SfiP
SECURITY & FW IP's

intel

INTRODUCTION TO TRUSTED PLATFORM MODULE (TPM 2.0)

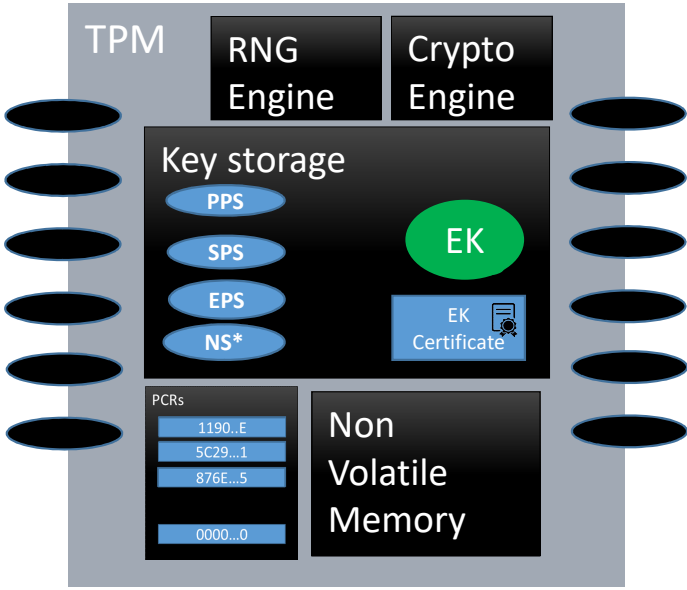
Liran Perez

Intel® PTT Product Owner and Firmware Architect
Liran.Perez@intel.com
November 2018

AGENDA

- TPM Introduction
  - Trust in Trusted computing
  - TPM 2.0
  - API - Command/Response interface

- Key Features and use cases
  - PCRs – Platform Configuration Registers
    - Measured Boot
  - Protected key storage and hierarchies
    - Endorsement key
    - Remote attestation
    - Hard disk encryption (Microsoft® BitLocker® / PGP etc.)
  - General purpose protected NV



INTRO – TPM 2.0 - TRUST

1. **Recognize platform properties**
2. **System behave as expected**
3. **Establish trust with other systems**

- TCPA and TCG
  - 1999 - Trusted Computing Platform Alliance (TCPA)
    Founding members: Compaq, Microsoft, Hewlett-Packard, IBM and Intel
  - Publish open standards for trusted computing
  - Define technology specification for any computing platform
  - Succeeded in 2003 by the TCG

INTRO - TPM 2.0 - TCG

TRUSTED COMPUTING GROUP®

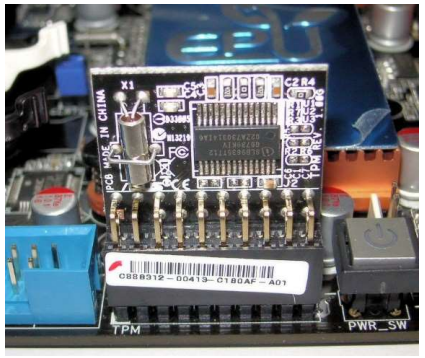
Promoters:

Contributing:

INTRO - TPM 2.0 – TCG

- Publish various **open** specifications for Trusted Computing
  "Not security by obscurity"
- Publish the TPM2.0 library specification (TPMWG)
  standardized in **ISO/IEC 11889:2015**
- Publish Platform Firmware and flows specification for different areas
  - PC-Client
  - Server
  - Embedded (IoT, Automotive, Industrial and more)
  - Storage and storage encryption
  - Trusted Network Computing (TNC)

INTRO – TPM 2.0

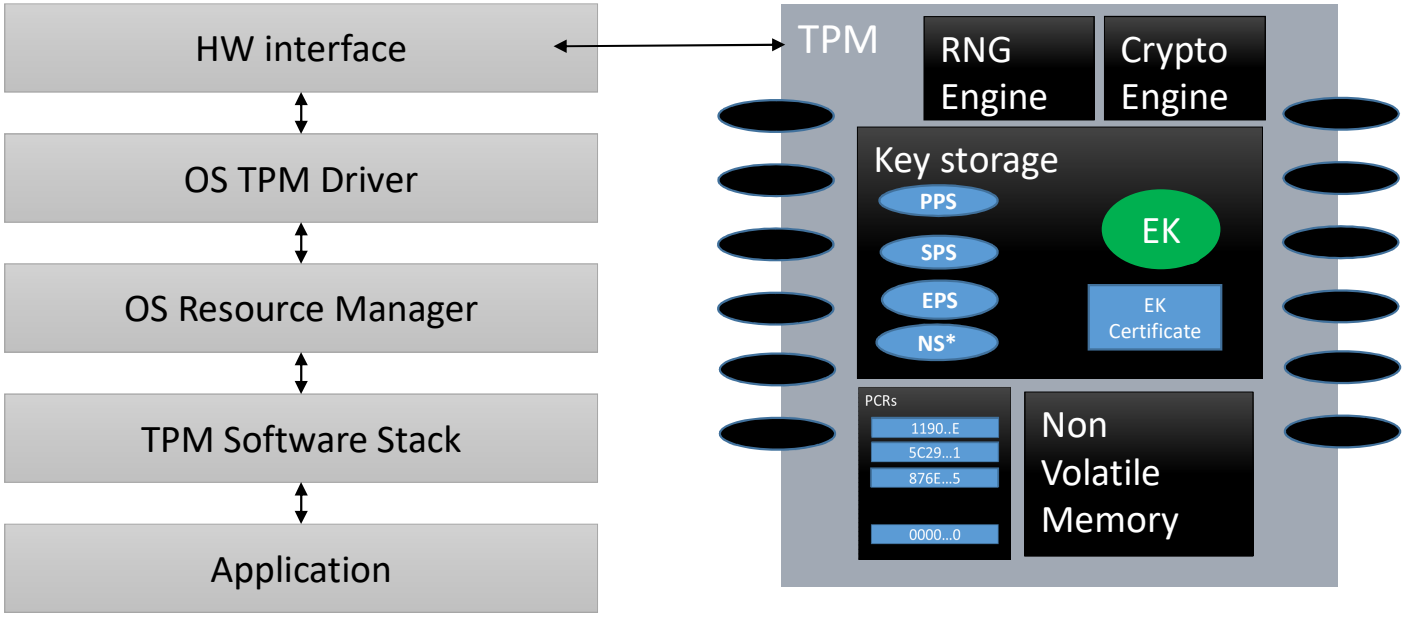


- TPM – Trusted Platform Module
  - Dedicated micro controller providing “trust” to the host platform
  - Provide secured environment for cryptography, secured **storage**, **measurements** and **reporting**
  - Provide API for authorization, policies and Dictionary Attack (DA) protection
  - Specification and API defined by the **Trusted Computing Group (TCG)**
    http://www.trustedcomputinggroup.org/
    http://www.trustedcomputinggroup.org/tpm-library-specification/
  - Use cases:
    Microsoft® BitLocker®/Health attestation/Virtual Smart-card/
    Windows Hello (Pin/Camera login)

INTRO - TPM 2.0 - ALGORITHMS

- SHA1, SHA256 (Hash/HMAC)
- AES-128/256 (ECB/CBC/CFB/OFB/CTR/ CMAC)
- RSA 1024/2048
  - RSAES-PKCS1-v1_5 / RSAES_OAEP (IETF RFC 8017)
  - RSASSA-PKCS1-v1_5 / RSASSA-PSS (IETF RFC 8017)
- ECC
  - ECDH (NIST SP800-56A)
  - ECDSA (ISO/IEC 14888-3)
  - EC-SCHNORR (TCG)
  - ECDAA (TCG)
- RNG and DRBG engines (NIST SP800-90)
- SM2 256, SM3 256, SM4 128/256

INTRO - TPM 2.0 - COMMAND/RESPONSE INTERFACE



INTRO - TPM 2.0 - COMMAND/RESPONSE INTERFACE

- Described in details in part 3
- Structures described in part 2

9.31 TPMI_ST_COMMAND_TAG

This interface type is used for the command tags.

The response code for a bad command tag has the same value as the TPM 1.2 response code (TPM_BAD_TAG). This value is used in case the software is not compatible with this specification and an unexpected response code might have unexpected side effects.

Table 68 — Definition of (TPM_ST) TPMI_ST_COMMAND_TAG Type

Values	Comments
TPM_ST_NO_SESSIONS	
TPM_ST_SESSIONS	
#TPM_RC_BAD_TAG	

14.2.2 Command and Response

Table 45 — TPM2_RSA_Encrypt Command

Type	Name	Description
TPMI_ST_COMMAND_TAG	tag	TPM_ST_SESSIONS if an audit, encrypt, or decrypt session is present; otherwise, TPM_ST_NO_SESSIONS
UINT32	commandSize	
TPM_CC	commandCode	TPM_CC_RSA_Encrypt
TPMI_DH_OBJECT	keyHandle	reference to public portion of RSA key to use for encryption Auth Index: None
TPM2B_PUBLIC_KEY_RSA	message	message to be encrypted NOTE 1 The data type was chosen because it limits the overall size of the input to no greater than the size of the largest RSA public key. This may be larger than allowed for *keyHandle*.
TPMT_RSA_DECRYPT+	inScheme	the padding scheme to use if *scheme* associated with *keyHandle* is TPM_ALG_NULL
TPM2B_DATA	label	optional label *L* to be associated with the message Size of the buffer is zero if no label is present NOTE 2 See description of label above.

Table 46 — TPM2_RSA_Encrypt Response

Type	Name	Description
TPM_ST	tag	see clause 6
UINT32	responseSize	
TPM_RC	responseCode	
TPM2B_PUBLIC_KEY_RSA	outData	encrypted output

INTRO - TPM 2.0 - COMMAND/RESPONSE INTERFACE

- Described in details in part 3
- Structures described in part 2

9.31 TPMI_ST_COMMAND_TAG

This interface type is used for the command tags.

The response code for a bad command tag has the same value as the TPM 1.2 respo
(TPM_BAD_TAG). This value is used in case the software is not compatible with this specificati
unexpected response code might have unexpected side effects.

Table 68 — Definition of (TPM_ST) TPMI_ST_COMMAND_TAG Type

Values	Comments
TPM_ST_NO_SESSIONS	
TPM_ST_SESSIONS	
#TPM_RC_BAD_TAG	

14.2.2 Command and Response

Table 45 — TPM2_RSA_Encrypt Command

Type	Name	Description
TPMI_ST_CO		TPM_ST_SESSIONS if an audit, encrypt, or decrypt
UINT32		
TPM_CC		
TPMI_DH_OE		
TPM2B_PUBI		
TPMT_RSA_I		
TPM2B_DAT/		

Table 19 — Definition of (UINT16) TPM_ST Constants <IN/OUT, S>

Name	Value	Comments
TPM_ST_RSP_COMMAND	0x00C4	*tag* value for a response; used when there is an error in the tag. This is also the value returned from a TPM 1.2 when an error occurs. This value is used in this specification because an error in the command tag may prevent determination of the family. When this tag is used in the response, the response code will be TPM_RC_BAD_TAG (0 $1E_{16}$), which has the same numeric value as the TPM 1.2 response code for TPM_BADTAG. NOTE In a previously published version of this specification, TPM_RC_BAD_TAG was incorrectly assigned a value of 0x030 instead of 30 (0x01e). Some implementations my return the old value instead of the new value.
TPM_ST_NULL	0X8000	no structure type specified
TPM_ST_NO_SESSIONS	0x8001	*tag* value for a command/response for a command defined in this specification; indicating that the command/response has no attached sessions and no *authorizationSize*/*parameterSize* value is present If the *responseCode* from the TPM is not TPM_RC_SUCCESS, then the response tag shall have this value.
TPM_ST_SESSIONS	0x8002	*tag* value for a command/response for a command defined in this specification; indicating that the command/response has one or more attached sessions and the *authorizationSize*/*parameterSize* field is present

Type
TPM_ST
UINT32
TPM_RC
TPM2B_PUBI

INTRO - TPM 2.0 - COMMAND/RESPONSE INTERFACE

- Described in details in part 3
- Structures described in part 2
- Marshalled in big endian (always!)
- Divided to 4 areas
  - Header
  - Handles
  - Authorizations/Sessions
  - Parameters
- TSS – MSFT/TCG/Intel/IBM

http://www.trustedcomputinggroup.org/tcg-software-stack-tss-specification/
https://github.com/tpm2-software/tpm2-tss
https://github.com/Microsoft/TSS.MSR
https://trustedcomputinggroup.org/wp-content/uploads/TPM-Rev-2.0-Part-3-Commands-01.38-code.pdf
https://trustedcomputinggroup.org/wp-content/uploads/TPM-Rev-2.0-Part-2-Structures-01.38.pdf

14.2.2 Command and Response

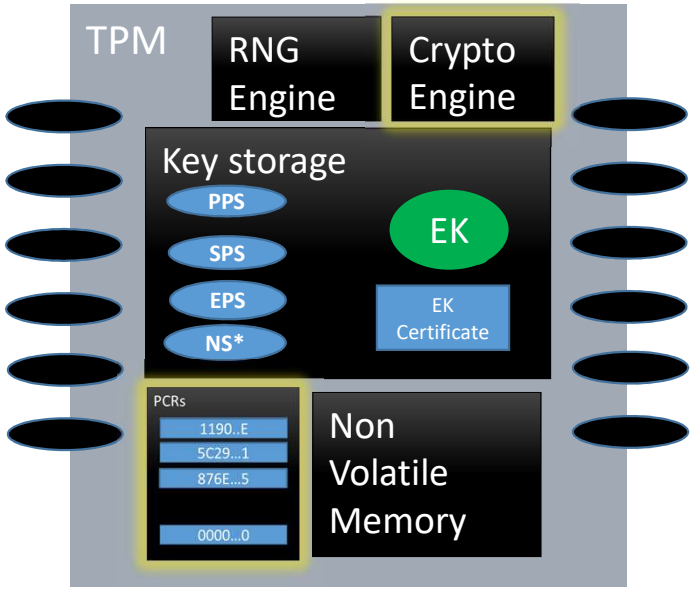
Table 45 — TPM2_RSA_Encrypt Command

Type	Name	Description
TPMI_ST_COMMAND_TAG	tag	TPM_ST_SESSIONS if an audit, encrypt, or decrypt session is present; otherwise, TPM_ST_NO_SESSIONS
UINT32	commandSize	
TPM_CC	commandCode	TPM_CC_RSA_Encrypt
TPMI_DH_OBJECT	keyHandle	reference to public portion of RSA key to use for encryption Auth Index: None
TPM2B_PUBLIC_KEY_RSA	message	message to be encrypted NOTE 1 The data type was chosen because it limits the overall size of the input to no greater than the size of the largest RSA public key. This may be larger than allowed for *keyHandle*.
TPMT_RSA_DECRYPT+	inScheme	the padding scheme to use if *scheme* associated with *keyHandle* is TPM_ALG_NULL
TPM2B_DATA	label	optional label *L* to be associated with the message Size of the buffer is zero if no label is present NOTE 2 See description of label above.

Table 46 — TPM2_RSA_Encrypt Response

Type	Name	Description
TPM_ST	tag	see clause 6
UINT32	responseSize	
TPM_RC	responseCode	
TPM2B_PUBLIC_KEY_RSA	outData	encrypted output

KEY FEATURES – PLATFORM CONFIGURATION REGISTERS (PCR)



KEY FEATURES - PCRS

- TPM 2.0 Provides API for secure measurement of **large size data flow** in a **small digest size** register.

REMINDERS ☺

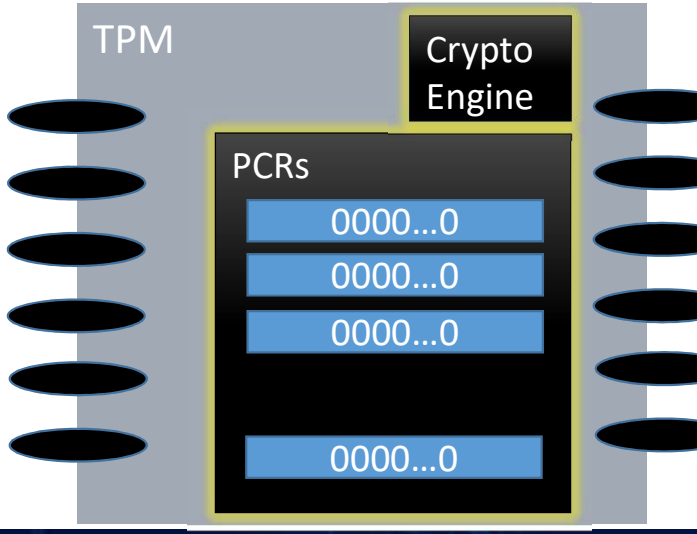
Hash function:

Any function which can map arbitrary size data (message) to fixed size (digest)

-Wikipedia more or less…

Cryptographic hash function:

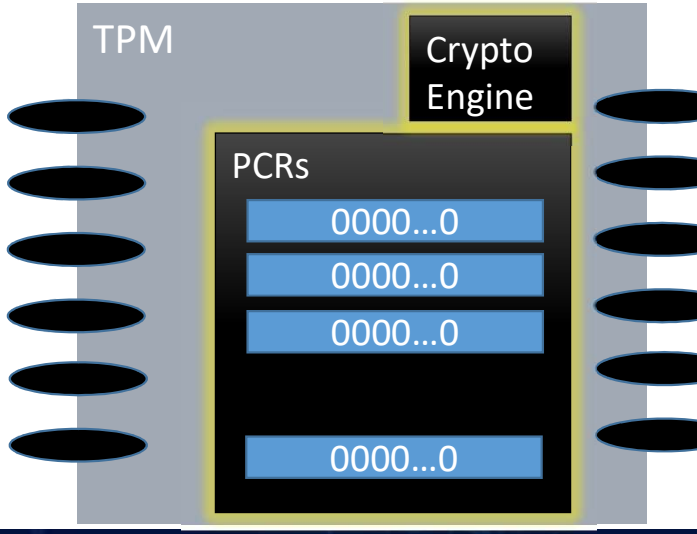
- Deterministic
- Easy to compute (for most cases…)
- Delicate to small change
- ***PreImage resistance:*** Given image - Infeasible to find a source
- ***2nd PI resistance:*** Given source - Infeasible to find an 'equal' source
- ***Collision resistance:*** Infeasible to find 2 sources with same digest



KEY FEATURES - PCRS

- TPM 2.0 Provides API for secure measurement of **large size data flow** in a **small digest size** register.
- PCR value can be modified only with the EXTEND operation:

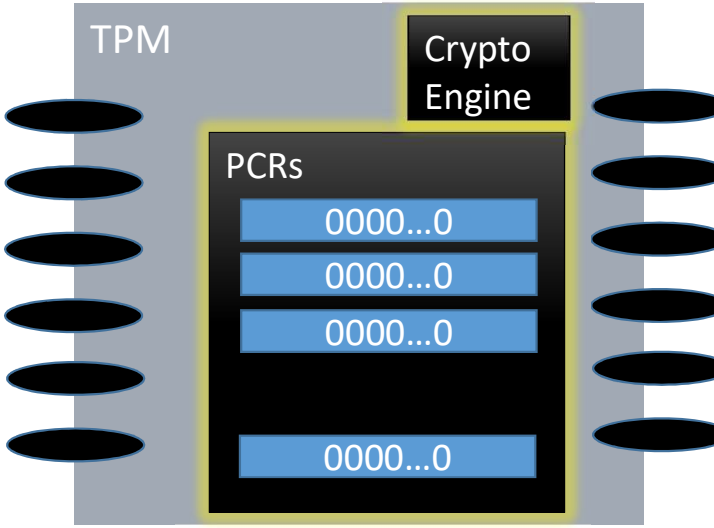
$$EXTEND(PCR, digest):$$
$$PCR_{new} := HASH\ (PCR_{old} || digest)$$



  - The only API which allow to modify a PCR:
    - TPM2_PCR_Extend
    - TPM2_PCR_Event
    - TPM2_EventSequenceStart

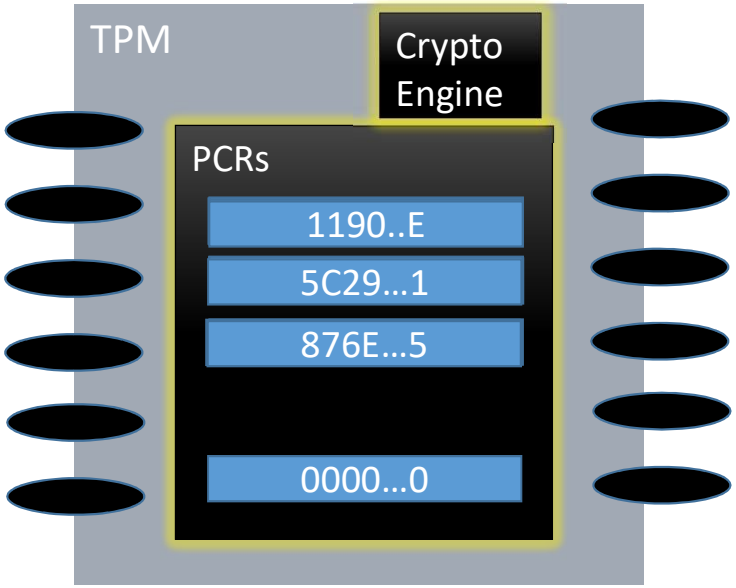
KEY FEATURES - PCRS

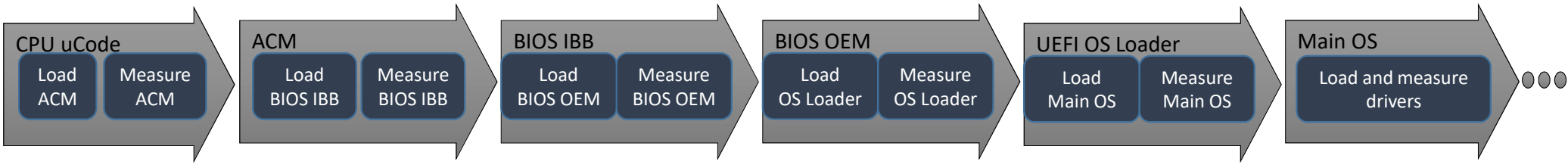
- Final PCR value attest the flow that was recorded
  - Minor change in single measurements is accumulated in the PCR
  - Order is recorded: Extend(A) -> Extend(B) ≠ Extend(B) -> Extend(A)
- TPM may implement multiple PCR **banks**
  PC Client requires 24 PCRs of SHA256
- Every PCR have a defined initial state and access control (Locality)
- Use cases:
  - **Measured Boot**
  - **Remote or local attestation** of HW/FW/SW state
  - **Sealing objects** (VPN keys/ **Windows BitLocker®**)



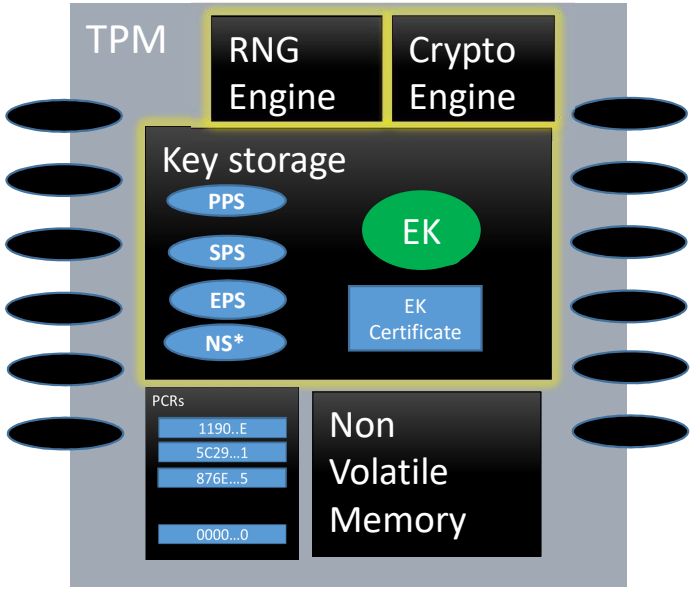
EXAMPLE: MEASURED BOOT

- Root of trust – preferably an immutable code
- Chain of trust
  - Load
  - Measure
  - Launch

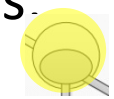




KEY FEATURES - PROTECTED STORAGE



KEY FEATURES - PROTECTED STORAGE - HIERARCHIES

- TPM maintains 4 protected hierarchies.
  Platform, Owner, Endorsement and Null
- Each hierarchy has a seed value and authorization values
- Every object in the hierarchy is “protected” by it’s parent
- There are few kind of roles in those hierarchies:
  - Storage  / Ordinary 
  - Primary (seed derived)  /Child (RNG/DRBG) 
  - TPM generated/User generated sensitive

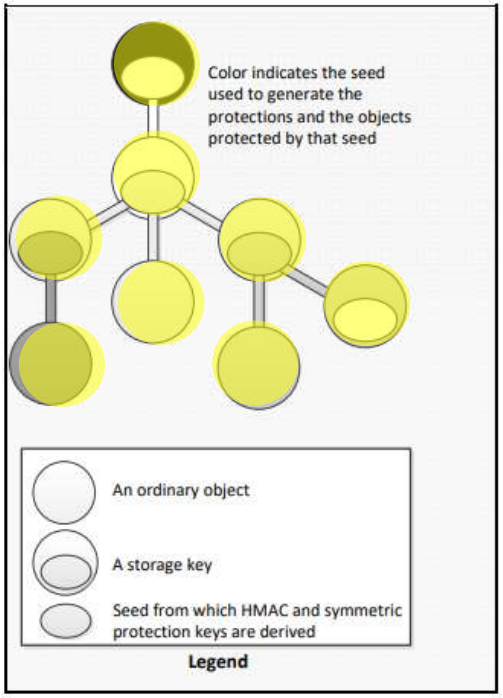
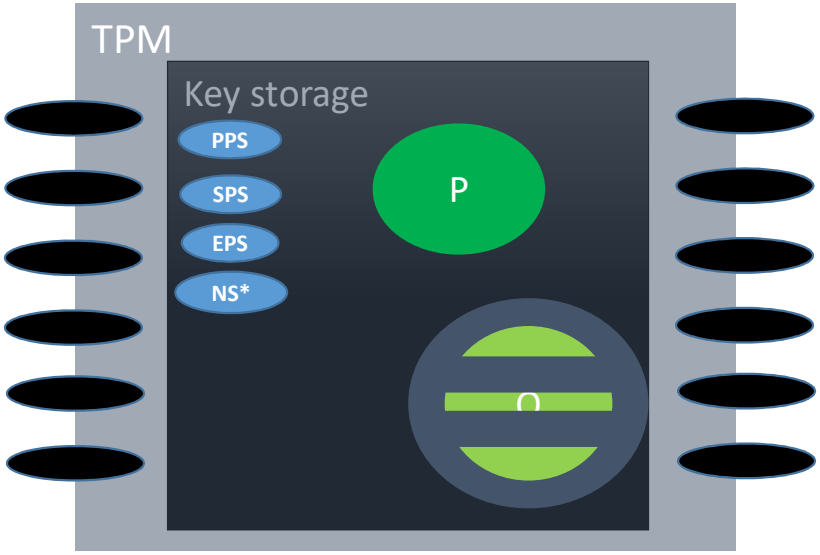


Figure 18 — Symmetric Protection of Hierarchy

https://trustedcomputinggroup.org/wp-content/uploads/TPM-Rev-2.0-Part-1-Architecture-01.38.pdf

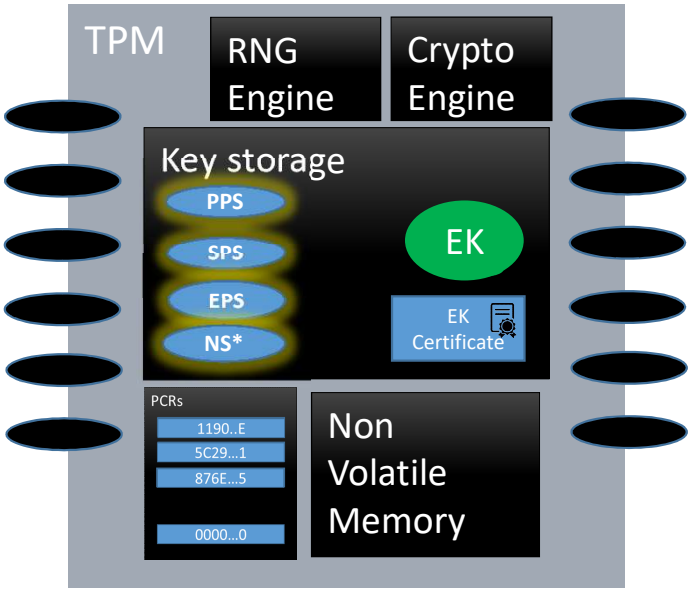
KEY FEATURES - PROTECTED STORAGE - OBJECT HIERARCHIES

- **TPM2_CreatePrimary**
  Create and load an object from the chosen hierarchy primary seed
  Usually never leaves TPM
- **TPM2_Create**
  Create an object from RNG
  Returned to caller when created
- **TPM2_Load**
  A child can be loaded at any time when its parent is loaded
- **TPM2_CreateLoaded**
  Create a loaded key (primary or RNG)



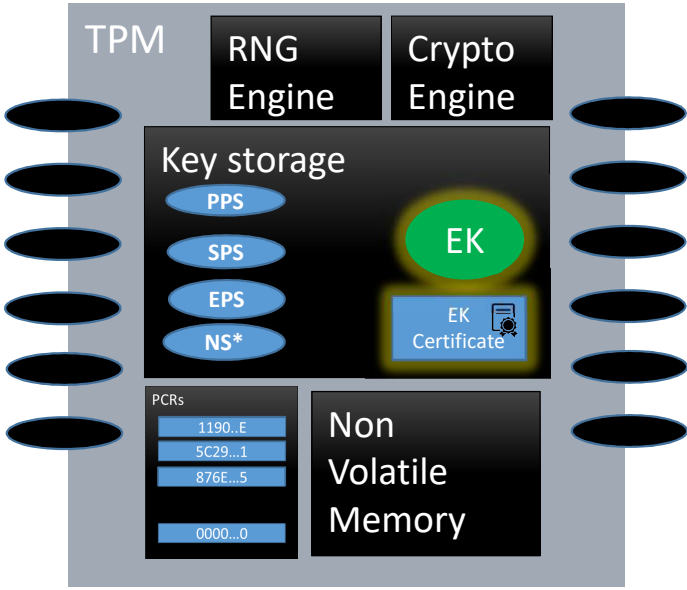
KEY FEATURES - PROTECTED STORAGE - HIERARCHIES

- Platform
  - Authorization resets on each boot, lifetime seed (*)
  - RTM- Root of trust for measurement
- Owner (aka Storage)
  - Authorization saved in NV. Seed resets by TPM2_Clear
  - Storage Root Key (SRK)
- Endorsement
  - Endorsement root key (EK). Seed change by TPM2_ChangeEPS.
- Null
  - Null (Empty buffer) auth. Seed resets on restart and reset.



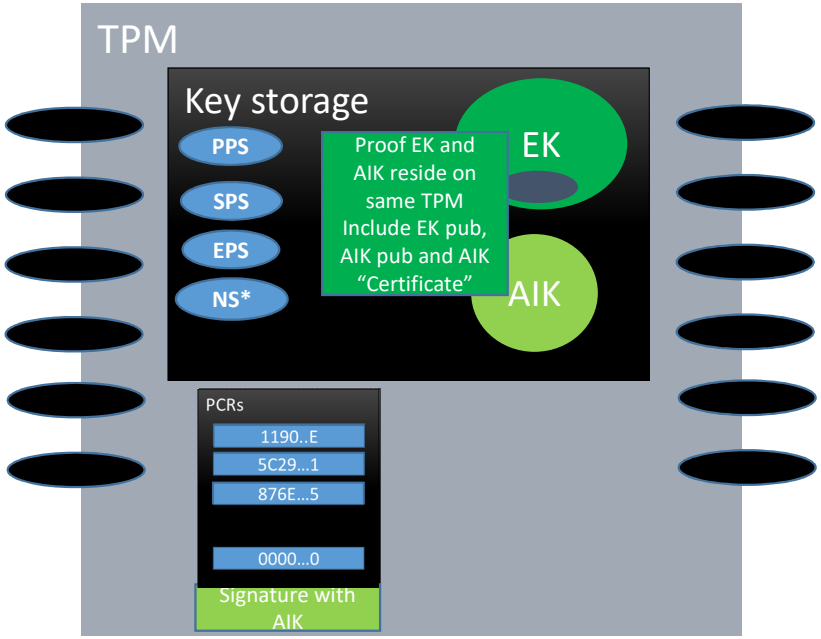
KEY FEATURES - PROTECTED STORAGE – ENDORSEMENT KEY

- A TPM must be shipped with a special unique key called the EK
- The TPM must provide a valid x509 certificate for the EK signed by the TPM vendor
- The EK is the Root of Trust for Reporting (RTR) of the endorsement hierarchy and used to certify the TPM Attestation Identity Keys (AIKs)
- Used in remote attestation flows



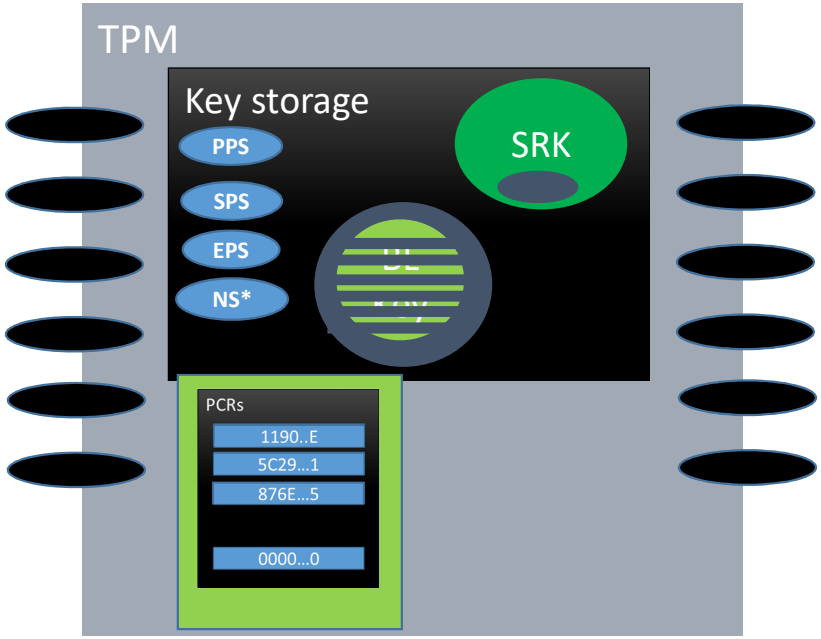
EXAMPLE: REMOTE ATTESTATION

- Create the certified EK
  TPM2_CreatePrimary with special template
- Create a “restricted sign key”
  TPM2_Create
  Attestation Identity Key (AIK)
- Bind the AIK to the EK
  TPM2_ActivateCredentials
- Attest the PCRs value (Sign)
  TPM2_Quote



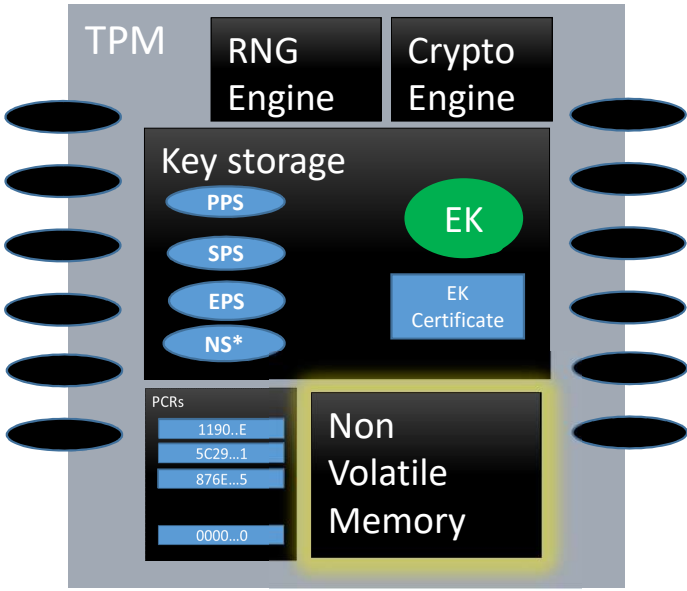
EXAMPLE: HARD DISK ENCRYPTION

- Provision
  - Create the SRK
    TPM2_CreatePrimary with special template
  - Create a "sealed" data object
    TPM2_Create using required policies (i.e. PCRs) and PIN
- Use:
  - Load the object
    TPM2_Load
  - Assert the required policies
    TPM2_Policy* with a policy session
  - Unseal the object
    TPM2_Unseal



KEY FEATURES - NV INDICES / OBJECTS

- TPM 2.0 provides API for general purpose Non-Volatile memory
- General purpose data/counters/bits/PIN counters
- Protected by the Platform or Owner hierarchy or by a defined Policy/Authorization
- NV (brief) API:
  - TPM2_NvDefineSpace, _NvUnDefineSpace
  - TPM2_NV_Extend/_Increment/_SetBits /_NV_Read /_NV_Write
  - TPM2_NV_ReadLock/_WriteLock/_GlobalWriteLock
  - TPM2_EvictControl*

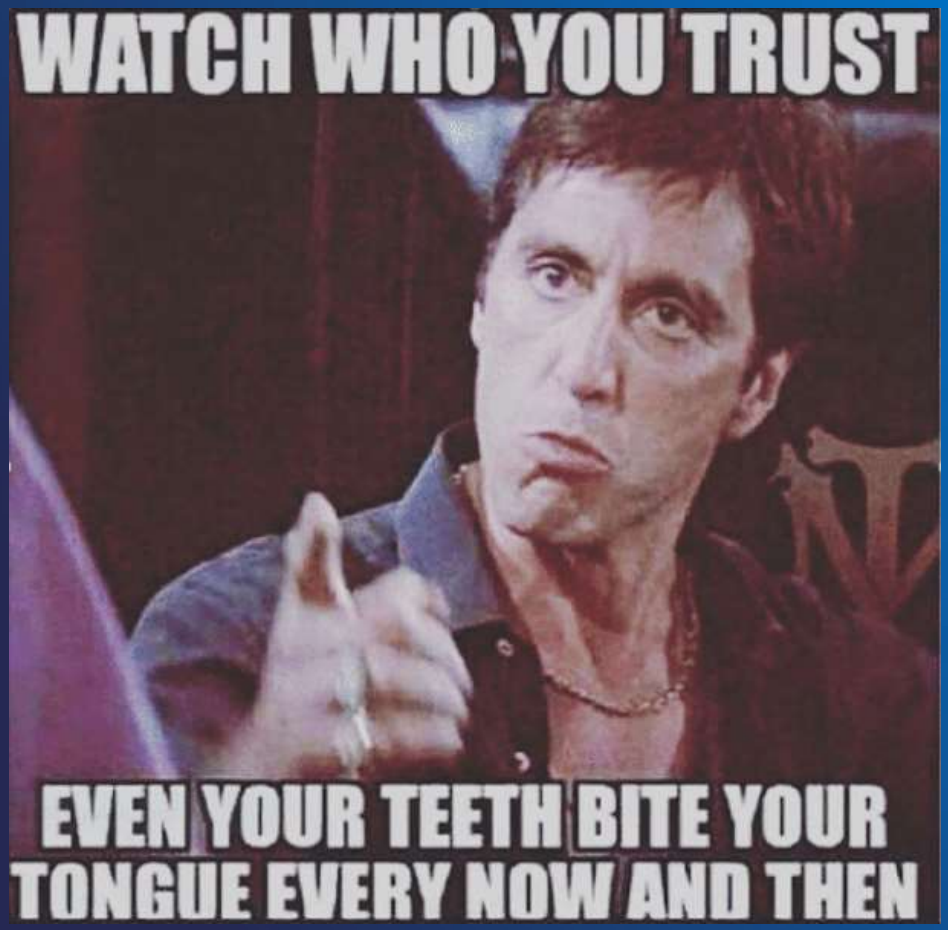


TPM 2.0 – INTEL® PLATFORM TRUST TECHNOLOGY (INTEL® PTT)

- Intel® PTT is Intel's TPM 2.0 implementation, integrated with Intel® **C**onverged **S**ecurity and **M**anageability **E**ngine (Intel® CSME)
- Not a traditional firmware TPM 2.0
- Has advantages and disadvantages vs discrete TPM
- Supports TCG's TPM 2.0 Library Specification revision 1.16 since Apollo Lake and Skylake, and revision 1.38 since Gemini Lake and Coffee Lake
- Meets Microsoft certification
- Reduces BOM cost

SfiP
SECURITY & FW IP's

intel



THANK YOU !