Distributed OS

Hermann Härtig

Authenticated Booting, Remote Attestation, Sealed Memory aka „Trusted Computing“

TECHNISCHE UNIVERSITÄT DRESDEN

Goals

Understand principles of:

- Authenticated booting
- The difference to (closed) secure booting
- Remote attestation
- Sealed memory

Non-Goal:

- Lots of TPM, TCG-Spec details
  → read the documents once needed

Some terms

- Secure Booting
- Authenticated Booting
- (Remote) Attestation
- Sealed Memory
- Late Launch / dynamic root of trust
- Trusted Computing (Group) / Trusted Computing Base

- **Attention:** terminology has changed

Trusted Computing (Base)

Trusted Computing Base (TCB)

- The set off all components, hardware, software, procedures, that must be relied upon to enforce a security policy.

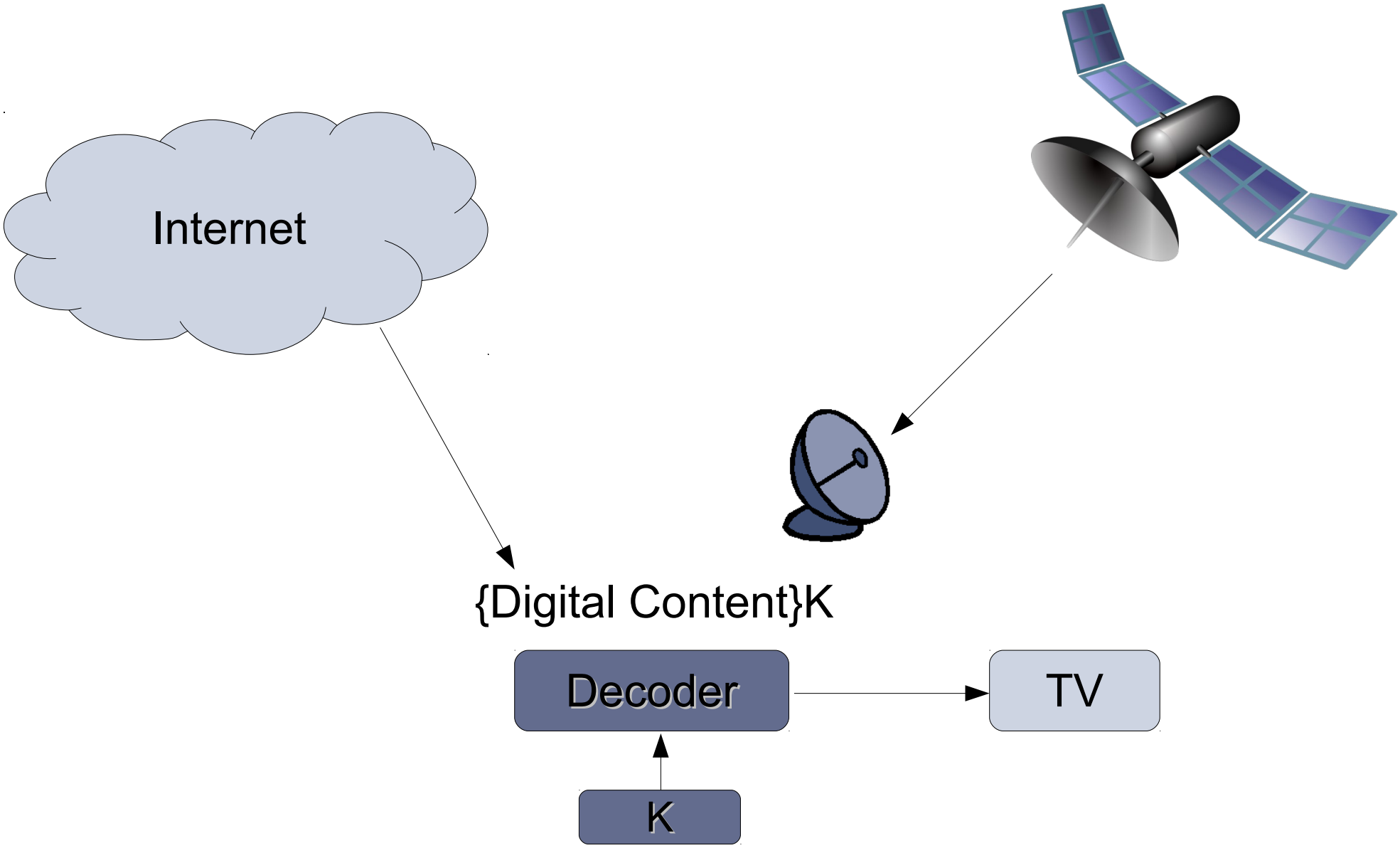
Trusted Computing (TC)

- A particular technology compromised of authenticated booting, remote attestation and sealed memory.

TC key problems

- Can running certain Software be prevented?
- Which computer system do I communicate with ?
- Which stack of Software is running?
  - In front of me?
  - On my server somewhere?
- Can I restrict access to certain secrets (keys) to certain software?

“Protect” Content



1) End User Example

Video Player:

- Provider sells content
- Provider creates key, encrypts content
- Client downloads encrypted content, stores on disk
- Provider sends key, but needs to ensure that only specific SW can use it
- Has to work also when client is off line
- PROVIDER DOES NOT TRUST CLIENT

2) Cloud Example

Virtual machine provided by cloud

- Client buys Cycles + Storage (Virtual machine)
- Client provides its own operating system
- Needs to ensure that provided OS runs
- Needs to ensure that provider cannot access data
- CLIENT DOES NOT TRUST PROVIDER

3) Industrial Plant Example

(Uranium Enrichment) Plant Control

- Remote Operator sends commands, keys
- Local operator occasionally has to run test SW, update to new version, ...
- Local technicians are not Trusted

4) Anonymizer example

Anonymity Service

- Intended to provide anonymous communication over internet
- Legal system can request introduction of trap door (program change)
- Service provider not trusted

Trusted Computing Terminology

Measuring

- “process of obtaining metrics of platform characteristics”
- Example for metric: Hash- Codes of SW

Attestation

- “vouching for accuracy of information”

Sealed Memory

- binding information to a configuration

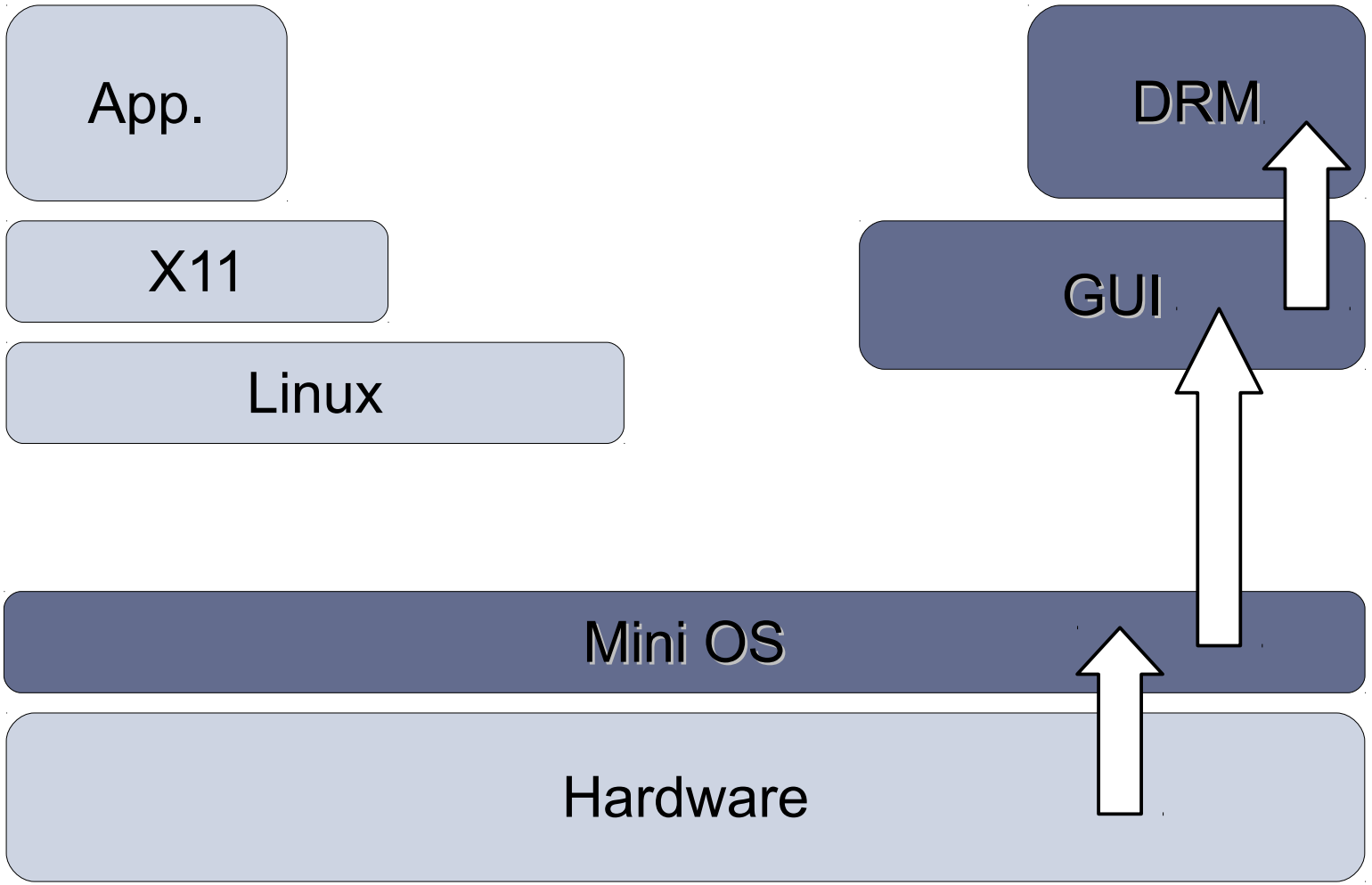
An example application: DRM

- „Digital Content“ is encrypted using symmetric key

- Smart-Card
  - contains key
  - authenticates device
  - delivers key only after successful authentication

- Assumptions
  - Smart Card can protect the key
  - „allowed“ OS can protect the key
  - OS cannot be exchanged

Secure Booting / Authenticated Booting



Notation

- **SK^{priv} SK^{pub}** Asymmetric key pair of some entity S

  $\{ M \}XK^{priv}$ Digital Signature for message M using the private key of signer X

  $\{ M \}YK^{pub}$ Message encrypted using public concellation key of Y

- **H(M)** Collision-Resistant Hash Function

- **Certificate** by authority Ca:

  $\{ ID, SK^{pub} , other\ properties \} CaK^{priv}$

Notation

Note:

- “{ M }Sk^{priv} Digital Signature”

  is short for: encrypt(H(M),Sk^{priv})

- “{ M }Sk^{pub} Message concealed ...”

  does not necessarily imply public key encryption

  for full M

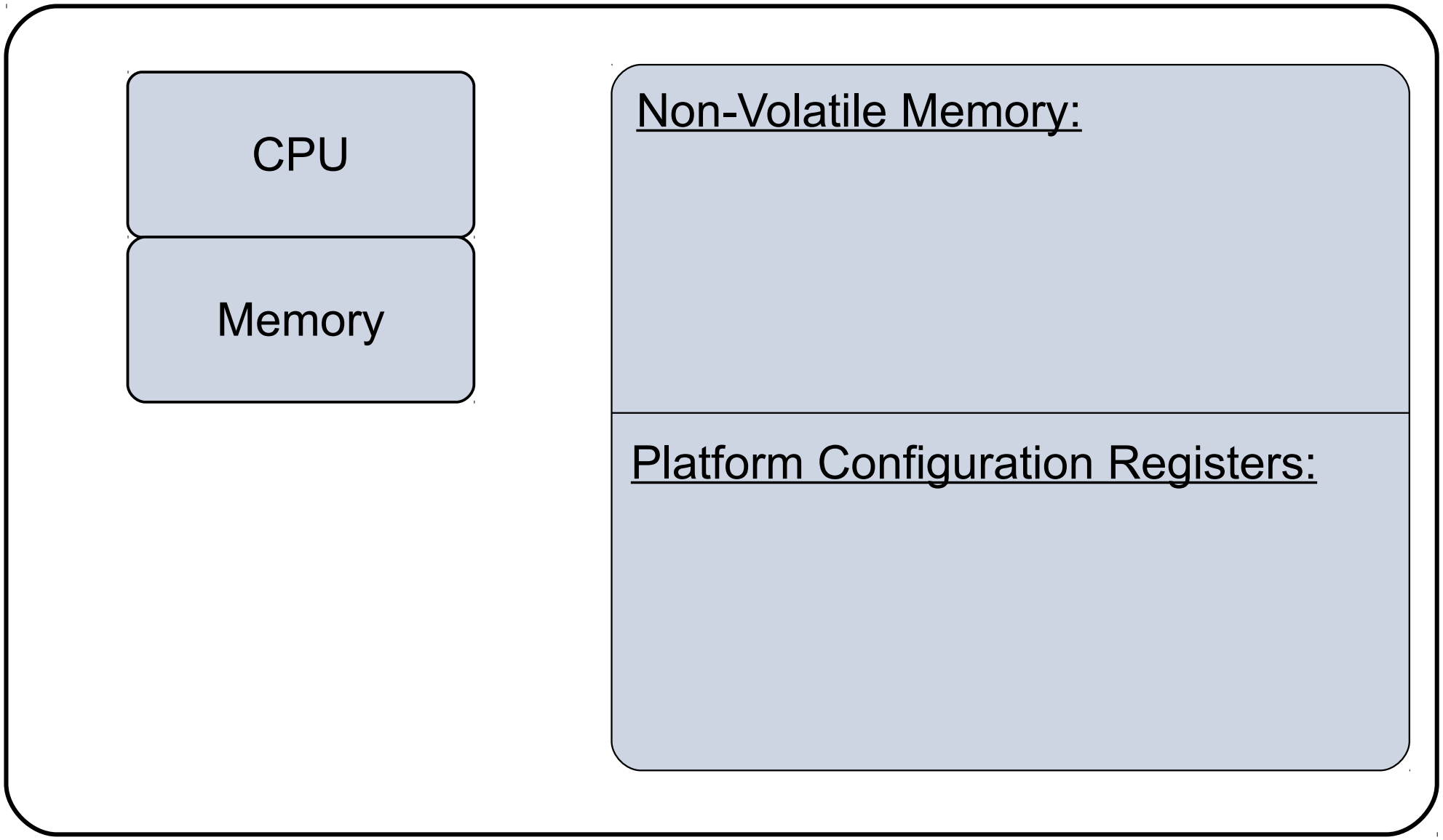
  (rather a combination of symmetric

  and asymmetric methods)

Identification of Software

- Program vendor: Foosoft FS
- Two ways to identify Software:
  - H(Program)
  - {Program, ID- Program}FSK^{priv}
    use FSK^{pub} to check the signature must be made available, e.g. shipped with the Program
- The „ID” of SW must be made available somehow.

Tamperresistant black box (TRB)



Ways to “burn in” the OS or secure booting

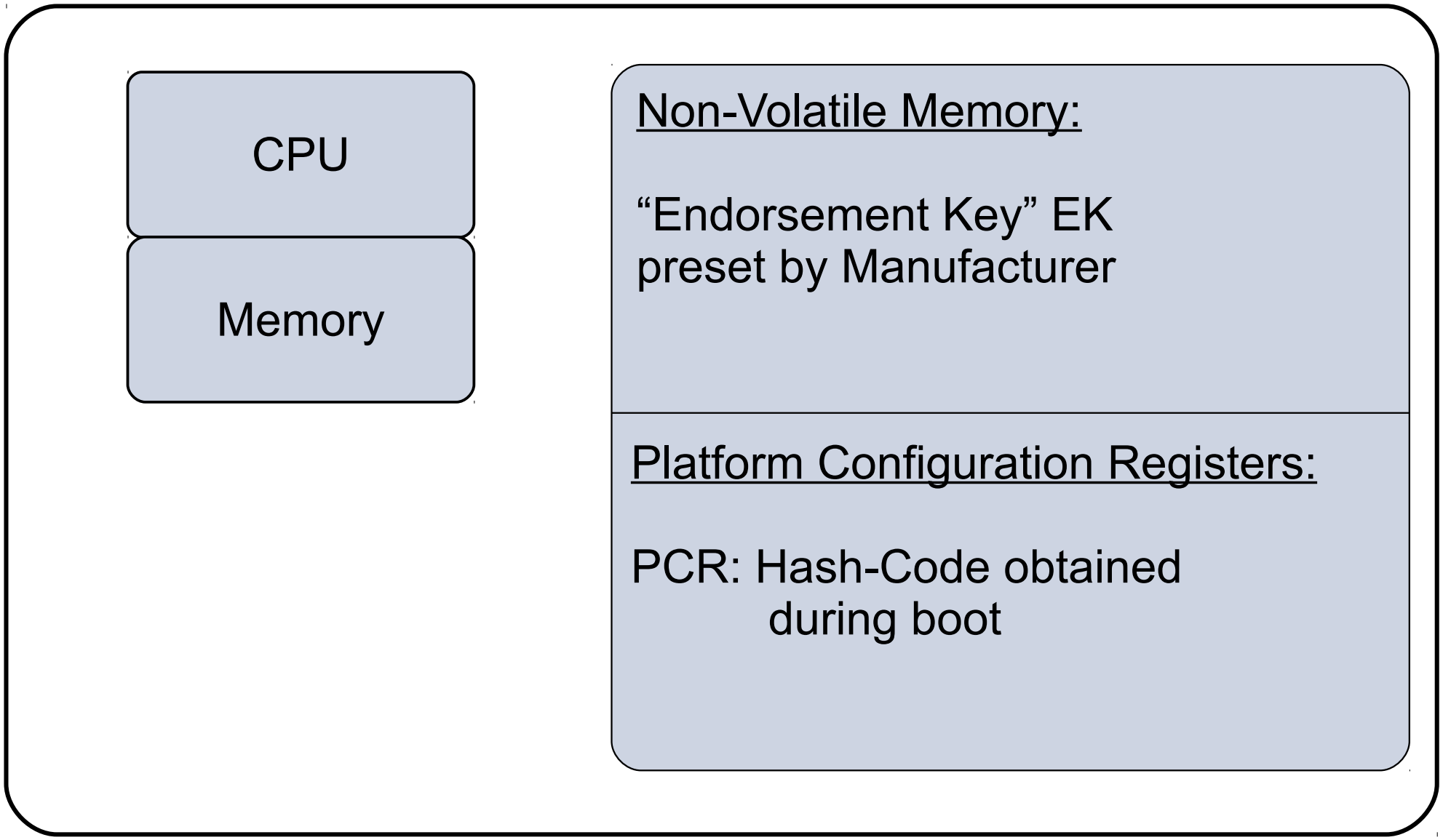
- Read-Only Memory
- Allowed H(OS) in NV memory preset by manufacturer
  - load OS- Code
  - compare H(loaded OS code) to preset H(OS)
  - abort if different
- Preset FSK^{pub} in NV memory preset by manufacturer
  - load OS- Code
  - check signature of loaded OS-Code using FSK^{pub}
  - abort if check fails

Authenticated Booting (AB)

Phases:

- Preparation by Manufacturers (TRB and OS)
- Booting & “Measuring”
- Remote attestation

Authenticated Booting (AB)



Vendors of TRB and OS

- TRB_generates key pair: „Endorsement Key“ (EK)
  - stores in TRB NV Memory: EK^{priv}
  - emits: EK^{pub}

- TRB vendor certifies: {“a valid EK”, EK^{pub}}TVK^{priv}
- OS-Vendor certifies: {„a valid OS“, H(OS)}$OSVK^{priv}$
- serve as identifiers: EK^{pub} and H(OS)

Booting & Attestation

Booting:

- TRB “measures” OS- Code (computes H(OS-Code))
- stores in PCR
- no other way to write PCR

Attestation:

- Challenge: nonce
- TRB generates Response: $\{PCR, nonce'\}EK^{priv}$

Remaining problems

- Now we know identities: H(loaded-OS) and EK^{pub}
- SIMPLE VERSION NOT PRACTICAL !!!

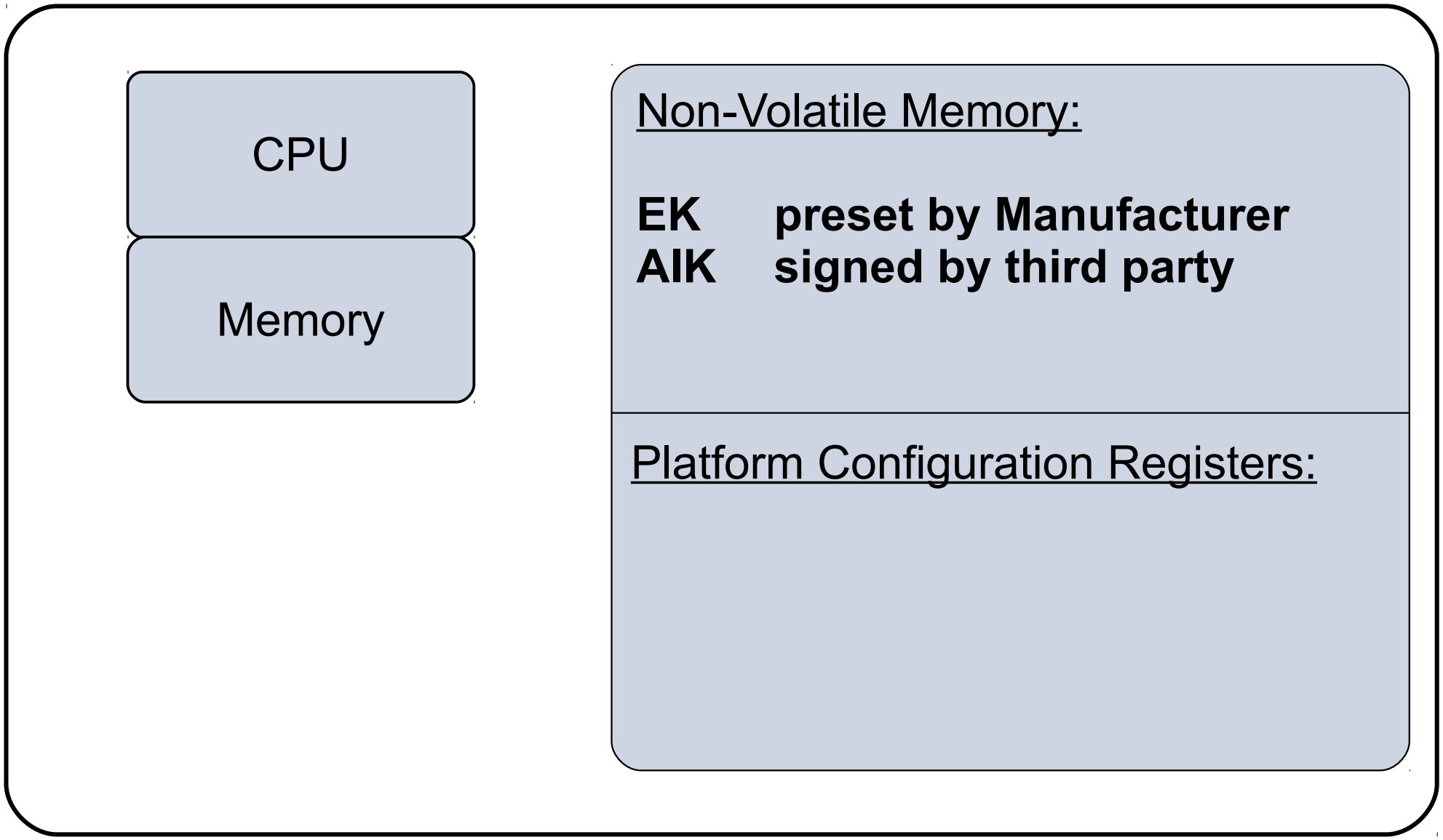
  Problems to solve:

  - Privacy: remote attestation requires (reveals) idendity (EKpub)
  - OS versioning
  - Attestates: Which system has been booted, but
    WHAT ABOUT REBOOT ?
    Remote attestation with EK^{pub} on each message ???
  - not only “OS” on platform: SW stacks or trees
  - Black box to big: TRB → TPM/ATM-TrustZone
  - Sealed memory

Remote Attestation and Privacy (use AIK)

- Remote attestation reveals platform identity: EK^{pub}
- add intermediate step:
  - Attestation Identity Key (AIK)
  - Trusted Third Party as anonymizer (TTP)

Remote Attestation and Privacy (use AIK)



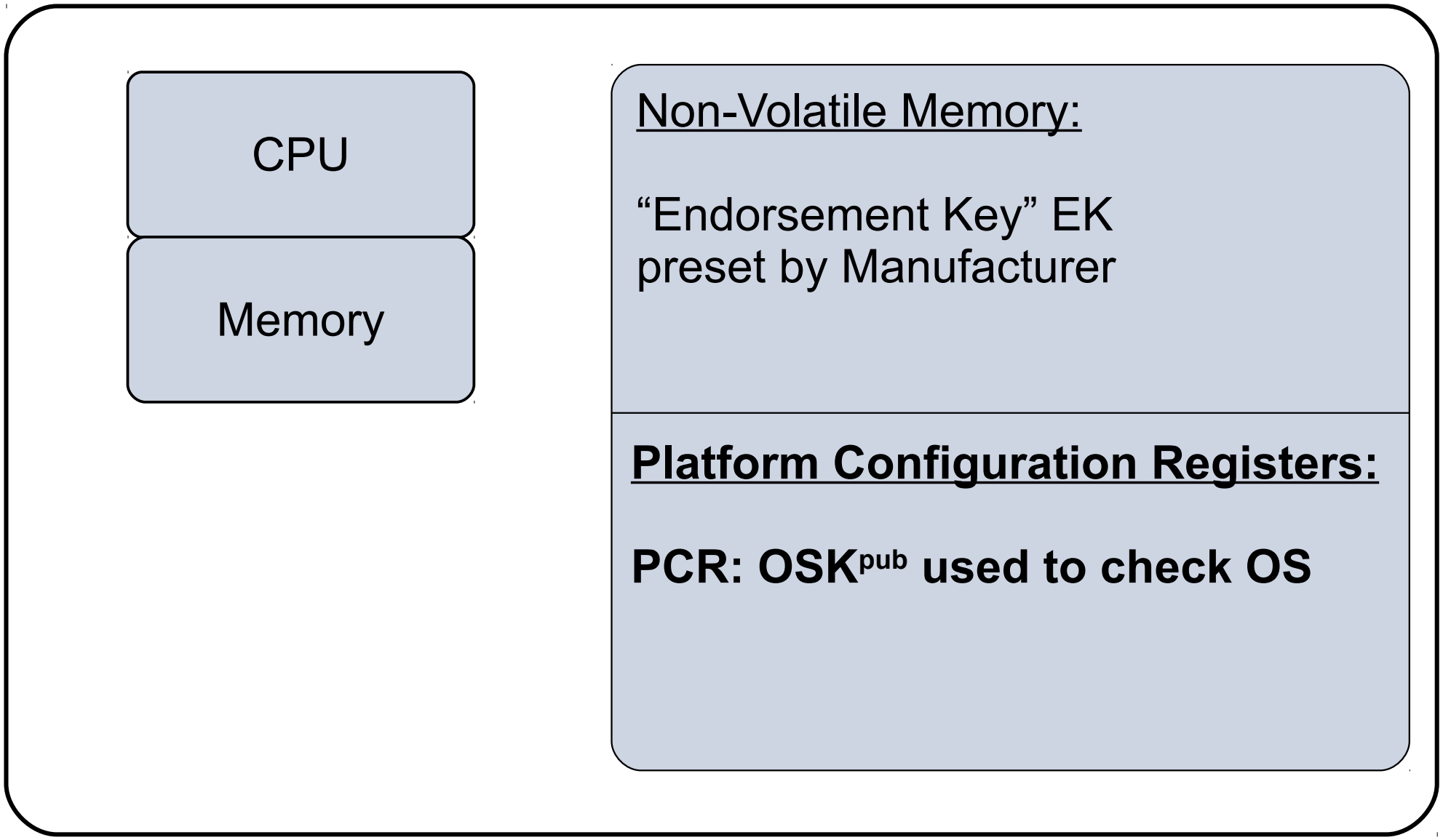
Remote Attestation and Privacy (use AIK)

- Generate AIK in TRB

- send { AIK } EK^{priv} to trusted third party
- third party certifies: {AIK, “good ID” } $TTPK^{priv}$

- AIK used instead of EK during remote attestation, response:
  - {AIK, “good ID” } $TTPK^{priv}$
  - { Nonce, H(OS)}AIK^{priv}

AB to allow OS versions



Vendors of TRB and OS

- TRB generates key pair:
  - stores in TRB NV Memory: EK^{priv}
  - emits: EK^{pub}
- TRB vendor certifies: {“a valid EK”, EK^{pub}}TVK^{priv}
- OS-Vendor certifies: {„a valid OS“, OSK^{pub}}$OSVK^{priv}$
- and signs OS-Code: {OS-Code}OSK^{priv}
- serve as identifiers: EK^{pub} and OSK^{pub}

Booting & Attestation (with versions)

Booting:

- TRB checks OS- Code using some **OSK^{pub}**
- stores **OSK^{pub}** in PCR
- no other way to write PCR

Attestation:

- Challenge: nonce
- TRB generates Response: $\{PCR, nonce'\}EK^{priv}$

AB considering reboot

- attestation required at each request:
  - {PCR, nonce' }EK^{priv}
  - PCR: H(OS) bzw. OSK^{pub}

- always requires access to and usage of EK
- race condition!

Instead:

- create new keypair on every reboot:
  - $OSrunningAuthK^{priv}$ $OSrunningAuthK^{pub}$

Booting (AB considering reboot)

Booting:

- TRB checks OS- Code using some OSK^{pub}
- stores OSK^{pub} in PCR
- creates OSrunningAuthK keypair
- certifies: $\{ OSrunningAuthK^{pub}, OSK^{pub}\}EK^{priv}$

Attestation (AB considering reboot)

Attestation:

- Challenge: nonce
- OS generates response:
  - $\{ OSrunningAuthK^{pub}, OSK^{pub}\}EK^{priv}$
  - $\{nonce'\} OSrunningAuthK^{priv}$

Establish Secure Channel to OSRunning

Booting:

- TRB checks OS- Code using some OSK^{pub}
- stores OSK^{pub} in PCR
- creates OSrunningAuthK keypair
- creates OSrunningConsK keypair
- certifies: { $OSrunningAuthK^{pub}$, OSrunningConsKpub, OSK^{pub}}EK^{priv}

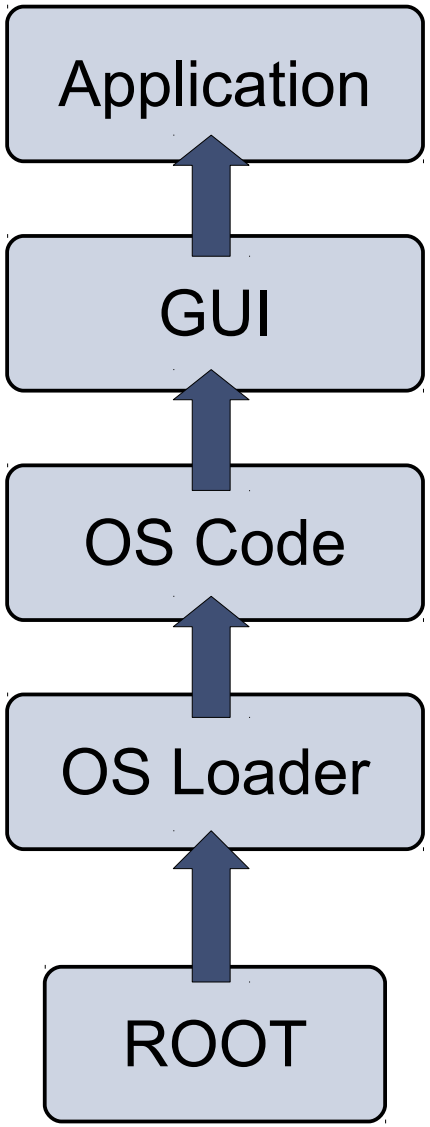
Secure Channel:

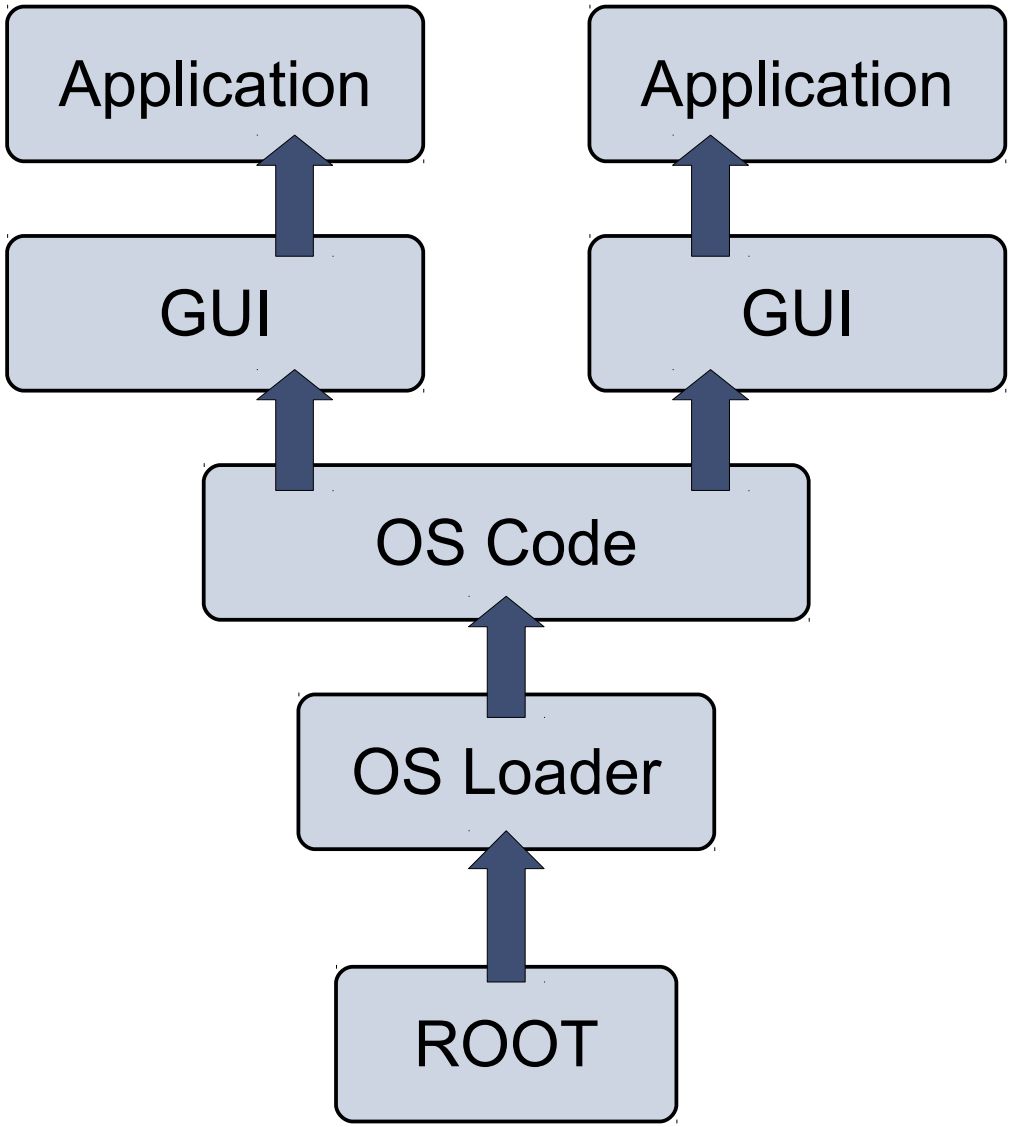
- { message } $OSrunningConsK^{pub}$

Assumptions

- TRB can protect: EK, PCR
- OS can protect: OSrunningK^{priv}
- Rebooting destroys content of
  - PCR and Memory Holding OSrunningK^{priv}

Software stacks and trees





Software stacks and trees

- “Extend” Operation
  - stack: $PCR_n = H(PCR_{n-1} \;||\; \text{next-component})$
  - tree: difficult (unpublished ?)

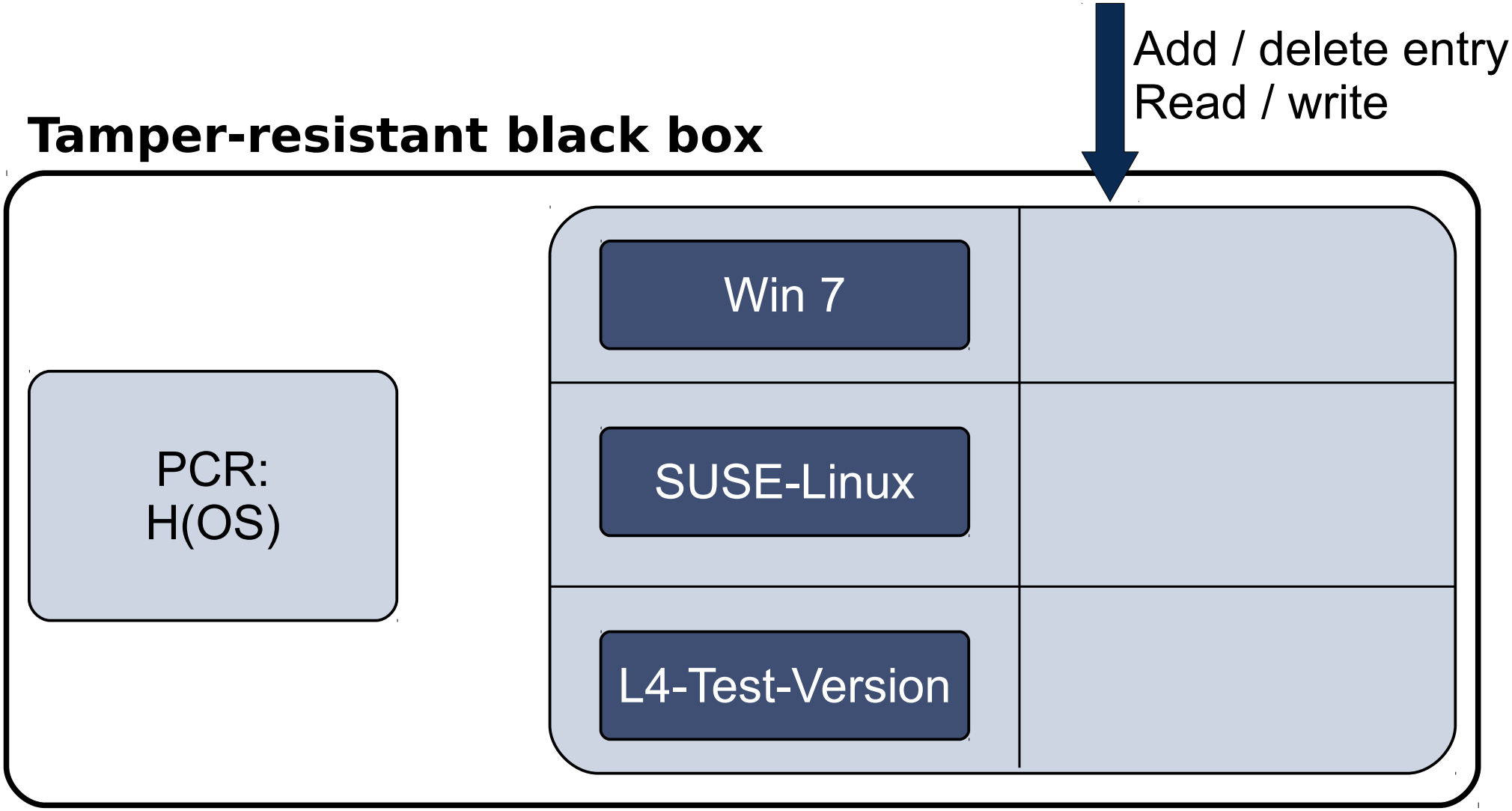
- Key pairs per step:
  - OS controls applications → generate key pair per application
  - OS certifies
    - { Application 1, $App1K^{pub}$ } $OsrunningK^{priv}$
    - { Application 2, $App2K^{pub}$ } $OSrunningK^{priv}$

Late Launch

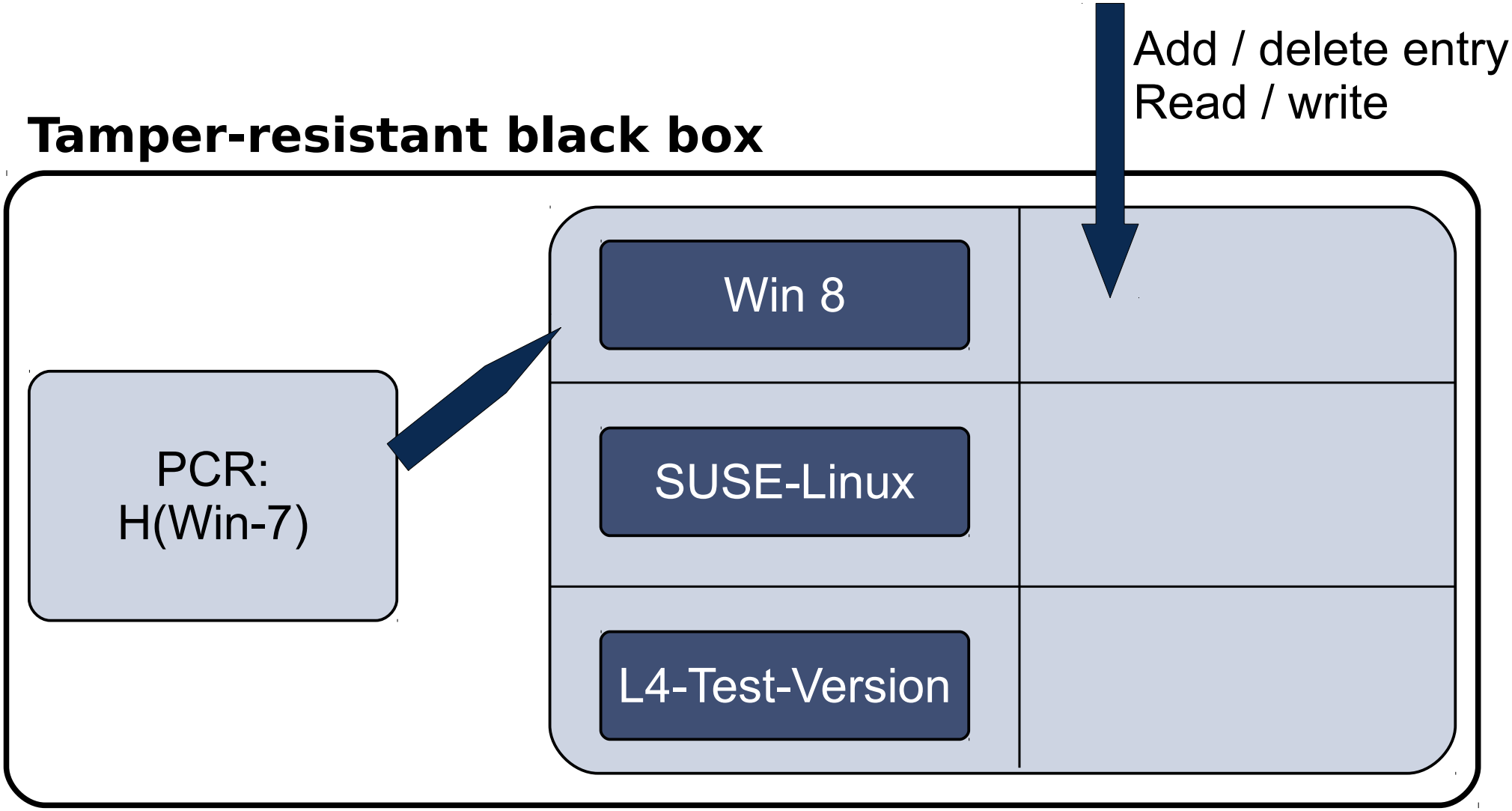
- Use arbitrary SW to start system and load all SW
- provide specific instruction to enter “secure mode”
  - set HW in specific state (stop all processors, IO, …)
  - Measure “root of trust” SW
  - store measurement in PCR

- AMD: “skinit” (Hash) arbitrary root of trust
- Intel: “senter” (must be signed by chip set manufacturer)

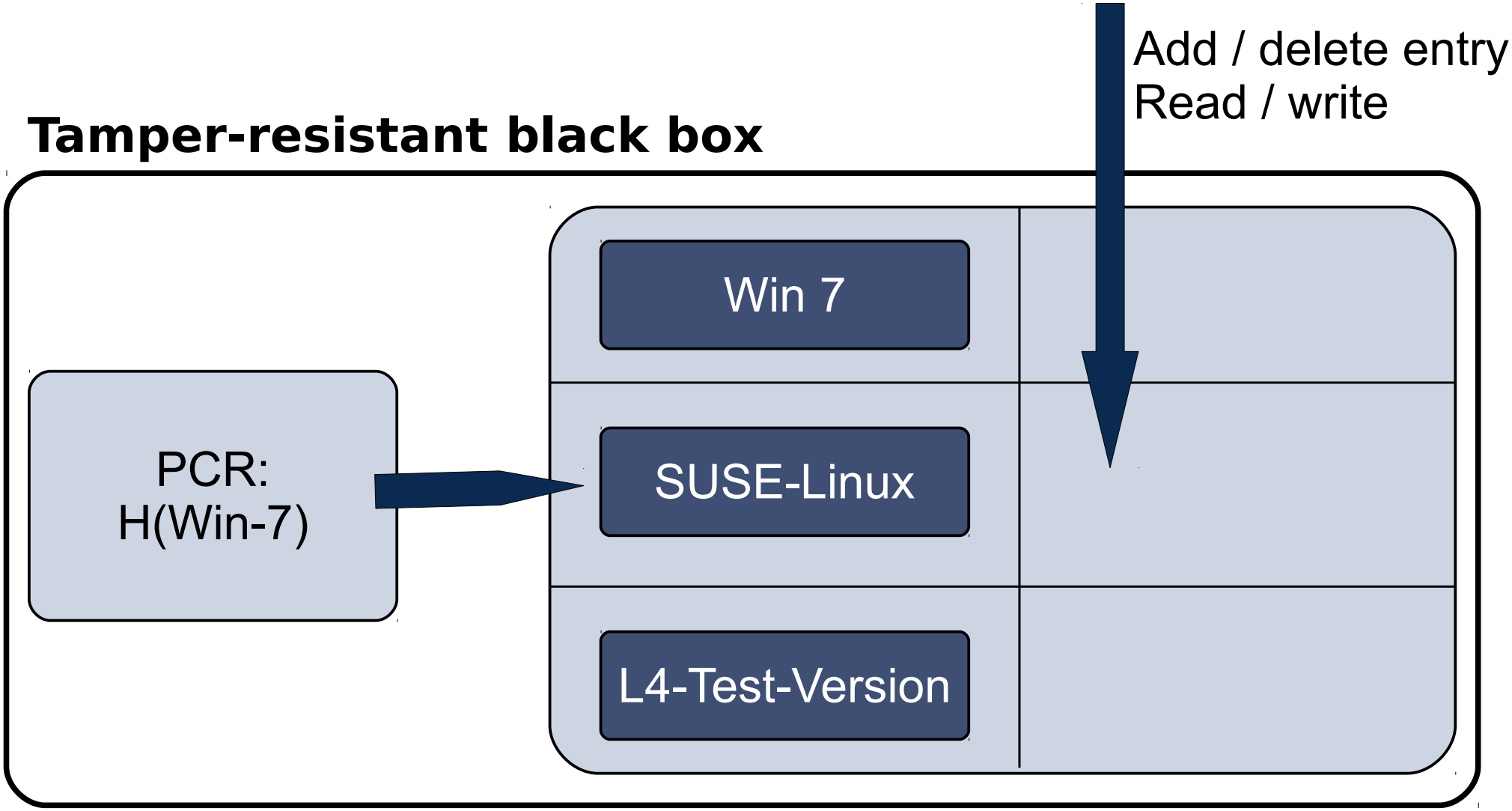
Sealed Memory



Sealed Memory

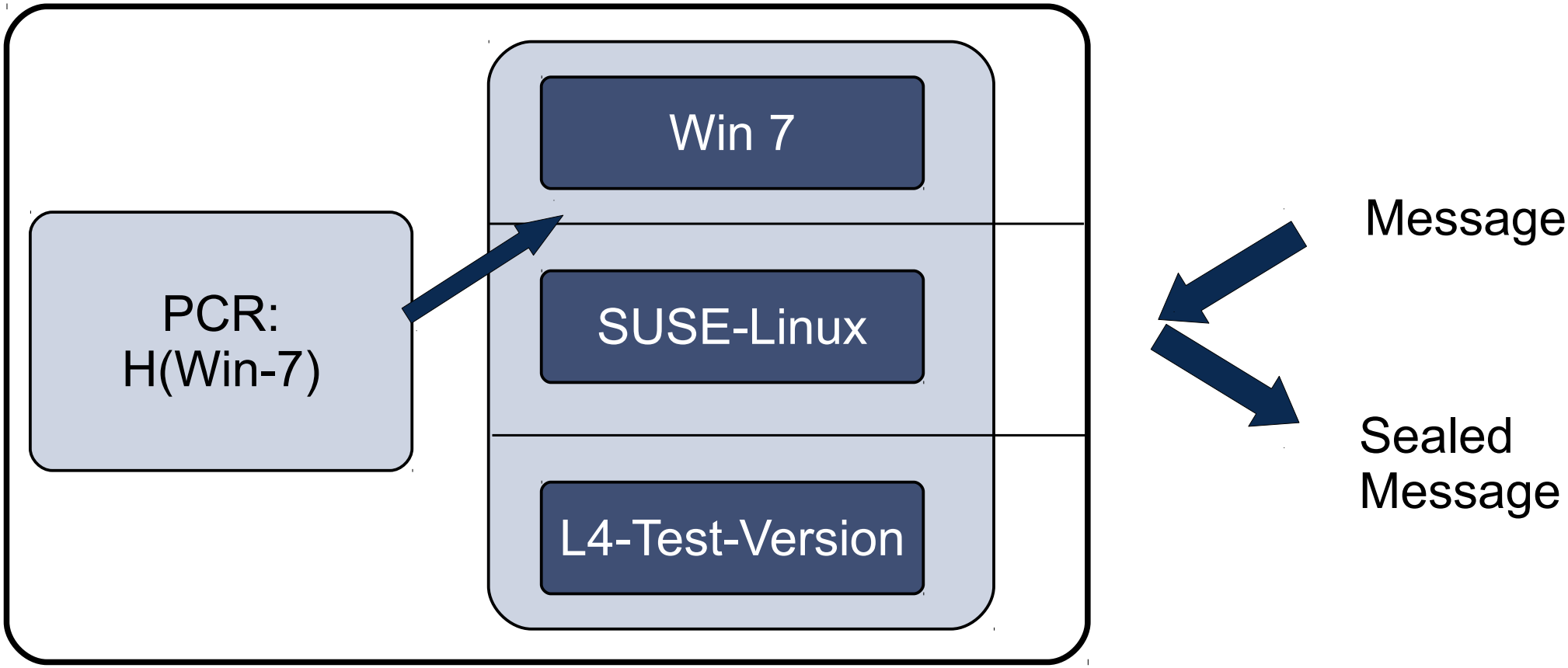


Sealed Memory



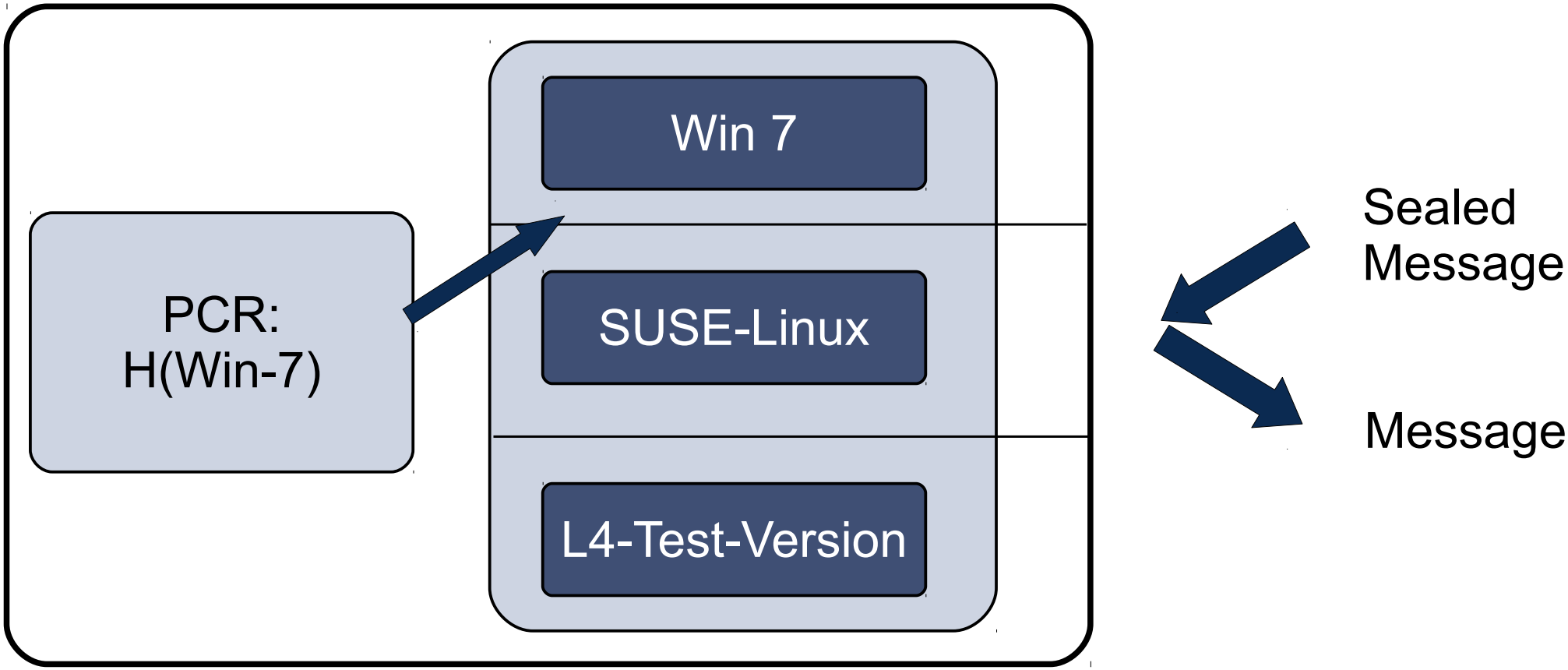
Sealed Memory: Seal Operation

Tamper-resistant black box

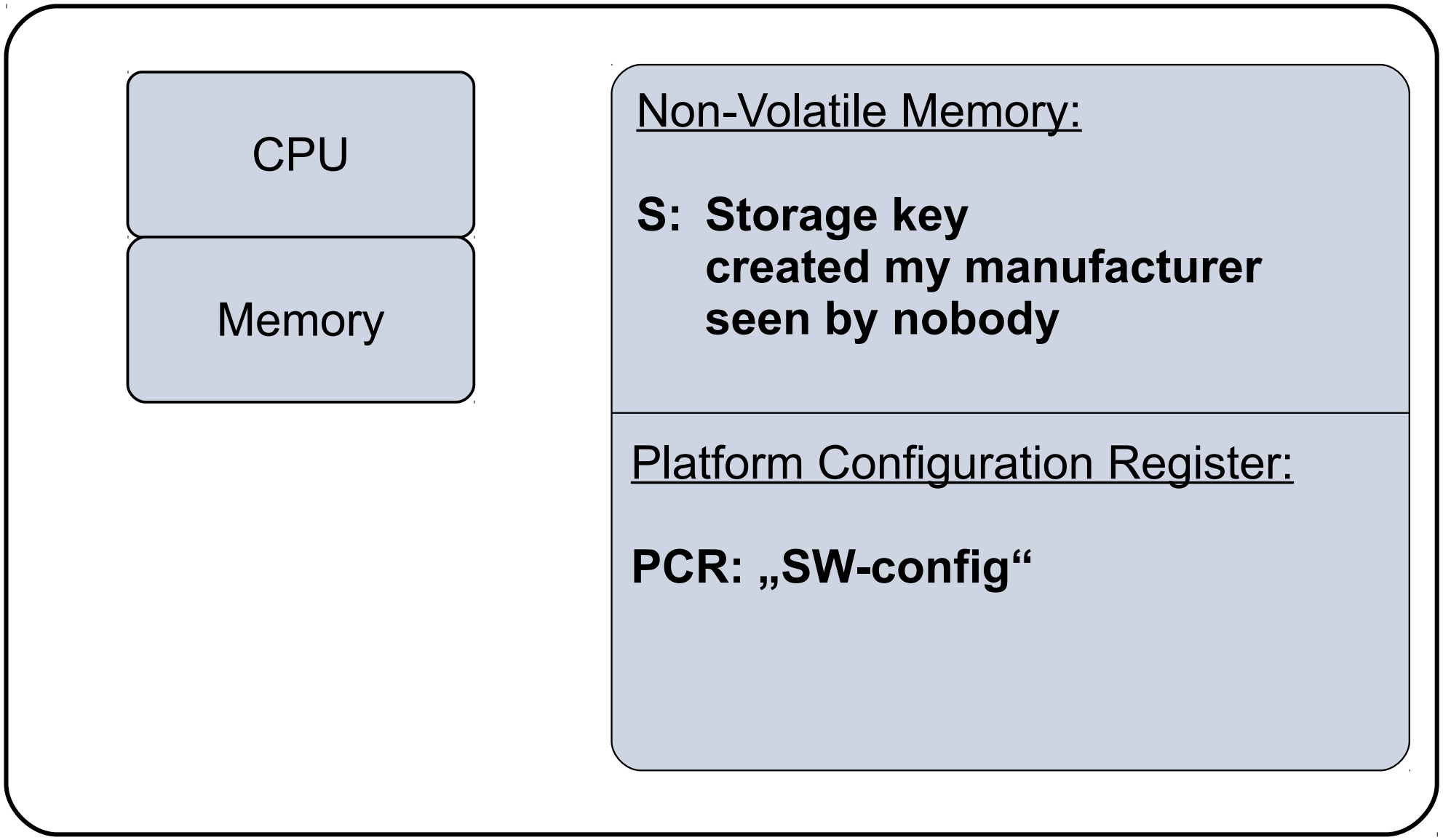


Sealed Memory: Unseal Operation

Tamper-resistant black box



Tamperresistant black box (TRB)



Sealed Memory

- Seal(message):

  encrypt(“PCR, message”, Storage-Key)

  → “sealed message”;

  emit sealed message

- Unseal(sealed_message):

  decrypt(“sealed_message”, Storage-Key)

  → “SW config, message”;

  If SW config == PCR then emit message else abort fi

Sealed Memory for future configuration

- Seal(message, **FUTURE_Config**):

  encrypt(“**FUTURE_Config**, message”, Storage-Key)

  → “sealed message”;
  emit sealed_message

- “seals” information such that it can be unsealed by a future configuration (for example: future version)

Example

- Win8: Seal („SonyOS, Sony-Secret“)

  → SealedMessage (store it on disk)

- L4: Unseal (SealedMessage)

  → SonyOS, Sony-Secret → PCR#SonyOS → abort

- SonyOS: Unseal(SealedMessage

  → SonyOS, Sony-Secret → PCR==SonyOS → ok

Migration ?

- How to transfer information form one TRB to another

  for example: key for decryption of videos

- Send information to third party

  Destroy information locally and prove to third party

  Third party provides information to another entity

Migration ?

- How to transfer information form one TRB to another

  for example: key for decryption of videos

- Send information to third party

  Destroy information locally and prove to third party

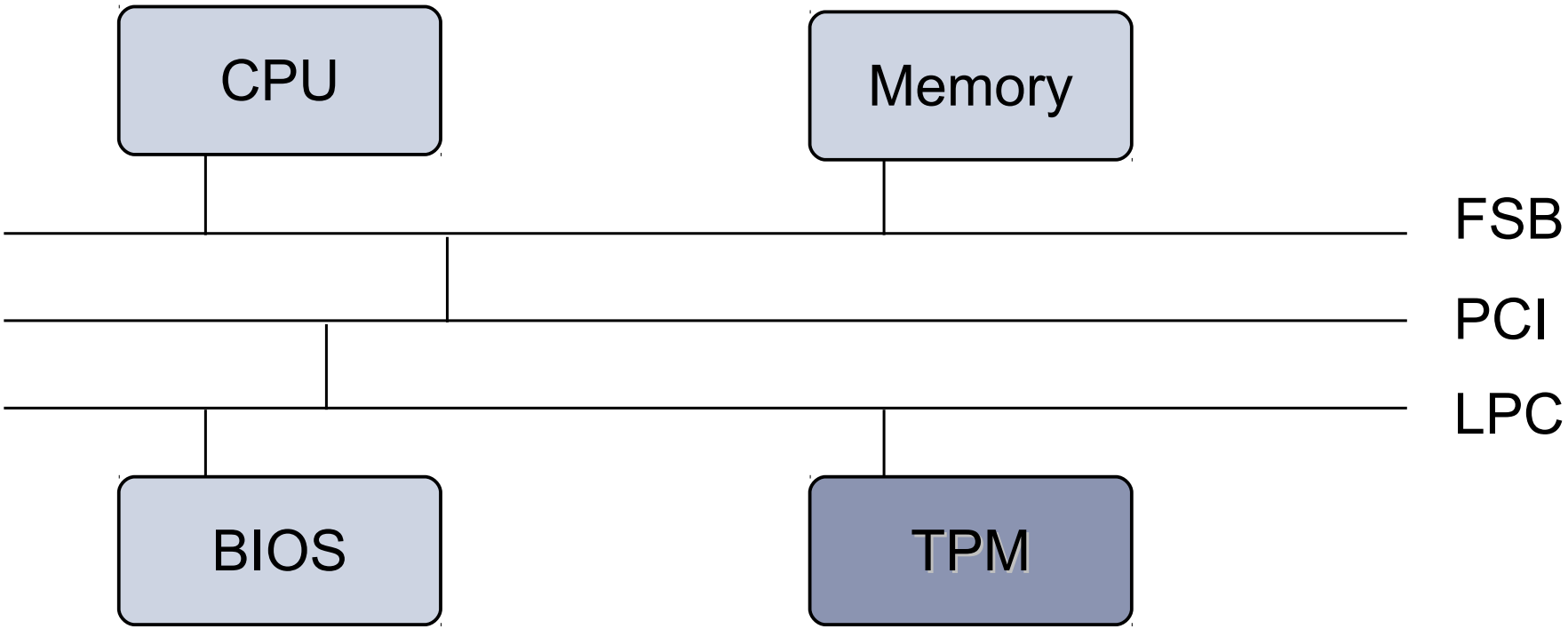
  Third party provides information to another entity

Tamper Resistant Box ?

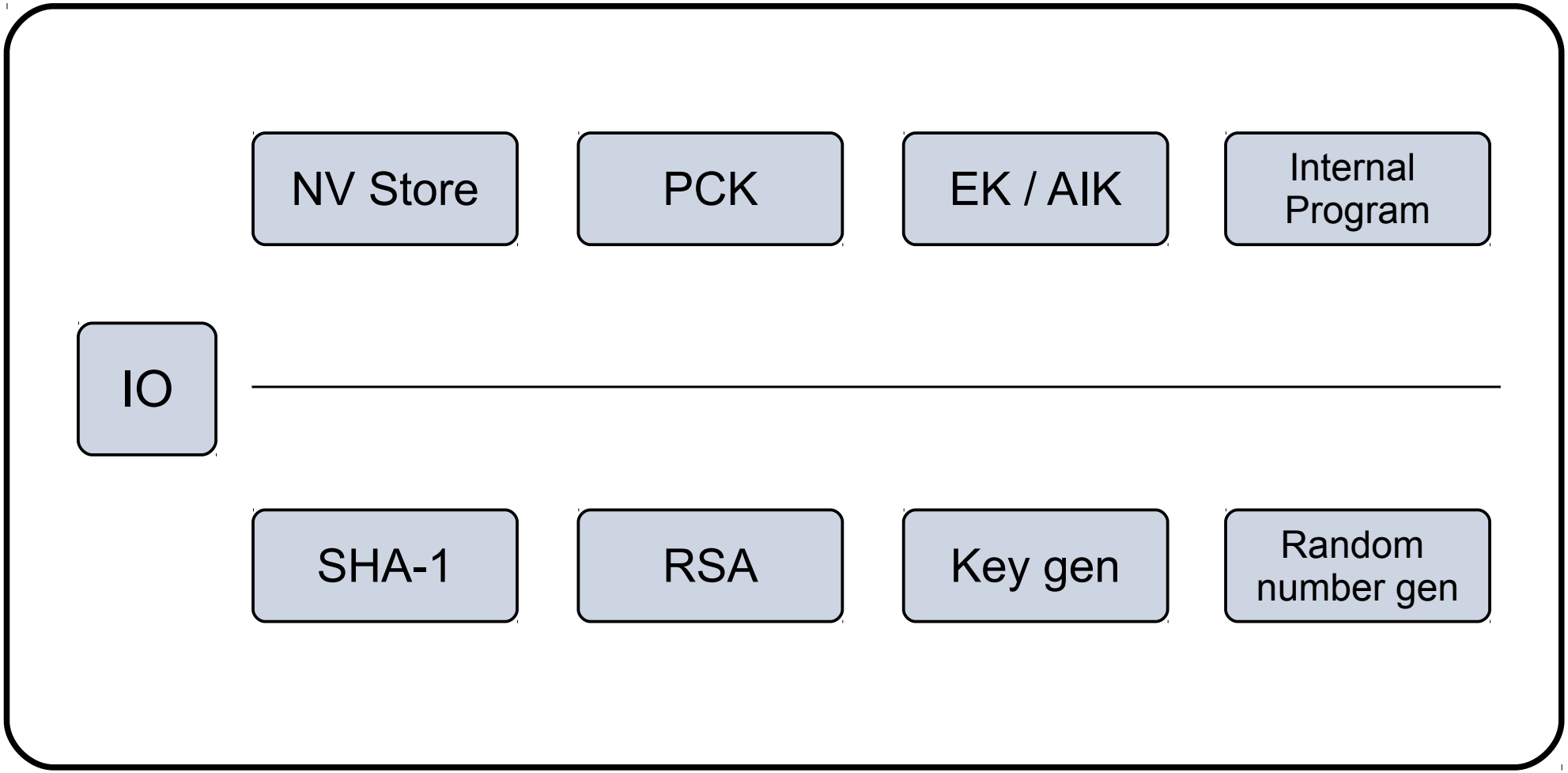
- Ideally, includes CPU, Memory, ...

- In practice
  - very rarely, for example IBM 4758 ...
  - Two HW versions
    - TPM:
      separate “Trusted Platform Modules” (replacing BIOS breaks TRB)
    - ARM TrustZone:
      Add a new privilege mode

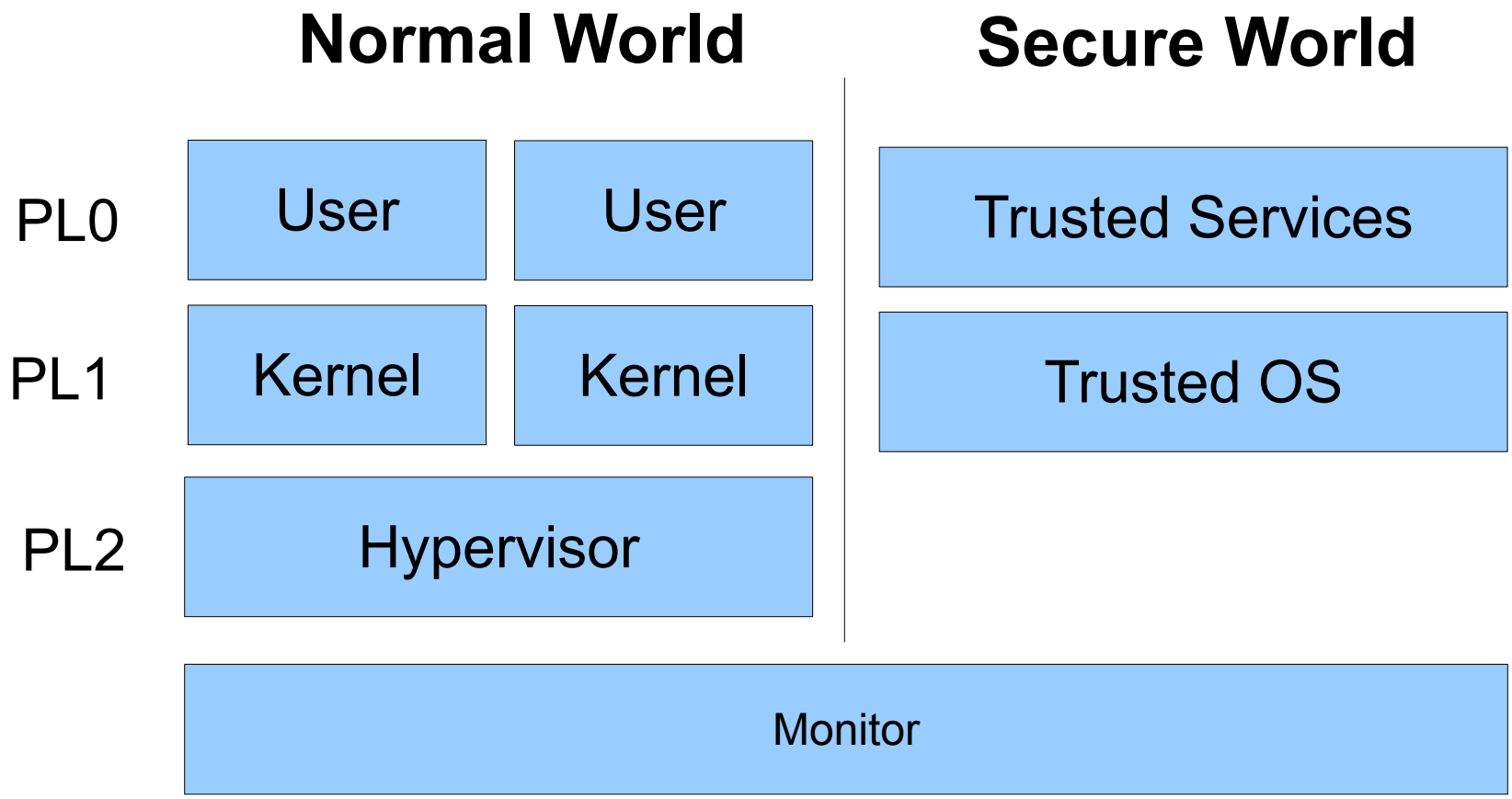
TPM: TCG PC Platforms



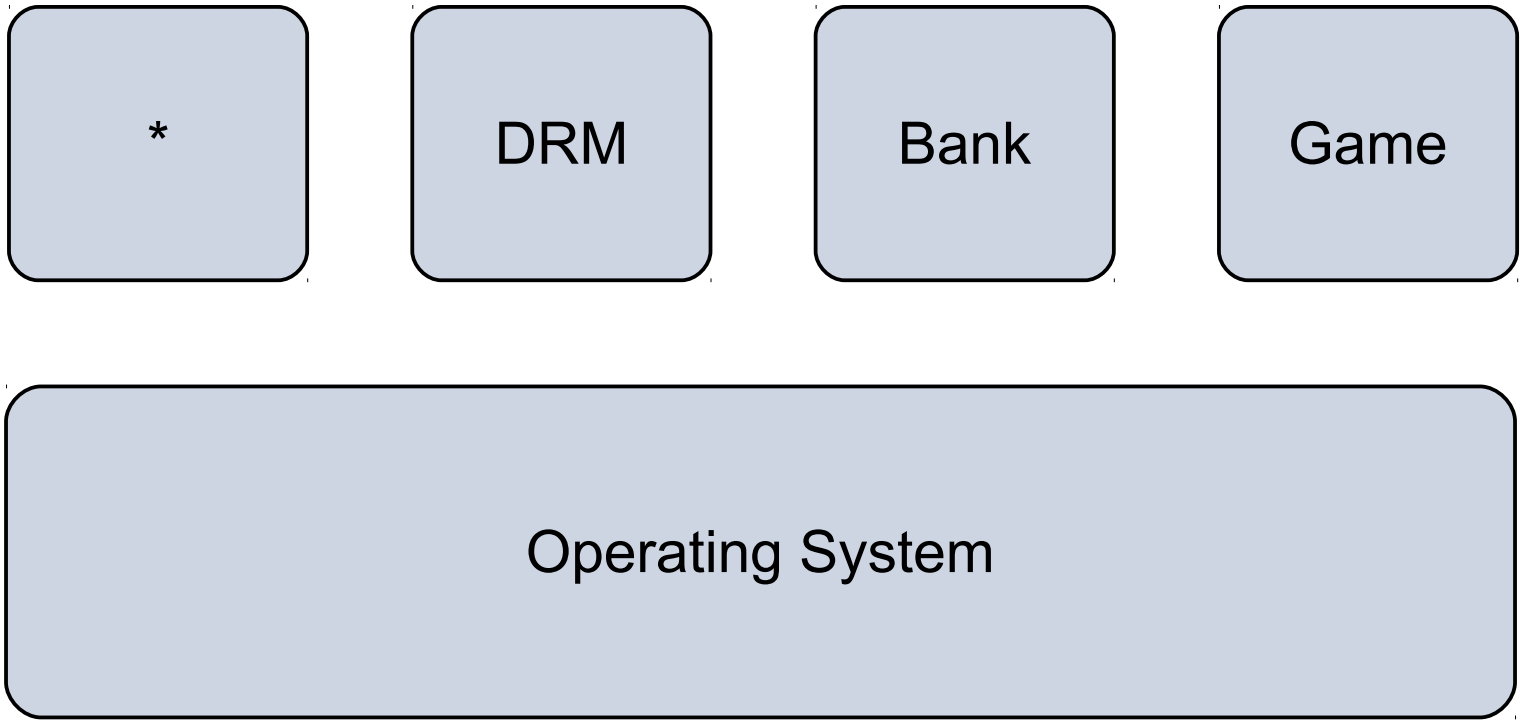
TPM


NV Store
PCK
EK / AIK
Internal Program
IO
SHA-1
RSA
Key gen
Random number gen

ARM TrustZone



Usage Scenarios and Technical Risks



Technical Risks

Hardware:

- Authenticity, Integrity, Tamper-Resistance
- Protection of CPU-priv
  Integrity of Rkey-OS-pub

Operating System

- Protection of keys (OSRunning, ...), Content, ...
- Isolation Applications
- Assurance

Side Channels !

References

- Specifications:

  https://www.trustedcomputinggroup.org/groups/TCG_1_3_Architecture_Overview.pdf

- Important Foundational Paper:

  Authentication in distributed systems: theory and practice

  Butler Lampson, Martin Abadi, Michael Burrows, Edward Wobber

  ACM Transactions on Computer Systems (TOCS)