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 / 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 securit 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 programs?

Trusted Computing Terminology

Measuring

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

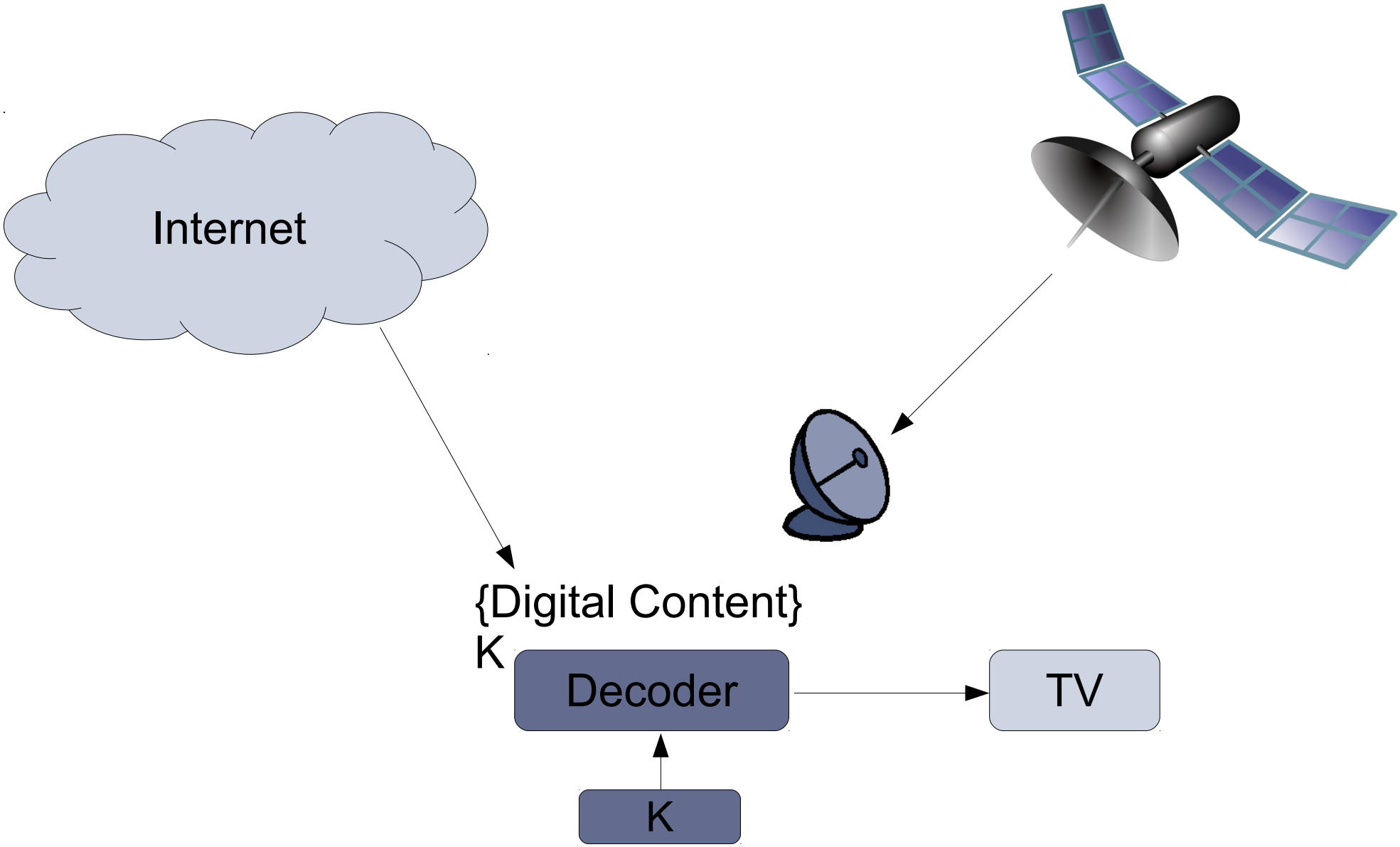
Attestation

- “vouching for accuracy of information”

Sealed Memory

- binding information to a configuration

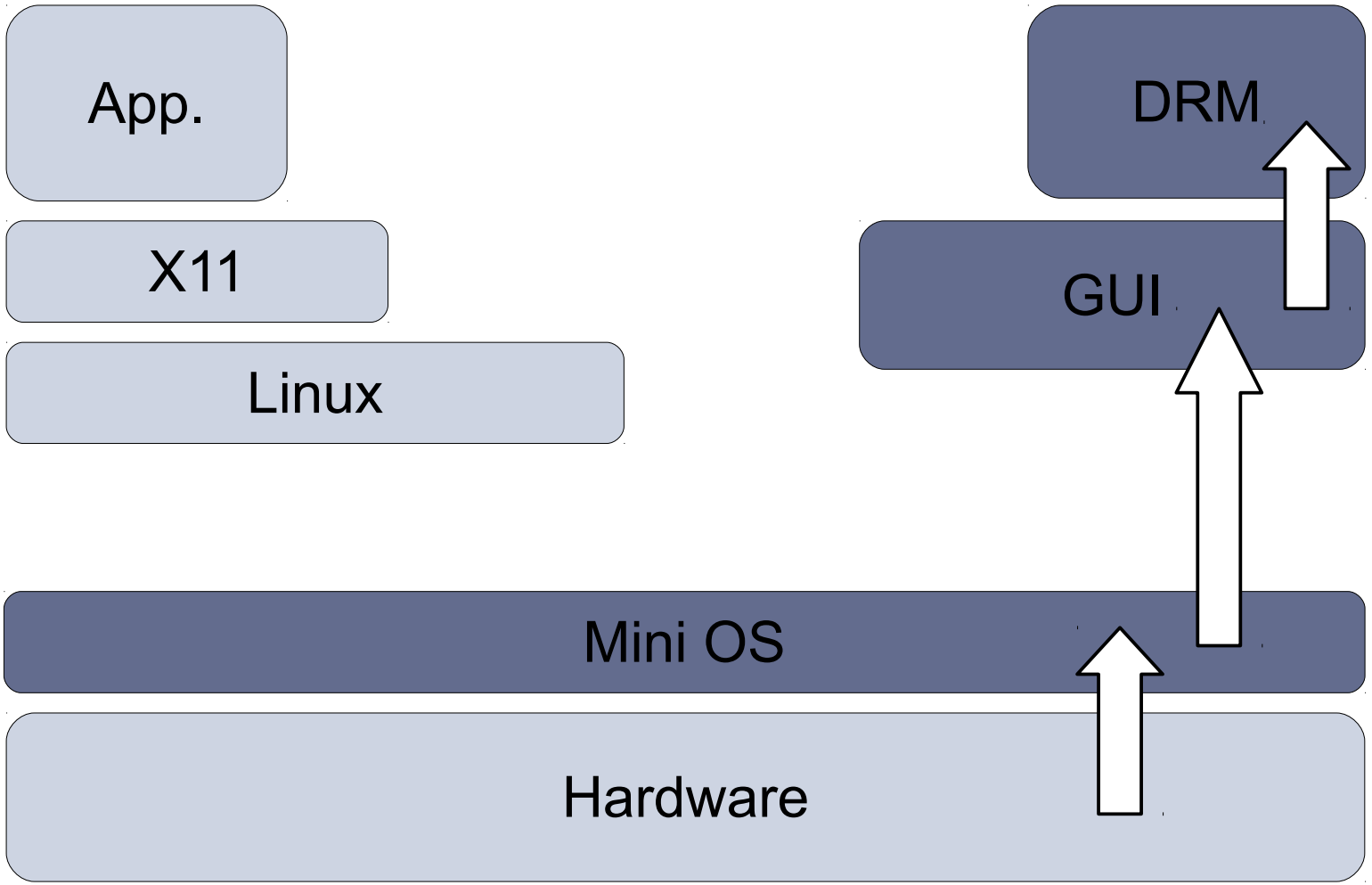
DRM: Trust ./. No Trust in end user



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})

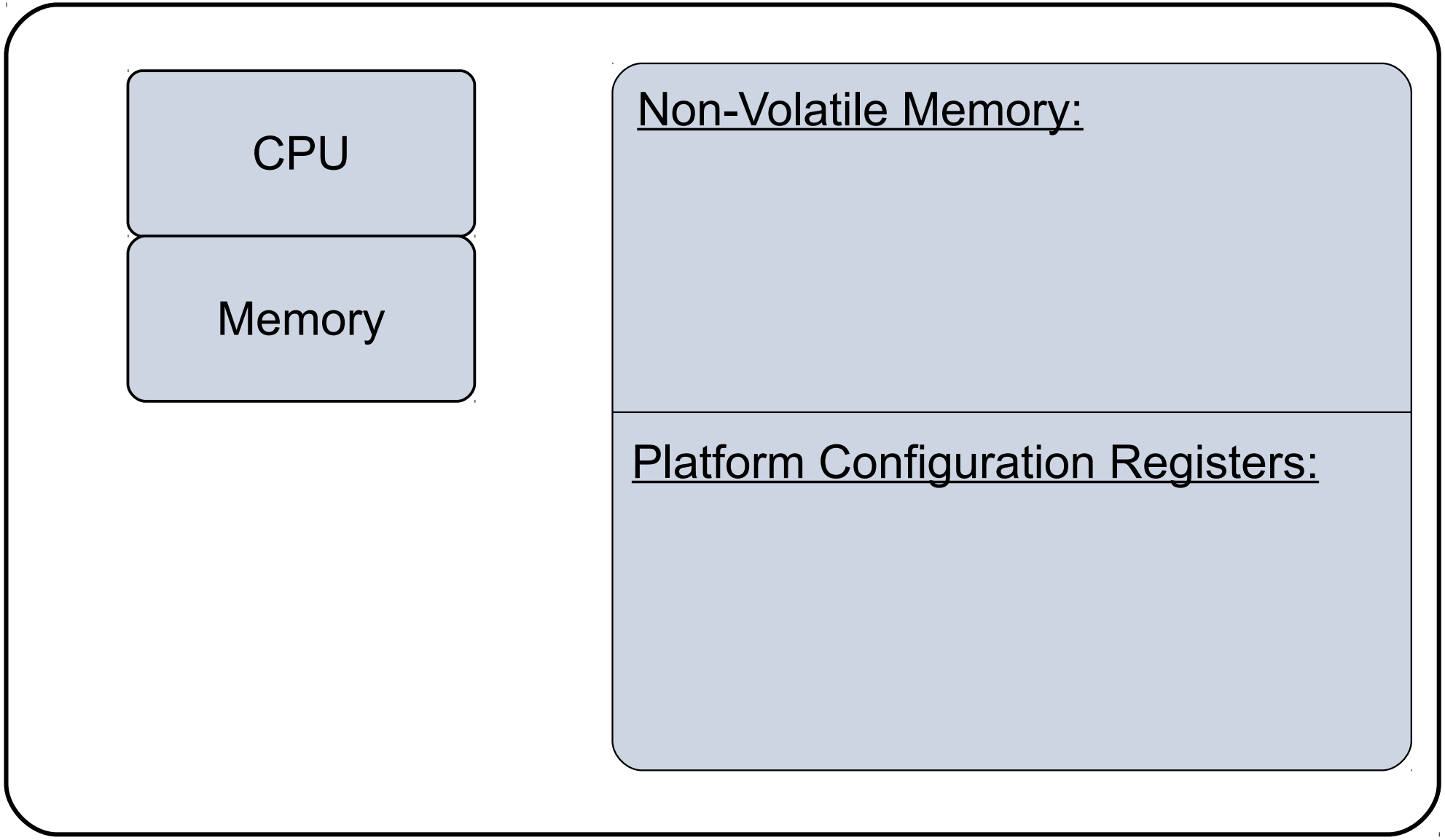
- "{ M }Sk^{pub} Message concealed ..."

  does not necessarily imply public key encryption for all of 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

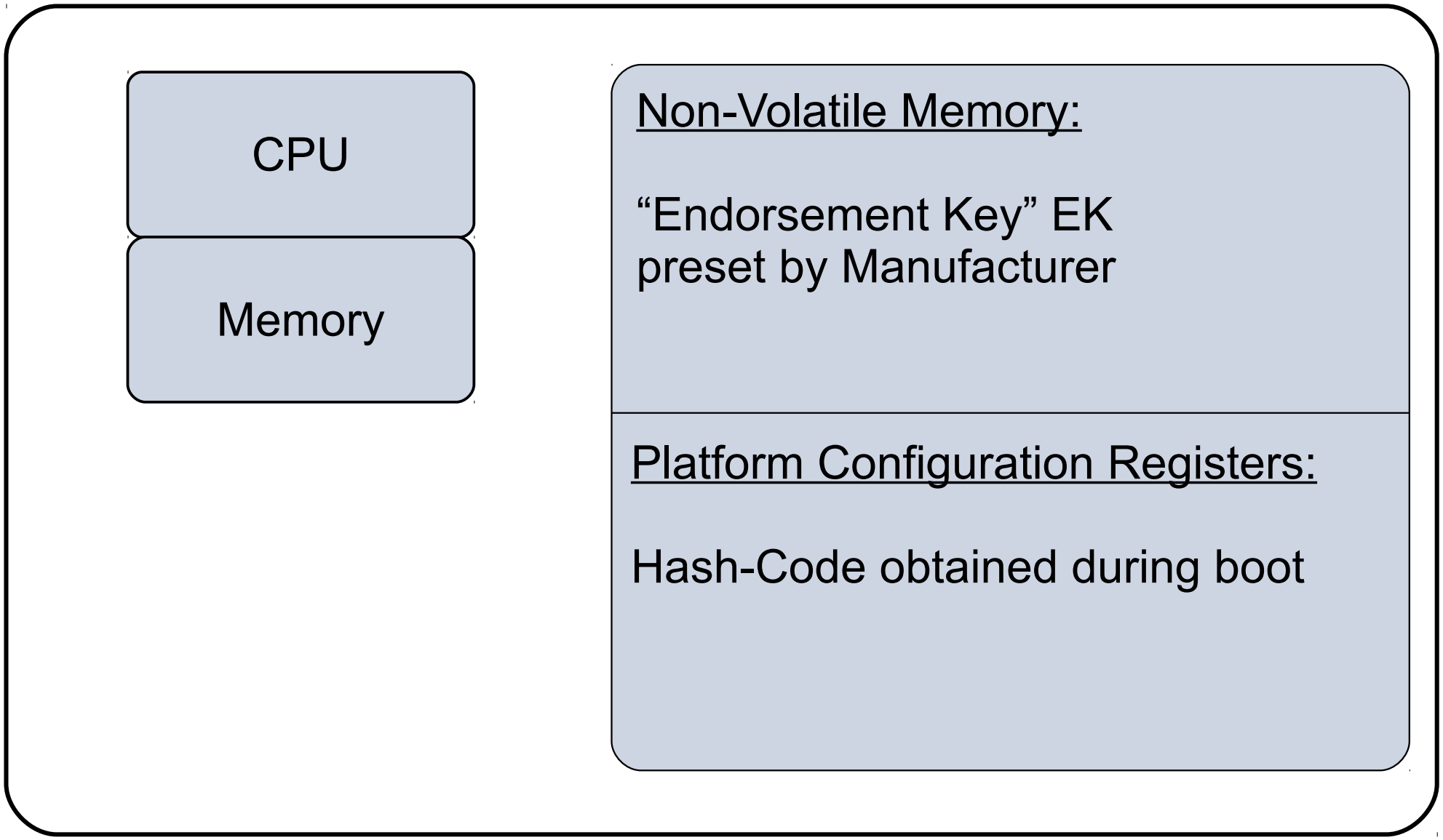
- Read- Only Memory
- Allowed H(OS) in NV memory preset by manufacturer
  - load OS- Code
  - compare H(loaded OS code) to preset H(OS)
  - abort if different
- Preset FSK^{pub} in NV memory preset by manufacturer
  - load OS- Code
  - check signature of loaded OS-Code using FSK^{pub}
  - abort if check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

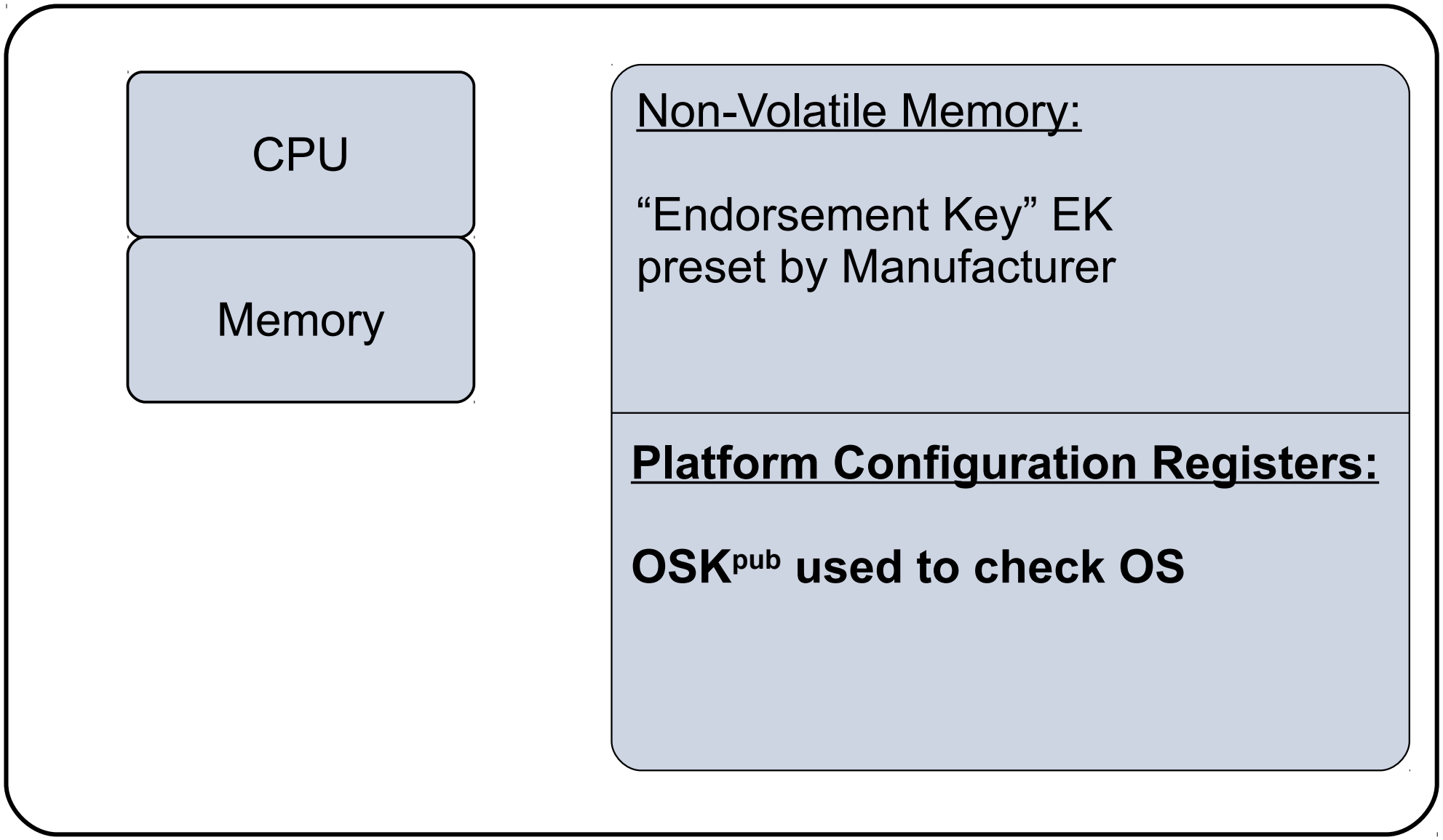
Attestation:

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

Remaining problems

- Now we know identities: H(loaded-OS) and EK^{pub}
- Problems to solve:
  - OS versioning
  - Remote attestation on each message (what about reboot ?)
  - not only “OS” on platform (SW stacks or trees)
  - Privacy: remote attestation always reveals EK^{pub}
  - Black box to big
  - Sealed memory

AB (Variant 2, 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 (Variant 2)

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 (Variant 3, check for 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 (Variant 3)

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 (Variant 3)

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}

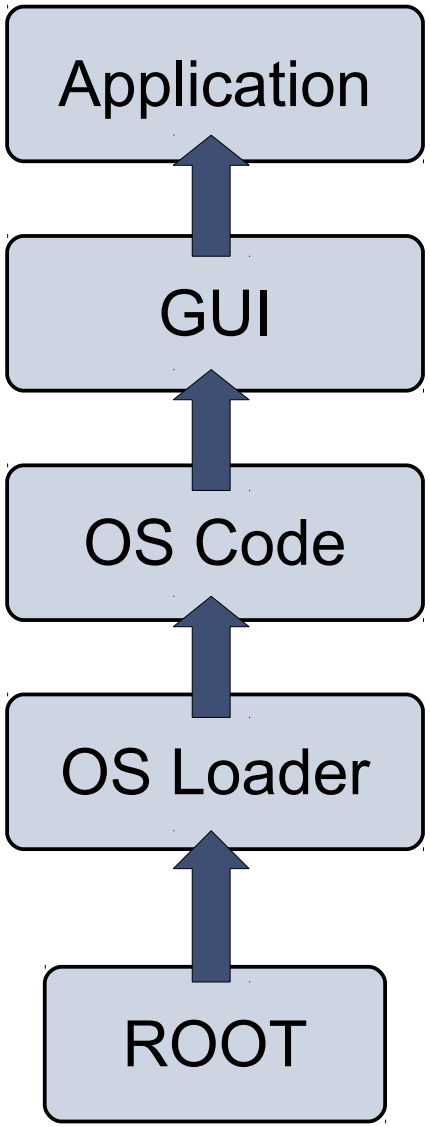
Secure Channel:

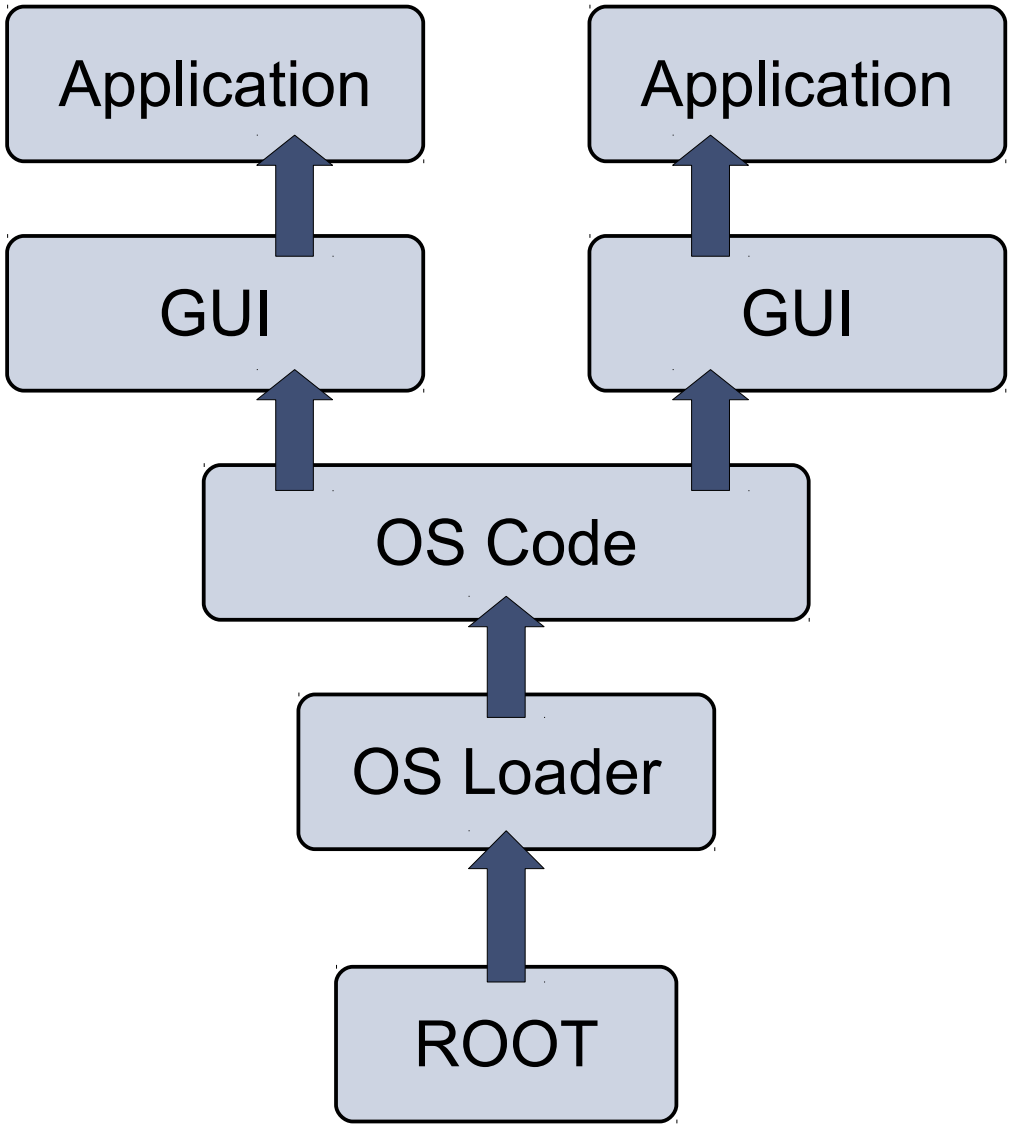
- { message } $OSrunningConsK^{pub}$

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

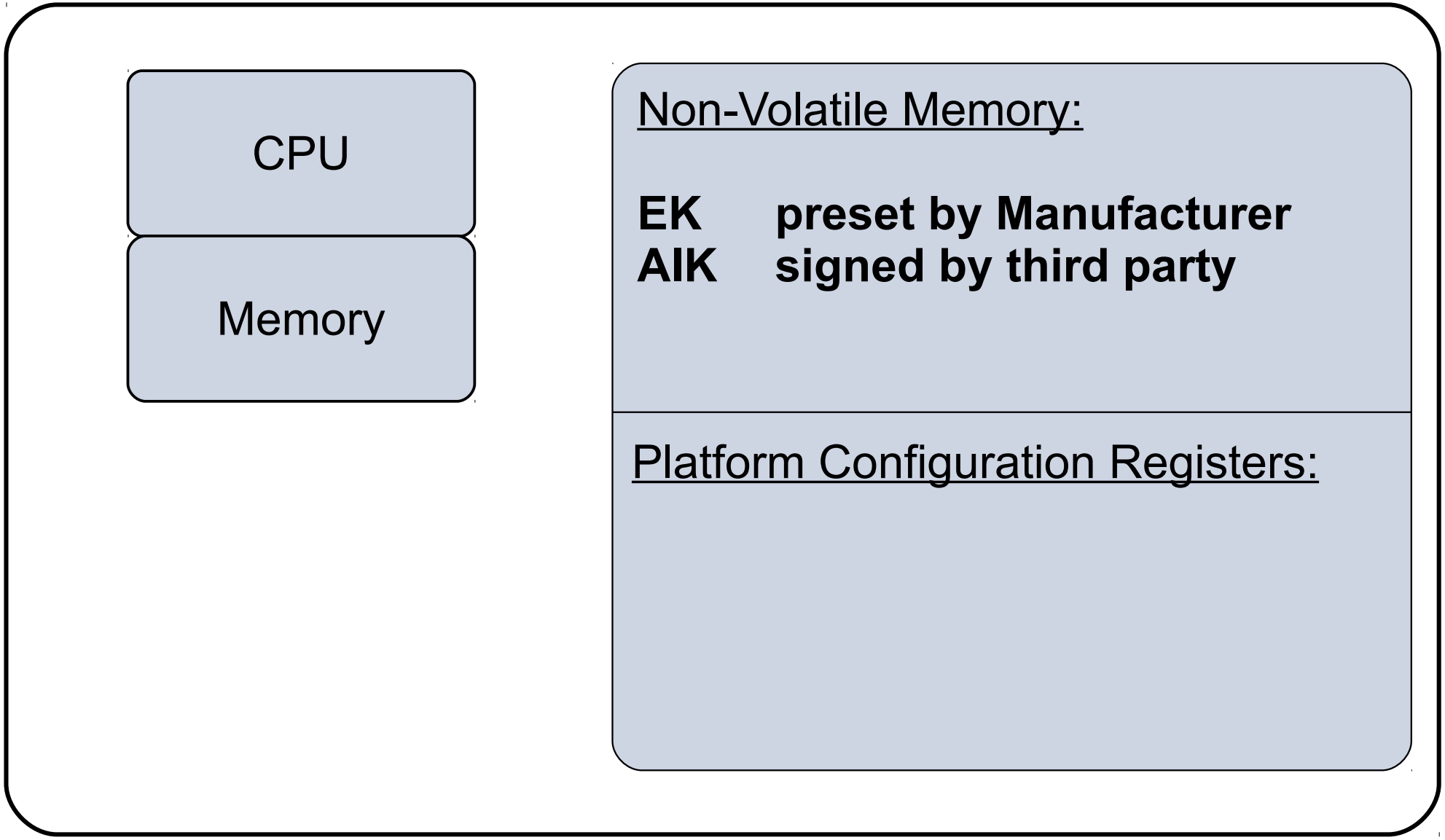
- "Extend" Operation
  - stack: $PCR_n = H(PCR_{n-1} \,||\, \text{next-component})$
  - tree: difficult (unpublished ?)

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

Remote Attestation and Privacy

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

Remote Attestation and Privacy



Remote Attestation and Privacy

- 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” } $TTPK^{priv}$
  - { $OSrunningK^{pub}$, H(OS)}AIK^{priv}
  - {nonce} $OSrunningK^{priv}$

Late Launch

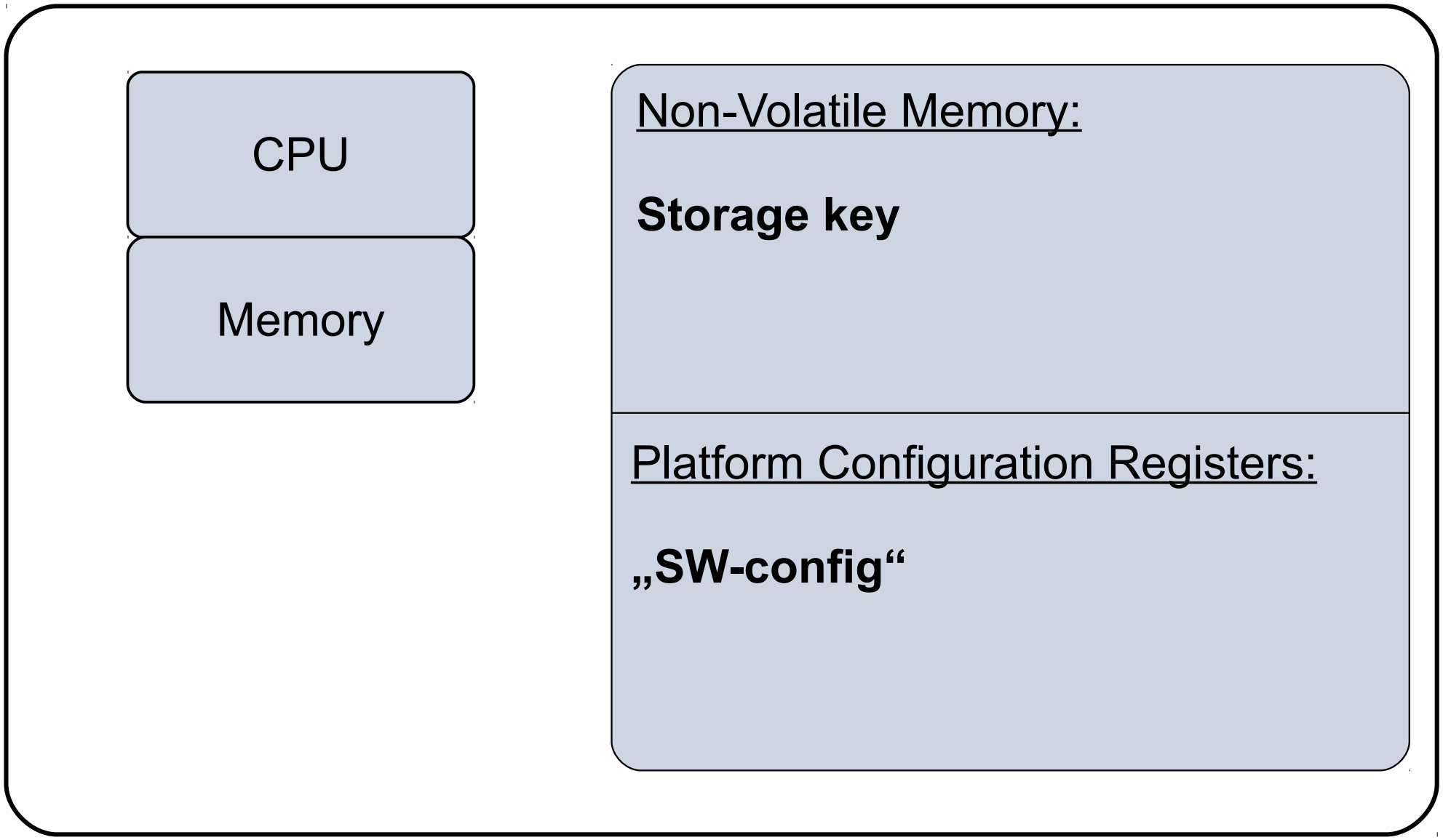
- Use arbitrary SW to start system and load all SW
- provide specific instruction to enter “secure mode”
  - set HW in specific state (stop all processors, IO, …)
  - Measure “root of trust” SW
  - store measurement in PCR

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

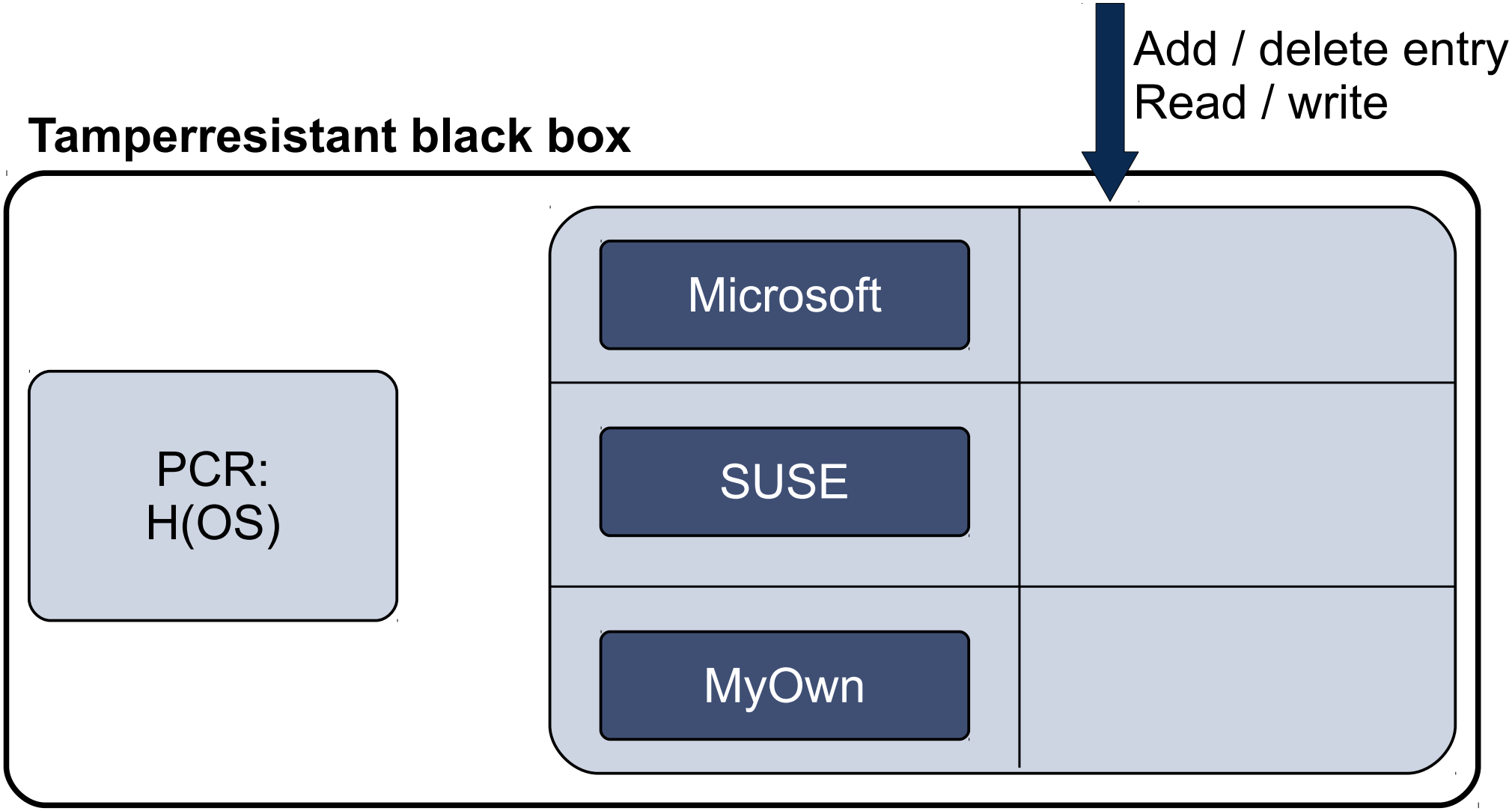
Sealed Memory

- Bind sensitive information to specific configuration
  (for example: keys to specific machine, specific OS)
- Provide information using secure channels
- How to store information in the absence of communication channels?

Tamperresistant black box (TRB)



Sealed Memory



Sealed Memory

- Seal(PCR, message):
  - encrypt(“PCR, message”, Storage-Key)
    → “sealed message”

- Unseal(sealed message):
  - decrypt(“sealed message”, Storage-Key)
    → “SW config, message”
  - If SW config == PCR then emit message else abort

Sealed Memory

- Seal(SW config, message):
  - encrypt(“future SW config, message”, Storage-Key)
    → “sealed message”

- “Storage Key” built into TPMs by manufacturer, known to nobody

Example

- Win7: 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

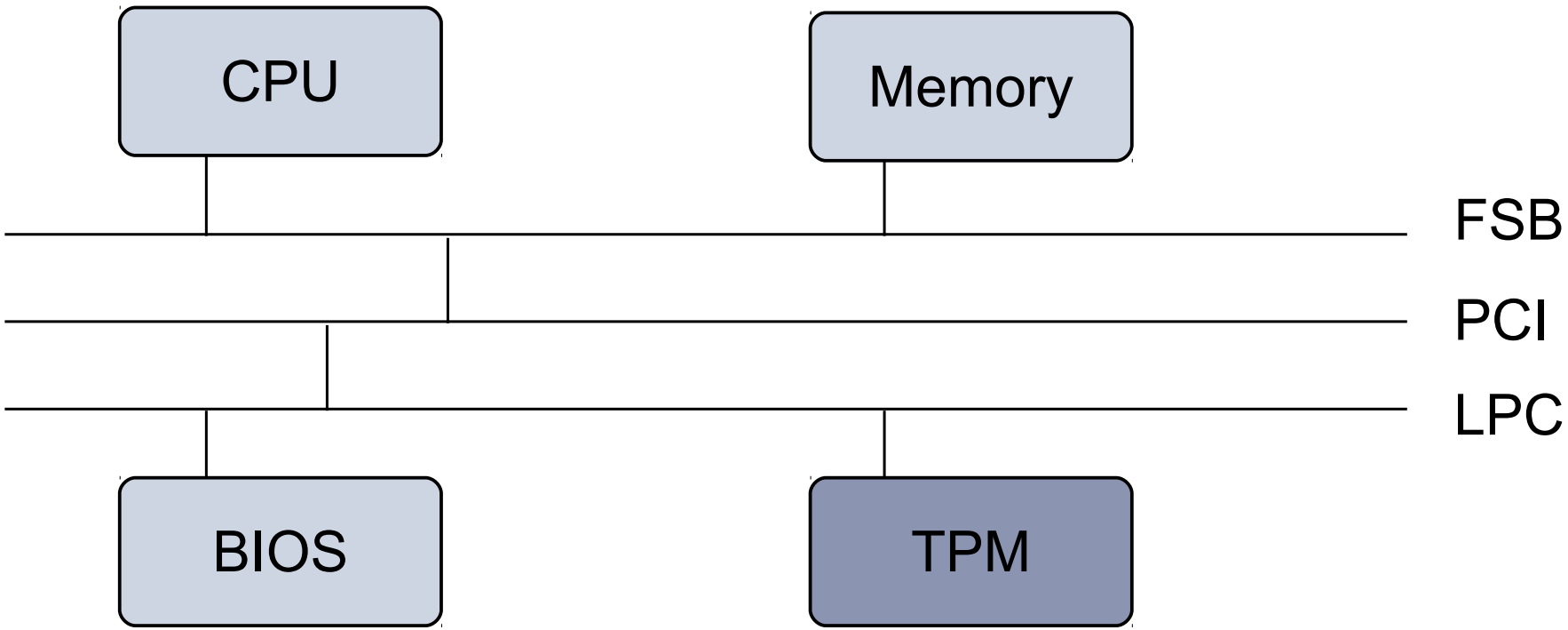
  Third party provides information to another entity

Tamper Resistant Box ?

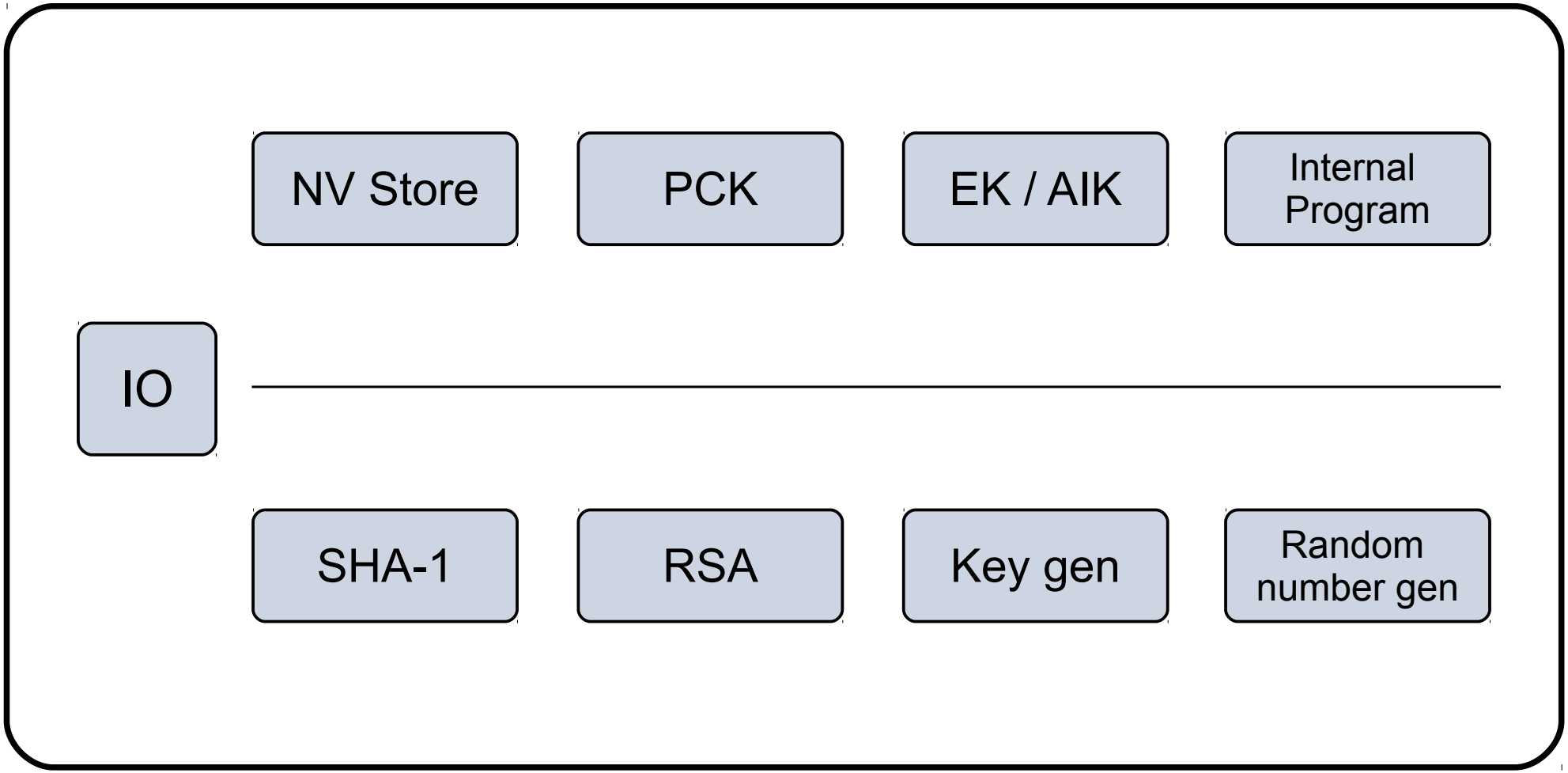
- Ideally, includes CPU, Memory, …

- In practise
  - very rarely, for example IBM 4758 ...
  - separate “Trusted Platform Modules” replacing BIOS breaks TRB

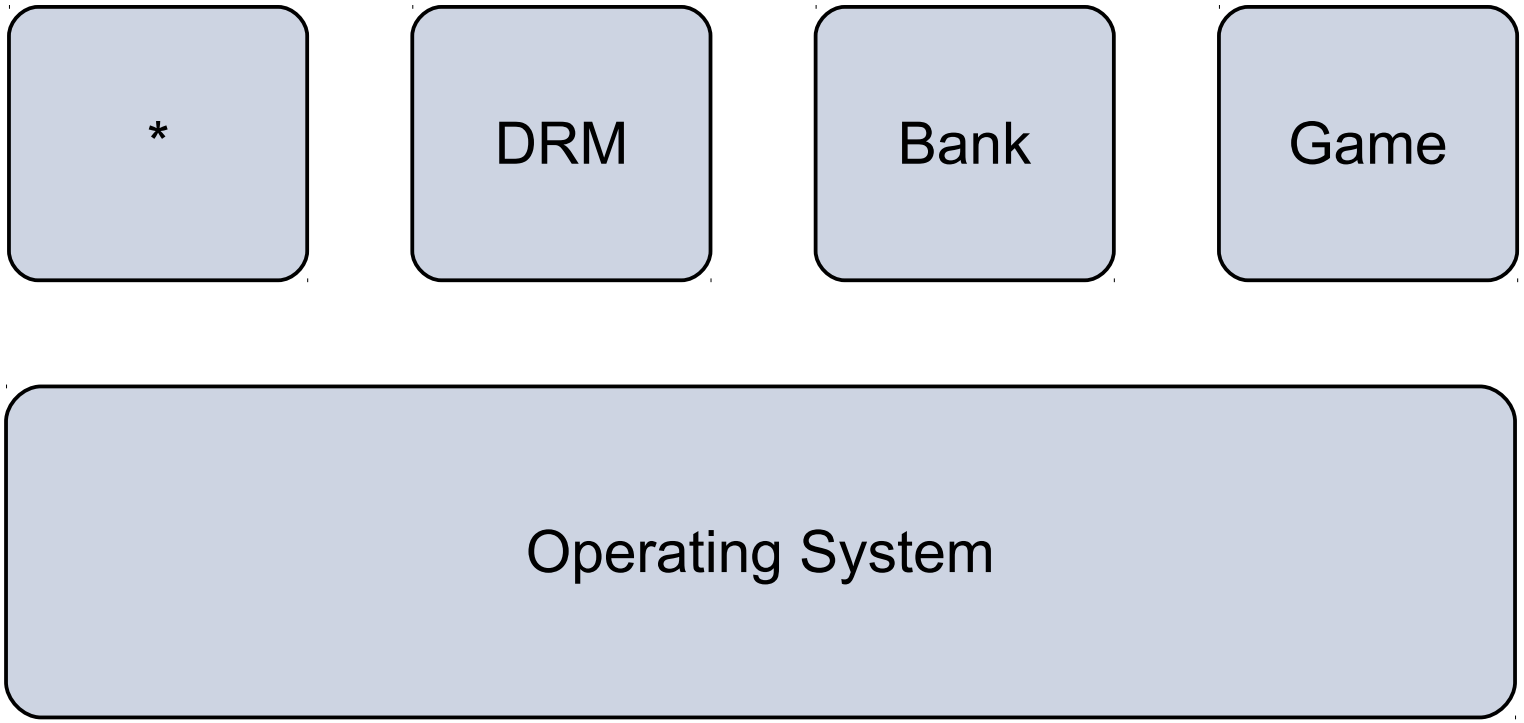
TCG PC Platforms



TPM


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

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)