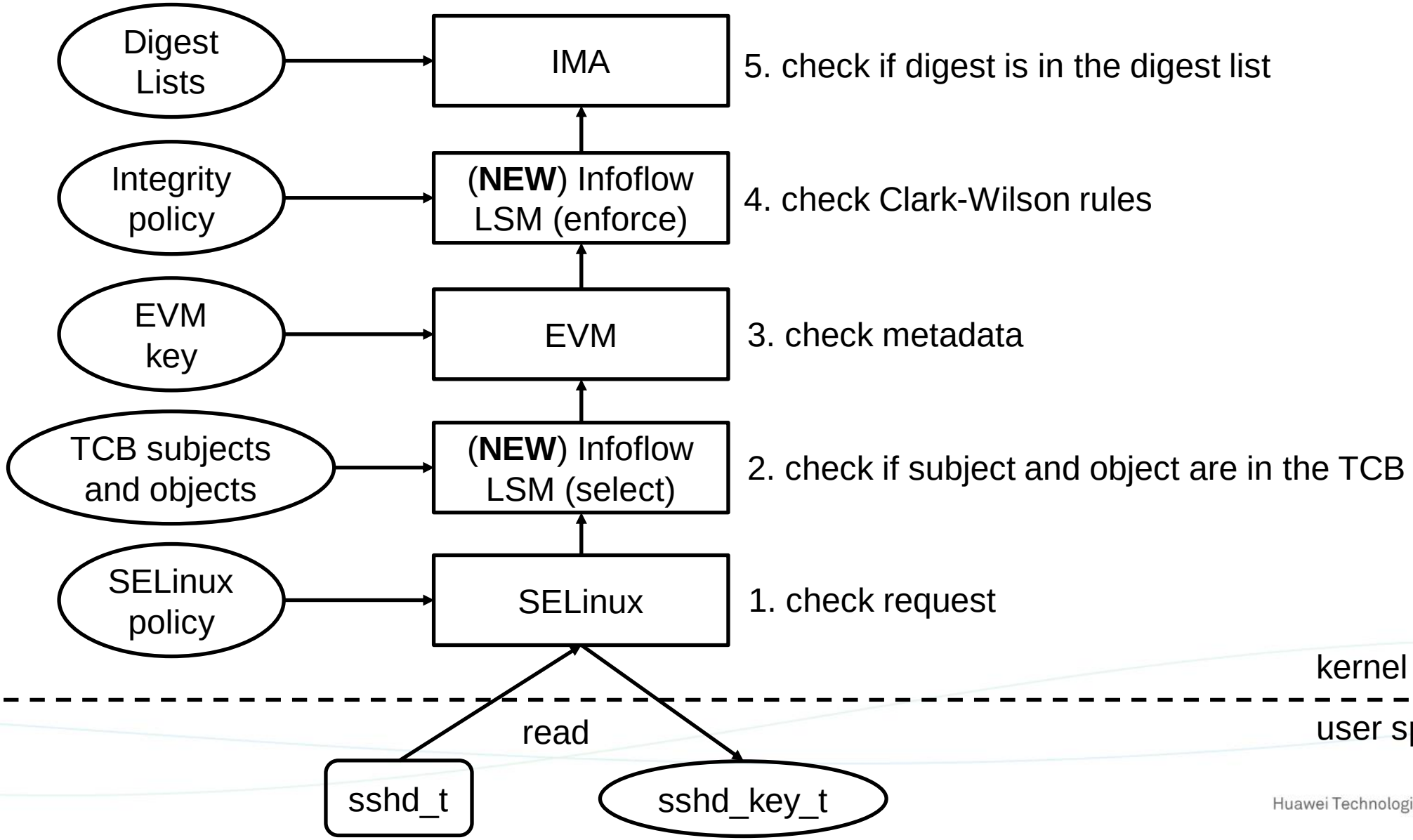


Option 2: verify SKAE with event log and data (from a repository/protocol extension) only once

Infoflow LSM Implementation – Setup Phase



Infoflow LSM Implementation – Deployment Phase



Source Code

Digest lists source code

- Kernel space: https://github.com/euleros/linux (tag: ima-digest-lists-v3)
- User space: https://github.com/euleros/digest-list-tools (tag: v0.2)

Binary packages for Fedora 27, openSUSE Leap 42.3

- Wiki: https://github.com/euleros/digest-list-tools/wiki

Digest lists overview

- https://develop.trustedcomputinggroup.org/2018/05/30/digest-lists-extension-for-linux-ima/

Conclusions

Currently, remote attestation is not widely adopted because

- Evaluating the integrity of the entire OS is very complex
- Existing products must be modified to include an additional protocol and dedicated server

Implicit RA is more suitable for integration into products, but currently not feasible because

- IMA PCR is not predictable
- Finding a TCB to protect mutable files requires significant effort
  - General purpose OSes are prioritizing backwards compatibility over integrity
  - Integrity models are often violated (e.g. ssh server reads data from the network)
  - For software images the TCB could be identified by system designers

We propose a solution that is both comprehensive and practical

- By evaluating the integrity of the entire OS, first with more strict assumptions on usability
- By lowering the requirements for integration with existing products (e.g. Network Management Systems)

FutureTPM Grant Agreement No. 779391

"The FutureTPM project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 779391."

If you need further information, please contact the coordinator:

TECHNIKON Forschungs- und Planungsgesellschaft mbH
Burgplatz 3a, 9500 Villach, AUSTRIA
Tel: +43 4242 233 55 Fax: +43 4242 233 55 77
E-Mail: coordination@futuretpm.eu

Thank You.