Some Use Cases

for Formal Verification

Format

<Some Use Case>

Context

Description

Goal(s)

Specification

Categorisation

- Emerging Protocols
- Running Code for the Arm Architecture

Emerging Protocols

Context

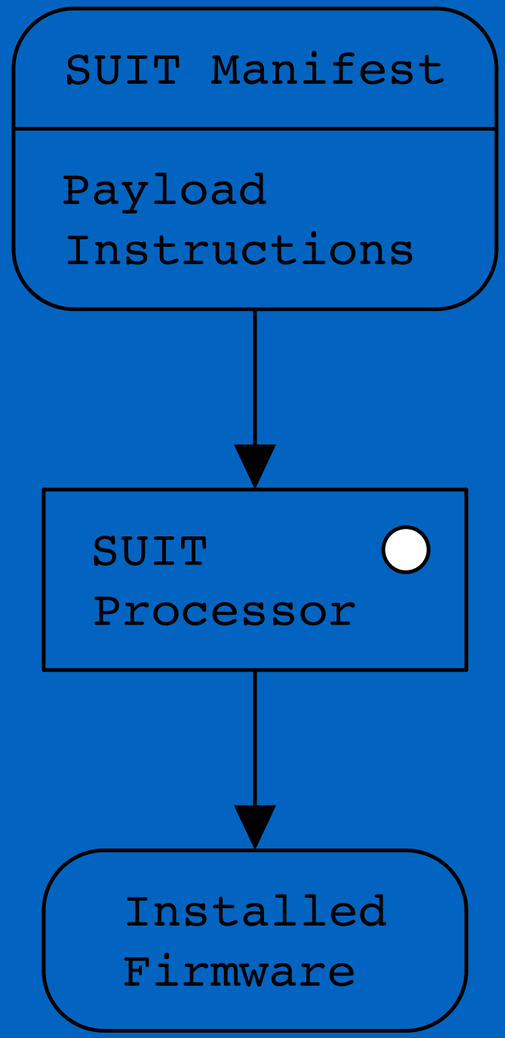
Software Update for the Internet of Things [SUIT-Architecture]

PSA Certified Firmware Update API

Description

SUIT "Manifests" contain *command sequences*, instructing the recipient on what to do with the to-be-installed payloads

Command sequences are fetched, decoded and executed by the SUIT Manifest *processor*


SUIT Manifest
Payload
Instructions
SUIT
Processor
Installed
Firmware

Goal(s)

The protocol meets the requirements against attackers described in the SUIT threat model

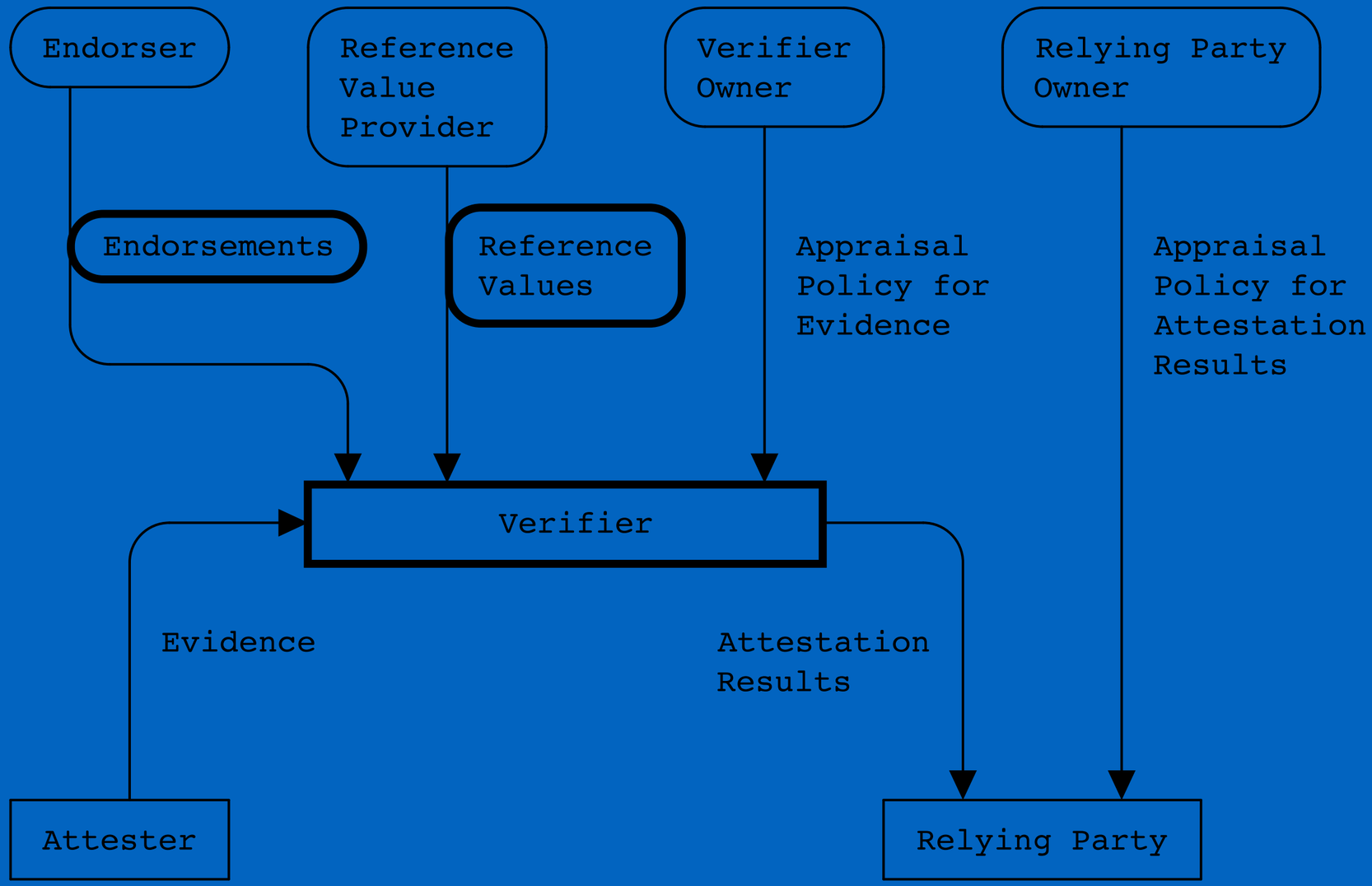
Specification

draft-ietf-suit-manifest

RFC 9124

Context

RATS Verifier


Endorser
Reference Value Provider
Verifier Owner
Relying Party Owner
Endorsements
Reference Values
Appraisal Policy for Evidence
Appraisal Policy for Attestation Results
Verifier
Evidence
Attestation Results
Attester
Relying Party

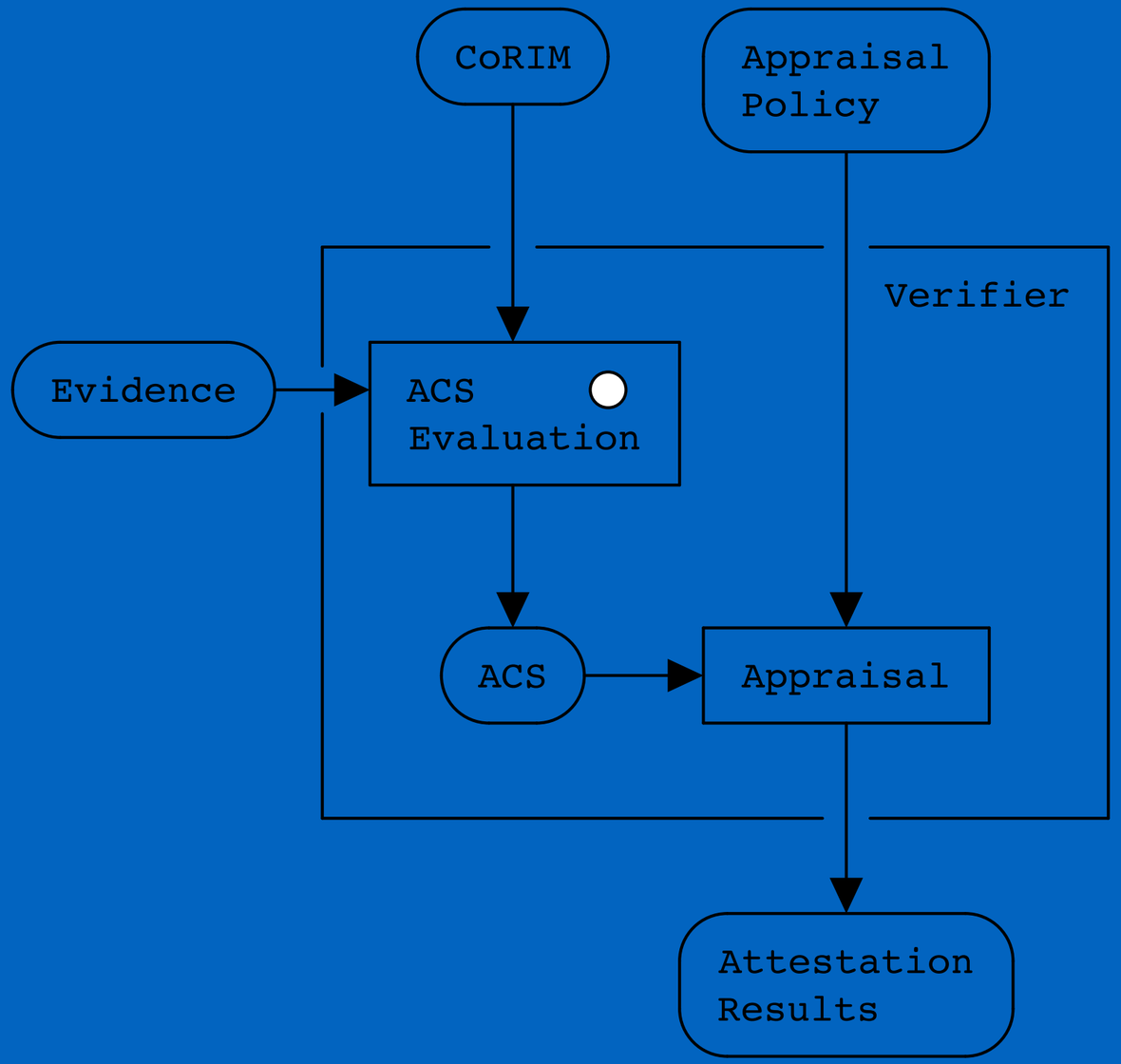
CoRIM Appraisal

Description

Given input Evidence and CoRIM description of reference and endorsed values, compute the Accepted Claims Set (ACS)

ACS = combination of reference state and actual state of the Attester

Attestation Results can be extracted as a View on the ACS determined by the Appraisal Policy


CoRIM
Appraisal Policy
Verifier
Evidence
ACS Evaluation
ACS
Appraisal
Attestation Results

Goal(s)

Deterministic computation of the ACS

CoRIM Appraisal

Specification

draft-ietf-rats-corim

Running Code for the Arm Architecture

Context

Real-time Linux (`PREEMPT_RT` + `SCHED_DEADLINE`) use cases - e.g., automotive, IIoT, medical devices

Characterising the behaviour of task synchronization and measure their impact on real-time tasks, in the *worst case*

"Automata-based Formal Analysis and Verification of the Real-Time Linux Kernel" (Daniel Bristot de Oliveira)

Context (cont.)

Model built as a set of formal specifications using automata theory

Key insights:

- A complex model can be built from simpler Lego blocks
- Overhead is acceptable (even in production)

Context (cont.)

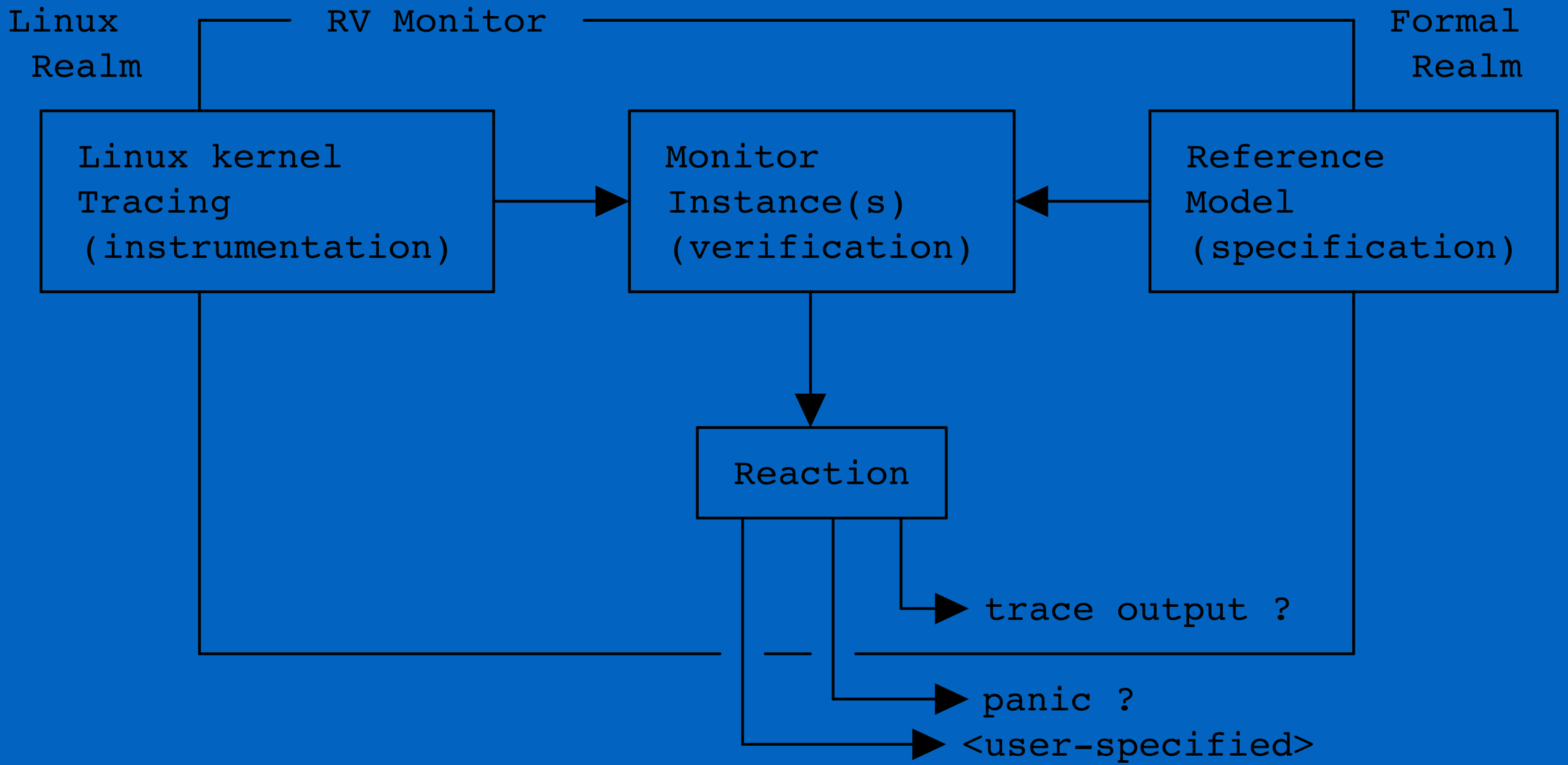
Validation of the model:

- Analysis of the properties of the automata
- Comparison of the model against system traces

Context (cont.)

Runtime Verification (RV) in the Linux Kernel

- Linux RV (>=6.0)


Linux
Realm
RV Monitor
Formal
Realm
Linux kernel
Tracing
(instrumentation)
Monitor
Instance(s)
(verification)
Reference
Model
(specification)
Reaction
trace output ?
panic ?
<user-specified>

Context

MCUboot is a secure bootloader for 32-bits microcontroller systems (IoT)

OS- and HW-independent

Targets: Zephyr, Mbed OS, RIOT, Apache's Mynewt and NuttX

Context (cont.)

Easy software upgrade

PSA certification (systems software) is in progress

Description

Currently, there is a (hand-written) run-time verification tool that checks the functional features. (Note: crypto is not covered)

But FV has not been attempted

Goal(s)

An aspect that would be interesting to cover from a FV perspective is image installation

E.g., check that behaviour is robust in face of power loss (i.e., the op completes successfully after power is restored)

Specification

Overall architecture

Support for encrypted images

Context

Android Virtualisation Framework (AVF)

Secure and private execution environments for executing application code

Security-oriented use cases that require stronger (even formally verified) isolation assurances

Android Virtualization Framework
pVM ⇄ TA

Description

pKVM, a trusted hypervisor that manages pVMs - "rich" TEE

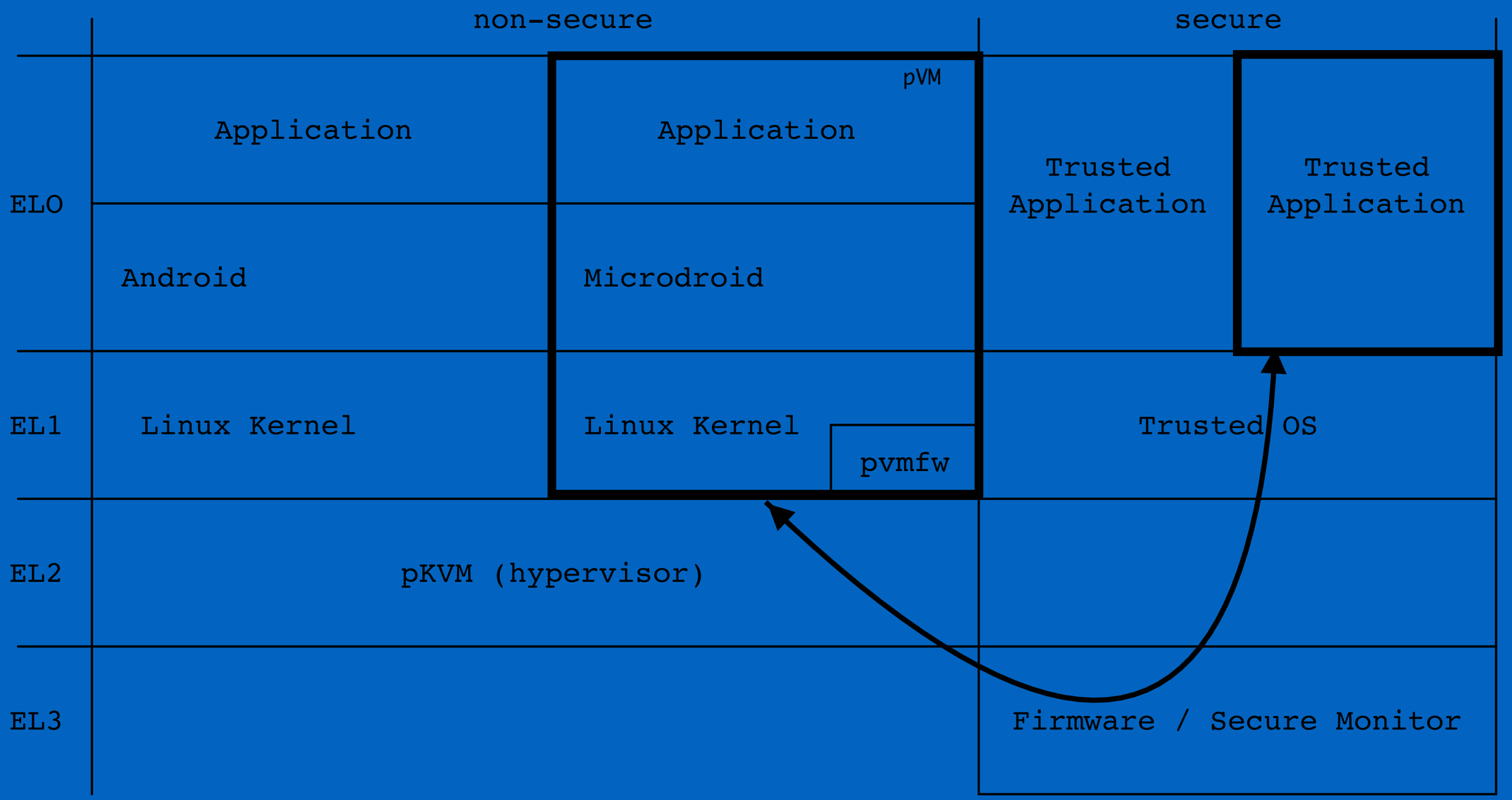
Trend: migrate Secure World TAs functionality to pVMs

However, *complete* TCB migration is not always feasible

Android Virtualization Framework
pVM ⇄ TA

Goal(s)

Ensure that the channel between pVM and TA can be trusted


non-secure
secure
pVM
Application
Application
Trusted Application
Trusted Application
EL0
Android
Microdroid
EL1
Linux Kernel
Linux Kernel
pvmfw
Trusted OS
EL2
pKVM (hypervisor)
EL3
Firmware / Secure Monitor

Specification

"The Android Platform Security Model (2023)"

"AVF Security"

FIN

Thank you David Brown, Ilias Apalodimas, Joakim Bech, Lorenzo Pieralisi & Vincent Guittot.