TPM and DevID discussion

Boris Balacheff
HP Security Office
HPLabs Trusted Systems Lab

A few TPM design principles

- Most cryptographic primitives - but not bulk encryption
- Privacy - Fully “opt-in”, and prevention of easy identity correlation
- No global secrets - If a TPM is cracked, it reveals information relating to the associated platform and nothing further
- Provides a low cost protected environment
- Ubiquitous security - at very low cost

Trusted Platform Mechanisms

- Platform Authentication
  - Identify a platform TCG properties to a challenging party
- Attestation or Integrity Reporting
  - Reliably measure and report on the platform's software configuration
- Protected Storage
  - Protect private and secret data on that platform

Two Roots of Trust

A Root of Trust for Measurement – A component that can be trusted to reliably measure and report to the Root of Trust for Reporting (the TPM) what software executes at the start of platform boot

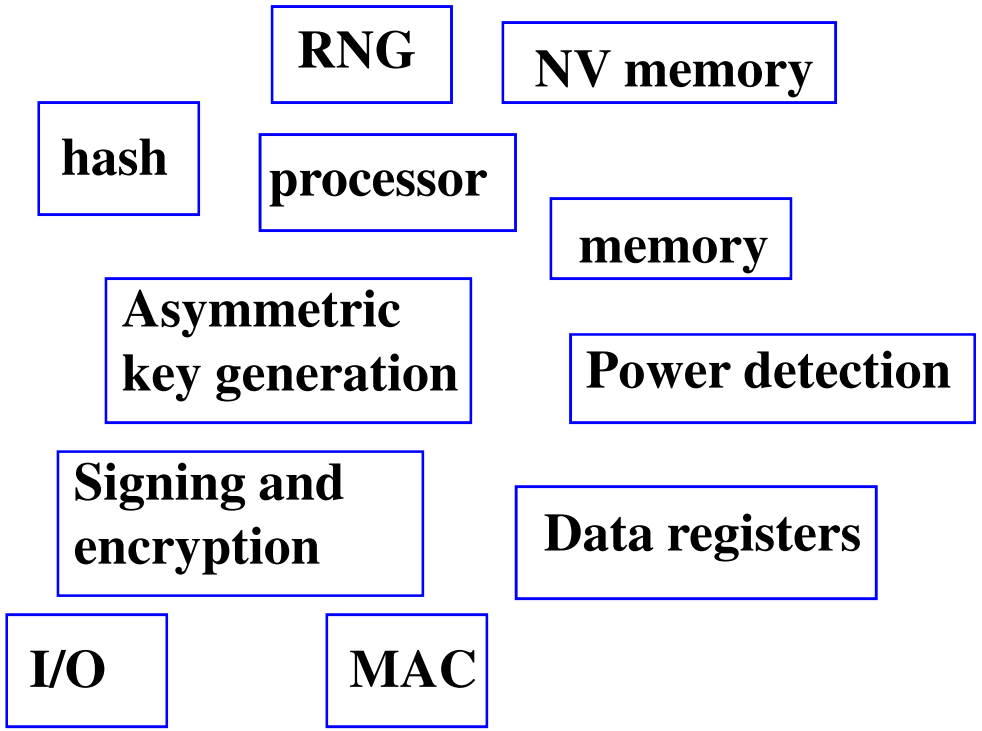
A Root of Trust for Reporting (the TPM) – The component that can be trusted to store and report reliable information about the platform

It must be possible for a third-party to establish trust in these Roots of Trust remotely and at runtime

=> The TCG relies on the use of Conformance and Certification

The Trusted Platform Module - TPM

- The TPM is the Root of Trust for Reporting
  - The TPM is trusted to operate as expected (conforms to the TCG spec)
  - The TPM is uniquely to a single platform
  - TPM functions and storage are isolated from all other components of the platform



The Trusted Platform Module

- Creates, stores and manages and protects cryptographic keys
- Performs cryptographic functions
  - RSA, SHA-1, RNG
- Ownership model that requires administrative setup
- Comes with a unique Endorsement Key (EK)
- Supports storing integrity measurement and reporting

TPM Endorsement Key

A TPM is designed to ship with and Endorsement key and an Endorsement Certificate from the manufacturer.

The Endorsement key usage is designed to:

- Enable to remotely establish trust in the manufacturer and therefore in the operations of the TPM
- Provide online proof to a third-party that a newly generated key was indeed generated and is protected by a genuine TPM of a certain origin
- Enable post-deployment generation and certification of signature keys

TPM Ownership and keys

- Only key before “TPM Ownership” is taken is the Endorsement key: privacy positive design choice
- TPM is initialized by an “owner”
  - Take Ownership:
    - Sets up authorization data for TPM owner
    - Initializes the key hierarchy: generates a Storage Root Key
- TPM clear/reset will reset Ownership
  - Owner controlled or based on physical presence
- Key creation requires Ownership to be established
- Each key can have an independent authorization value

Key Migration

- TPM keys can be created to be
  - Migratable: to support mobility or recovery needs

  OR
  - Non-Migratable: hardware identity (provable by the TPM)

- Migratable keys can be recovered/moved to other systems under TPM ownership privileges

- Non-Migratable keys can never exist outside a TPM.
  - Non-migratable keys are intended as IDs. New ones should be created when lost.

TPM and DevID alignment thoughts

- DevID functional requirements match the TPM well
  - Device authentication using asymmetric crypto
  - Digital signature operation is the one DevID requirement that needs to align with TPM
- Leveraging TPM to generate/protect DevID keys
  - Important for endpoints that will implement DevID
  - Multiple options exist to implement DevID with TPM
  - Crypto alignment necessary

Crypto Alignment

- TPM 1.2 algorithms:
  - RSA
    - RSASSA-PKCS1-v1.5 as defined in PKCS #1v2.0 with SHA-1 as hash operation
  - SHA-1

- DevID
  - Which signature algorithm and padding for DevID signature?
  - other questions:
    - 6.3.3 what does “128 bit security” if RSA key security is 2048 bits?
    - 6.3.4 why is AES mandated?

> consider supporting RSASSA-PKCS1-v1.5 w/ SHA-1 for signature

LDevID and IDevID - questions

- The role of LDevID vs IDevID?
  - Today: IDevID provides a persistent root of trust (credential) to:
    - allow to create DevID(s)
    - or be used as a DevID
  - What seems to be required is really one of:
    - a mechanism to securely establish DevID(s) post-manufacturing (i.e. a DevIDRoot)
    - a burnt in DevID
  - Where DevID(s) can be used to authenticate the device in an 802.1 protocol


hp
invent