# Introduction to Trusted Computing Concepts and the Trusted Platform Module (TPM) 2.0

# Introduction to Trusted Computing Concepts and Trusted Platform Module 2.0

G.J.Proudler, TCG Technical Committee Consultant

# The difficulty

- Security tends to get in the way of things that customers want to do
  - The less intrusive the security, the better the security as far as the customer is concerned
- The less obvious the security, the less that customers appreciate the security
  - It becomes less and less obvious that security exists and is providing protection
- The less that customers appreciate security, the more difficult it is to invest in security

# Secure computing

- Conventional fully-secure computing platforms are extremely desirable because they do exactly what they are supposed to do, nothing more and nothing less
- They are uncommon because they are difficult to design and maintain
  - it's cheap and easy to force a product to boot specific software, *but*
    - not all data necessarily needs protection, or needs the same protection
    - it's non-trivial to support updates, flexible functionality, user choice, and user privacy

# Trusted Computing

Trusted Computing allows a computing platform to do whatever it likes, but stops it pretending to be what it is not

# Support by Trusted Computing for different security levels

**fully secure**
Used in only exceptional circumstances

Doesn't need Trusted Computing, but it is still often cost-effective to use a trusted platform for protecting data-at-rest

**less-than-fully-secure**
Normal commercial operation

A trusted platform helps to ensure that identities and secrets can't be used unless the computer is currently performing the relevant functionality

**insecure**
Inappropriate code has been loaded

A trusted platform helps to reveal what the computer is doing, and helps ensure that identities and secrets can't be used until the computer is once doing what it is designed to do

# Services provided by Trusted Computing

Trusted Computing provides confidence in a product, especially if the product's behaviour isn't fully-secure or might become insecure

- Establish whether an individual product is the intended product, and whether it is doing what it is designed to do, even if that behaviour isn't secure or becomes insecure because of modification.
- Provide controlled access to keys and secrets that depends on the product's current behaviour

Trusted Computing allows a compromised product to be restored by installing and replacing software

- Otherwise, a trusted entity must confirm the product's proper operation before installing replacement secrets for communication and identification.

# Typical types of product that benefit from a Trusted Computing architecture

- Reprogrammable computers that provide services similar to locked-down computers
- Devices that execute instructions on general-purpose computing engines instead of being dedicated functions provided by bespoke hardware
- Products that are less-than-fully-secure and hence have unpredictable behaviour if attacked

# Doubts about Trusted Computing

- Some product manufacturers believe they don't need Trusted Computing, because they think they
  - can build complex secure commercial platforms and networks, or
  - can build something simpler or cheaper, or
  - need less protection than that offered by Trusted Computing
- Some commentators object to Trusted Computing because
  - it obstructs the repurposing of computing platforms
  - it can make it obvious that they've personalised their own platform (in ways that might make the platform less safe )

# How Trusted Computing works

# Trusted Computing is the simplest method of providing confidence in a platform or computer

Trusted Computing enables whoever has an unprotected copy of software or data to state which environment is used to protect that software or data

- This is the most liberal method of protection
  - It acknowledges that different data on the same platform might need different protection, or even no protection

Even secure computers can be improved by building them upon trusted platforms

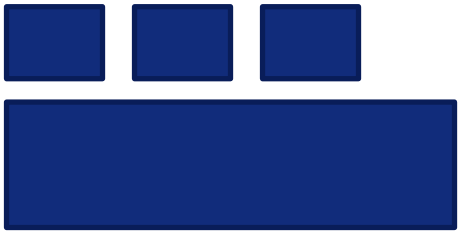
# Principles implemented by Trusted Computing

| | |
|---|---|
| Are you who I think you are? | Identity |
| Are you behaving normally? | Environment and circumstances |
| How do you normally behave? | Reputation |

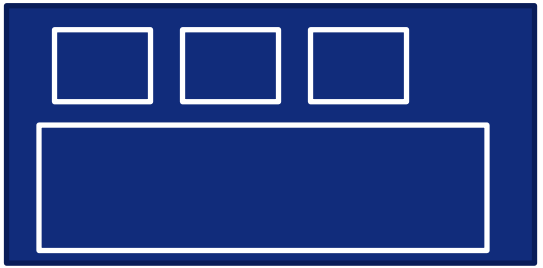
Trusted Computing is a technological implementation of methods we use in everyday life, when we decide whether to trust something

# Isolation

Any credible data protection mechanism must constrain who and what has access to programs and data – there's no known alternative



non-virtualised isolation



virtualised isolation

All computing platforms must ultimately rely upon isolation mechanisms to protect secrets and applications

- Ultimately, virtualised isolation relies upon physical isolation

# Data Isolation

Data isolation is essential

- If there is a risk to data when a platform is switched off
  - TCG-certified TPMs are well suited for this role because they have a robust degree of protection from tampering
- When platforms isolate processes that are intended to access particular sensitive data from processes that have no legitimate right of access
  - TCG-certified TPMs are well suited for this role because TPM processes have a robust degree of isolation from their host platform

# Security required by fixed-behaviour general-purpose computing engines

Not addressed by TCG

- Isolation for computing engines
- (often) a secure boot mechanism

Addressed by TCG

- Isolated secrets
- one cryptographic identity (key)
  - a new identity whenever the product‘s behaviour is modified
- more than one set of cryptographic algorithms

# Security required by flexible-behaviour general-purpose computing engines

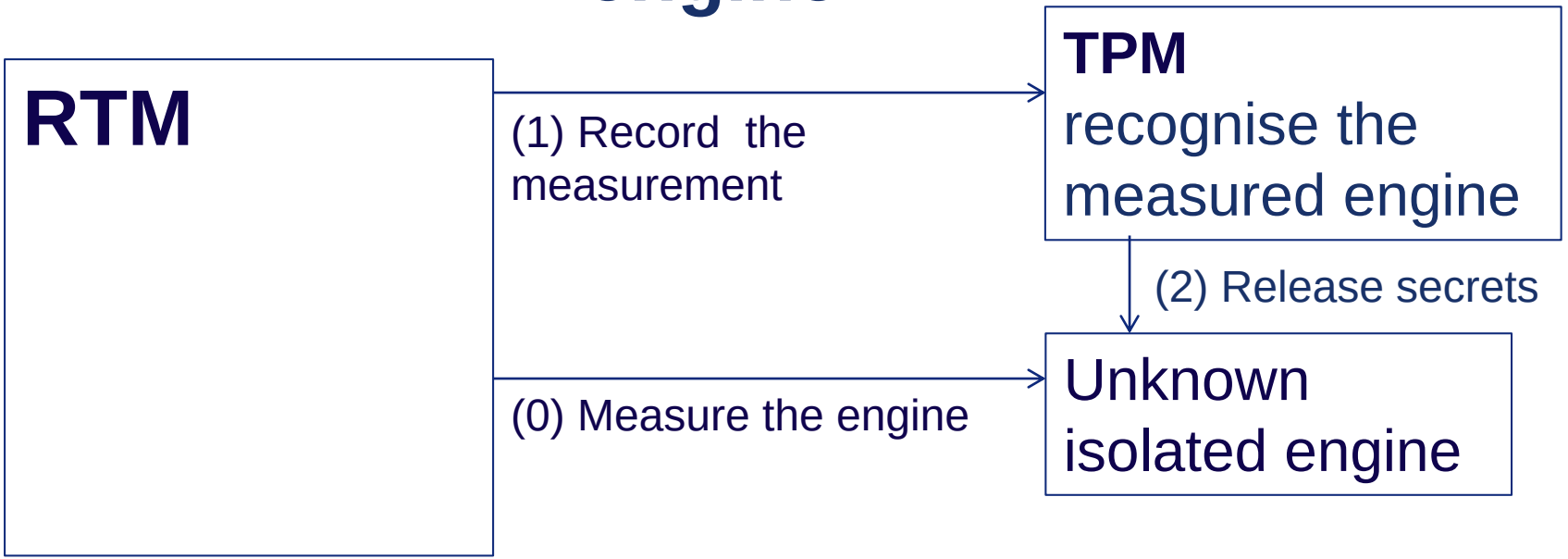
Not addressed by TCG

- Isolation for computing engines
- (often) a secure boot mechanism

Addressed by TCG

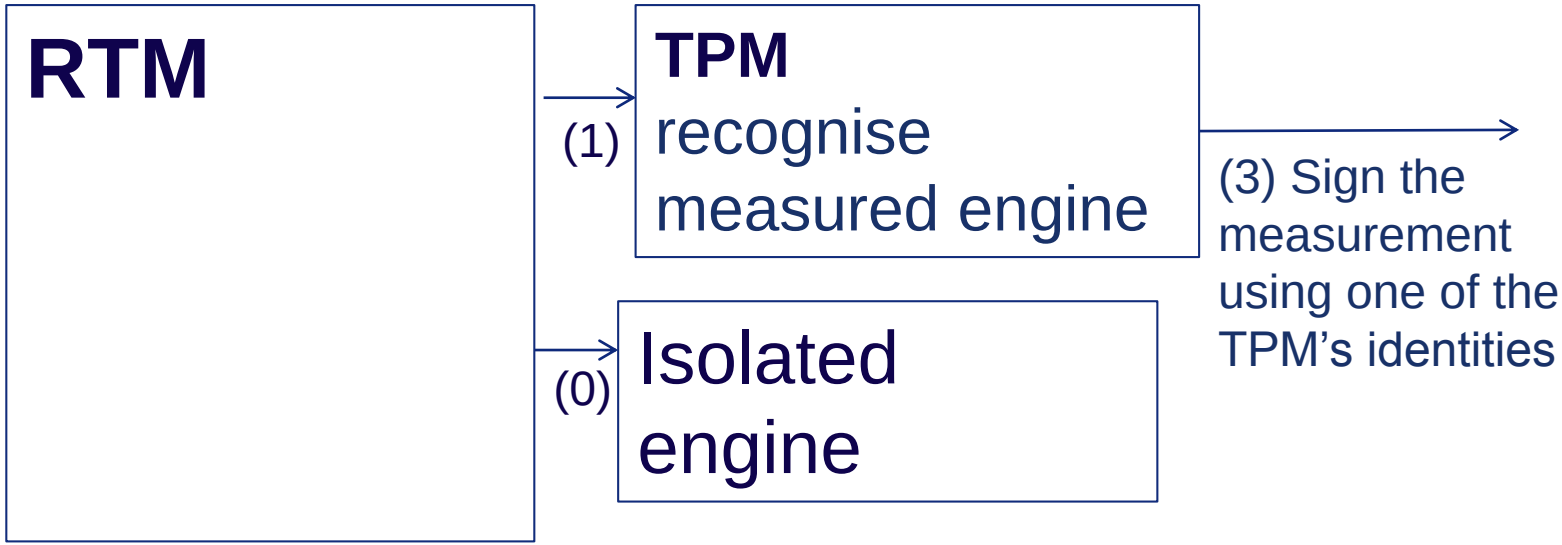
- Isolation for secrets and between secrets for different behaviours
- multiple cryptographic identities (keys)
  - a different identity for each product behaviour
- more than one set of cryptographic algorithms

# Isolating secrets belonging to an isolated engine




- The TPM recognises the engine by virtue of the measurements provided by the RTM
- If the TPM recognises an engine, it provides access to the engine's secrets that were protected by the TPM

# Reporting the identity and behaviour of an isolated engine




- The TPM signs the measurement of the isolated engine using one of the TPM's cryptographic identities
  - This identifies a platform and reports its current behaviour without needing to replace an identity when behaviour changes

# Trusted Platform Modules

# Trusted Platform Modules

- TPMs are what most people think of, if they think of Trusted Computing
- TPMs isolate secrets, and can be used to identify a product and reveal its behaviour
- TPMs eliminate the need for a new cryptographic identity whenever a product exhibits a new behaviour

# Business benefits provided by a TPM

- A foundation for a good commercial level of data protection and service protection
- Separate facilities to protect the platform's data and services, the user's local data and services, and privacy sensitive identities
- A simple commercial-cryptography import-and-export licence
- Low cost (because of economy of scale)

# Benefits of Trusted Platform Modules in the absence of an RTM

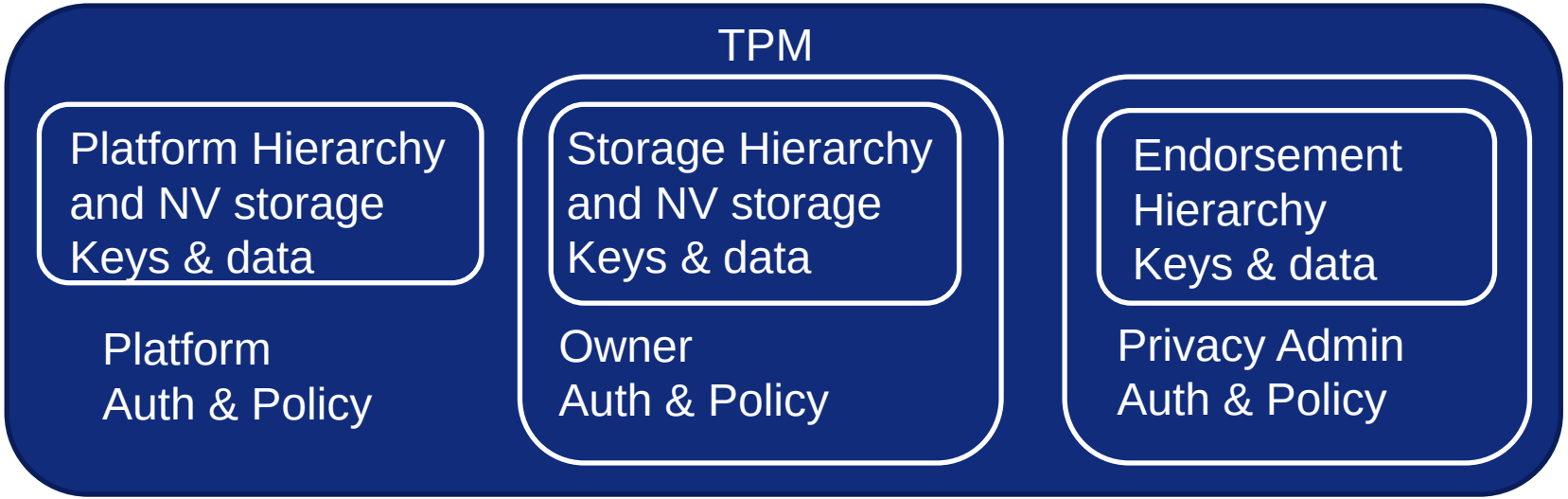
- TPMs are useful security modules irrespective of whether the product uses an RTM, because TPMs
  - initialise, protect, and manage keys and data
  - provide all common cryptographic functions and some unusual cryptographic functions, such as privacy-preserving identities
  - can use different types of cryptographic algorithm, to reduce the risk that a deployed product must be recalled when a defect is discovered in any single algorithm
  - provide very flexible multi-factor access controls for the functions and secrets they protect

# Technical services provided by a TPM

- Isolation *(can't protect keys and data unless they are isolated*)
  - Physical isolation (*protect data at rest, if nothing else*)
  - Cryptographic isolation (*protect an unrestricted amount of data*)
  - Access controls (*isolation is useless unless it has access controls*)
    - Passwords (*if no one is looking*)
    - HMAC signatures (*if someone is looking or can interfere with the channel*)
    - Enhanced Authorization Policies (*for versatility and flexibility*)
    - Protection against Dictionary Attack (*enables a DoS attack if we are not careful*)
- Platform identification (*will keys and data be protected?)*
- Platform privacy (*cryptography might pose a risk to privacy)*
- Trust services (*sealing and attestation services are unique to trusted platforms*)
- Secure updates (*can't continue to provide protection unless updates are done properly*)
- Pseudo multitasking (*because TPMs are actually single-tasking devices*)
- Clock and Timer and Boot Counter (*because time can be useful as an access control and for audit*)
- Cryptography (*TPMs provide a crypto API*)

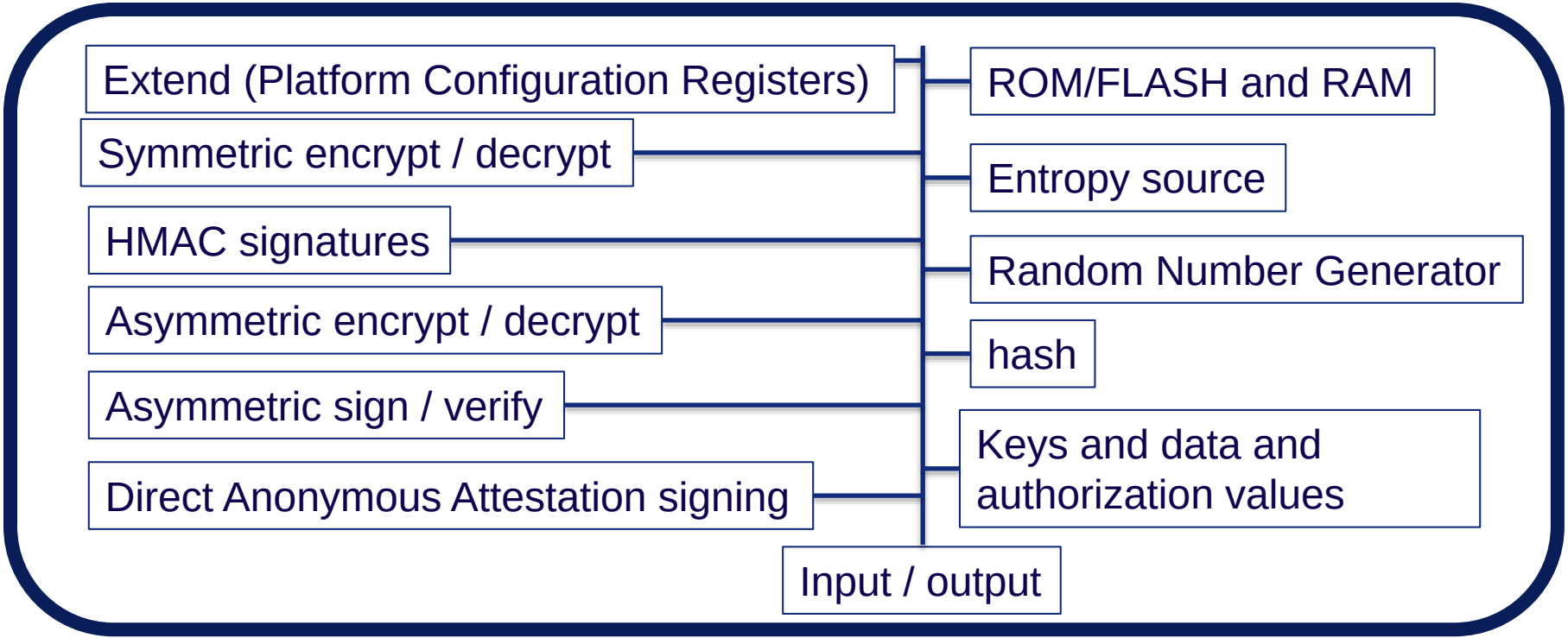
# What a TPM provides

A hierarchy of resources, each with individual access controls




Individual keys and data in hierarchies and NV storage have their own individual authorisation and policy values

# What a TPM uses




Isolation and tamper resistance

# How to use a TPM

1. Use TPM-driver software (efficient, enables all possible options, but requires significant TPM knowledge and expertise)
2. Create proprietary API software to manage the TPM and hide unwanted TPM complexity (elegant, provides all desired options, but requires significant TPM knowledge and expertise)
3. Use published API software that manages the TPM and hides TPM complexity (elegant, provides commonly used options, requires minimal knowledge and expertise)
   - TCG's "TPM Software Stack" (TSS) https://github.com/01org/TPM2.0-TSS
   - IBM's "TPM Software Stack" https://sourceforge.net/projects/ibmtpm20tss/
   - Microsoft's TSS https://github.com/Microsoft/TSS.MSR

# TPM specifications

- TCG library specifications define TPM architecture (what a TPM does and how it does it)
  - www.trustedcomputinggroup.org/resources/tpm_library_specification
  - www.trustedcomputinggroup.org/resources/errata_for_tpm_library_specification_20
- TCG platform-specific specifications define TPM functionality
  - Which commands are mandatory and which are optional?
  - Which algorithms are mandatory and which are optional?
  - How many keys must be concurrently supported?
  - How much persistent storage is required?
  - Which platform measurements must be recorded, and where?
  - and so on . . . .
  - (for example, see the PC-Client specifications www.trustedcomputinggroup.org/resources/pc_client_platform_tpm_profile_ptp_specification)

# Further reading

- An “open access” book intended to get one started with TPMs: “A Practical Guide to TPM 2.0 - Using the Trusted Platform Module in the New Age of Security”; Arthur, Challener
  - http://www.springer.com/us/book/9781430265832
- A reference book intended to help explain TPMs: “Trusted Computing Platforms - TPM2.0 in Context”; Proudler, Chen, Dalton; Springer
  - http://www.springer.com/us/book/9783319087436
- Software
  - https://sourceforge.net/projects/ibmswtpm2/
  - https://chromium.googlesource.com/chromiumos/third_party/tpm2/
  - https://github.com/vbendeb/tpm2_server
  - http://research.microsoft.com/en-US/downloads/35116857-e544-4003-8e7b-584182dc6833/default.aspx
  - https://github.com/PeterHuewe/linux-tpmdd/tree/tpm-emulator
  - https://github.com/PeterHuewe/linux-tpmdd/commit/9329f13c403daf1f4bd1e715d2ba0866e089fb1d
  - https://github.com/PeterHuewe/linux-tpmdd/commit/bbf2f7064c1452b47f11dfad340326b1205d863a