

## 2 Trustworthy Computing

The decision, if a utilized platform may be considered trustworthy, is not a trivial one. Even if the complete software configuration of a computing platform is known, there is still the uncertainty of inadvertent modification. A paramount example of such modifications are computer viruses and Trojan Horses, which may be detected by the sophisticated heuristic methods embodied in current virus scanners, but their detection is not guaranteed.

Trustworthiness itself is a rather vague expression and therefore the meaning of this term within the scope of this thesis is clarified in the following.

### Definition: Trustworthiness

Notion of the dependability of a specific entity to comply to a given policy within the scope of a relationship. A trustworthy entity is expected to behave honestly. For the application to computing systems, trustworthiness shall be a verifiable property of the system.

The plain definition of trustworthiness does not cover the semantic difference between *trustworthy* and *trusted*. To *trust* sth. is an action that defines relationships between entities and it does neither have to be complete nor mutual. If an entity is *trusted*, the decision whether the entity will act honestly within the scope of this particular relationship, has already been made. In contrast, the term *trustworthy* denotes to the estimation of the *worthiness* of an entity to be *trusted*. An entity is referred to be *trustworthy* if it behaves honestly according to a given policy. If an entity is *trustworthy* it may be also *trusted*, but if it is *trusted* it is required to be *trustworthy*. Within this thesis, the term *trustworthy* is favored, which is oppositional to the term Trusted Computing advocated by the TCG.

The *trustworthiness* of a device relies on the continuous monitoring of the *system integrity* and the devices' capability to proof its *authenticity*. Establishing trustworthiness in a device is only possible if the system state is

known and if it can be verified. The determination, if a system is trustworthy or not, can be made by a comparison to previously recorded known-good system states. Moreover, *trustworthiness* is a dynamic decision which in principle needs to be evaluated continuously.

During the *trustworthiness* evaluation of a given platform, the *system integrity* and the *authenticity* are assessed. Both fundamental properties of a *trustworthy* system may be regarded as binary (0/1, true/false) decisions. A signature is authentic or not and the integrity of a system is within the defined constraints or not. The decision on the *trustworthiness* fails with at least with one of these properties. For simplification, often only a subset of the systems properties are assessed for trustworthiness.

A failing *system integrity* check, for example, is indicating a malicious modification or a yet unrecorded update of the platform had taken place. *Authenticity* is provided by cryptographically signed messages that unveil modifications of the platform upon signature verification. In special cases, a finer granularity is favored to record the system integrity, allowing the user to trust the device only for certain tasks. An example for this is the usage specification of the Platform Configuration Registers (PCRs) in a TPM. Including a subset of the available PCRs within an assessment allows the user to select which components are crucial for a trustworthy system (cf. Section 2.2.3). This assessment may be recorded by utilizing a bit-vector representation of *system integrity* measurements. If and only if all of the selected values are matching the expectation, the application shall be able to access confidential data.

All components being critical to the systems security, namely hardware, software, and firmware, are often referred to as Trusted Computing Base (TCB). Due to the complexity, these components are not included in the TCB as a whole, but only relevant parts affecting the security policy. For the remainder it is assumed that vulnerabilities shall not provide the attacker with elevated privileges higher than the ones already given outside of the TCB.

Exhaustive security assessments by means of compliance to security policies or by program verification are only feasible for reduced complexity problems. Consequently, developers as well as research are striving to minimize the TCB [MPP<sup>+</sup>08, Par10]. Moreover, Heiser et al. are advocating solutions supporting formal verification in [HMK12]. Their verification of a microkernel forms the basis for building trustworthy systems.

A judgment on whether a device is considered to be trustworthy or not, can only be made if certain conditions are fulfilled. Current systems face

different threats, which have to be taken into account while compiling the list of requirements.

The basic component to realize Trusted Computing for commodity computers is the TPM. It is a hardware-based security module providing the functions to measure and attest the system state of the underlying platform. Although the TPM provides mechanisms to facilitate the decision about the trustworthiness of the platform, it is not able to deduct any notion of trustworthiness by itself. This kind of service may be provided, if the software part of the TCB is supporting the comparison to known-good values. A TPM is mainly providing a secure storage on the one hand and access to certain cryptographic primitives on the other. The cryptographic primitives include components providing signature generation, encryption, and protocol operations. The secure storage is utilized to protect cryptographic keys and to record a representation of the system state. Digital signatures are generated by applying the RSA. Cryptographic protocols always require random numbers to guarantee the freshness of every query or reply in order to prevent replay attacks. Therefore, the TPM is additionally providing a random number generator.

The remainder of this chapter will shed some light on Trusted Computing, especially on its capabilities and its limitations. A reasoning on the practical implications of trustworthiness will be given. Then, the measures provided by the TCG will be detailed, followed by a presentation of specific concepts for mobile application and virtualization scenarios. Additional commercial products providing similar or extended protection mechanisms are outlined afterwards. This chapter finally concludes with a compilation of existing approaches utilizing TPM(-like) technology for reconfigurable cyber-physical systems.

## 2.1 The Meaning of Trustworthiness

Trustworthiness cannot be derived from the knowledge of the current system configuration alone. An initial assessment is required to enable the decision which components need to be included in the configuration. The consideration if a system is trustworthy, is carried out based on these values. The comparison of the known-good values to reported measurements of current configurations enables the final decision on the trustworthiness of a given device.

In general, there are multiple entities involved in a Trusted Computing architecture, at least the system under assessment and a remote requester

questioning the trustworthiness of a given platform. The system is able to present its configuration to a remote requester exploiting the remote attestation protocol (cf. Section 2.2.7). With the knowledge of the system configuration, the remote requester decides if the system is operating within a certain set of valid configurations. The identity of the system is additionally revealed, as cryptographic digital signatures are used. The identities may be matched to different valid system configuration according to a given policy. Therefore, a given system state may represent a trustworthy configuration for one identity, but an untrustworthy for the other.

A practical premise for trustworthiness is the certification of utilized components. Meaningful certifications are carried out according to certain standards, such as Common Criteria for Information Technology Security Evaluation (CC) (cf. Section 3.4.2) or the National Institute of Standards and Technology (NIST) FIPS 140-2. The Federal Office for Information Security (BSI) identified this as a key requirement for trustworthy systems in [BSI07] and required a minimum certification of Evaluation Assurance Level (EAL)4+ for governmental use. Certified systems may be built exploiting a common security architecture, such as the TPM. Together with a TCB and its higher level services, effective measures for representing the system configuration are provided.

According to Parno, the TCB may be realized following two approaches, namely solutions providing the identity of code, such as Trusted Computing and solutions monitoring the behavior, such as information flow control [Par10]. The identity of code may be verified utilizing a TPM and to additionally certify the integrity of software. The other approach advocated by Parno, is to include certain functions during the development of the TCB, such as static code analysis or the use of hypervisors/security kernels for the separation of privileges. As already mentioned, the security-relevant code shall be assessed and therefore needs to be reduced to enable formal verification (cf. [HMK12]). A smaller TCB further simplifies the effort for assessing a systems configuration.

A trustworthy system shall fulfill the aforementioned requirements to support the assessment of the system configuration. Complying to this requirements, such systems may be considered trustworthy to process confidential information as they guarantee high security standards.

## 2.2 Trusted Computing Group

The Trusted Computing Group (TCG) is a non-profit organization developing and promoting industry standards of building blocks used for Trusted Computing. Members of the TCG are divided into three groups consisting of 11 promoters, 59 contributors, and 38 adopters<sup>1</sup>. Trusted Computing solutions are based on *data protection* mechanisms, *strong authentication* schemes, and measures to provide *network security*. The paramount contribution of the TCG is the specification on how to use a Root-of-Trust when building trustworthy systems. This Root-of-Trust is mostly implemented as a dedicated HSM separating the security-sensitive cryptographic key material and the security sensitive functions from the rest of the system.

Further, the TCG is organized in working groups, which focus on certain topics, ranging from PC-clients and servers to mobile systems and cloud solutions. These working groups define specifications reflecting the needs of the particular applications they focus on. The PC Client Work Group, the Storage Work Group, and the Mobile Platform Work Group are some examples.

A recent outcome of the Mobile Platform Work Group is an article relating to the growing threat of compromised Smartphones [TCG11a]. The updated mobile platform specification includes use cases of trustworthy mobile phones for strong authentication of enterprise employees and privacy sensitive e-Health applications.

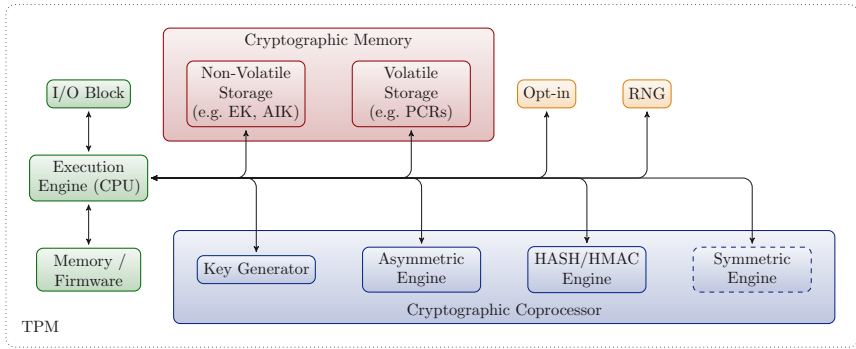
### 2.2.1 Trusted Computing Limitations

Trusted Computing only includes static program analysis and does not prevent against runtime attacks which may occur after a program has been loaded. Moreover, it is not mandatory to do statical program analysis to assess programs before they are included. Runtime vulnerabilities, such as buffer overflows, exploited by return oriented programming [CFK<sup>+</sup>09, CDD<sup>+</sup>10] and similar attacks, are not covered.

TPMs being used in cyber-physical systems have to face various threats, which have not been considered in the specification, e.g. physical attacks. These attacks range from reverse engineering attacks on TPMs, as presented by Tarnovsky in [Tar10], to other sophisticated techniques (e.g. side-channel analysis), as outlined in Section 4.4. Although, physical attacks have been explicitly excluded from the protection profile of the TPM, they may have practical relevance. The often required CC certification of EAL4+ (cf.

---

<sup>1</sup>[http://trustedcomputinggroup.org/about.tcg/tcg\\_members](http://trustedcomputinggroup.org/about.tcg/tcg_members) (2012-08-16)



**Figure 2.1:** Conventional TPM Architecture [TCG11c, p. 17]

Section 3.4.2) only provides protection against moderate attack potential, which does not include any of the aforementioned physical attacks.

## 2.2.2 Trusted Platform Module Specification

The TPM specification [TCG11c] is an extensive compilation of schemes to provide a variety of different features. These features range from data protection to anonymous authentication, to name a few. Each of these features is realized by successively executing a well-defined sequence of commands. The specification contains the descriptions for each of the more than one hundred commands available on current TPMs.

The TCG defines the minimum set of required features for a trusted platform as: *protected capabilities*, *integrity measurement*, and *integrity reporting*. *Protected capabilities* refers to the shielded execution environment and a protected cryptographic key storage. *Integrity measurement* refers to the process of cryptographically hashing the executed programs before execution. Whereas system *integrity reporting* is referring to the process of attesting the system state to an external requester.

The TPM is in principle a coprocessor realizing cryptographic services, similar to the services provided by SmartCards. Some manufacturers even share the security controller architecture between SmartCards and TPMs. The main difference between a TPM and a *SmartCard* is the ownership concept [Tom08]. A SmartCard is usually owned by the issuer, especially the keys and data on the card. In contrast, the user of a TPM is able to reset all cryptographic keys, destroying the identity of the TPM, while still being able to make use of it after reinitialization.

An overview of the conventional TPM architecture is given in Figure 2.1. A TPM is conceived as System-on-a-Chip, hence all security-sensitive services are executed within a closed system. This, together with a cryptographic key storage, allows for the realization of the *protected capabilities*, as documented in the TCG specification [TCG07]. The execution engine is a microcontroller which forms the basis of the TPM realization. This execution engine processes incoming commands and controls the cryptographic engines accordingly. It is also intended for the realization of cryptographic communication protocols.

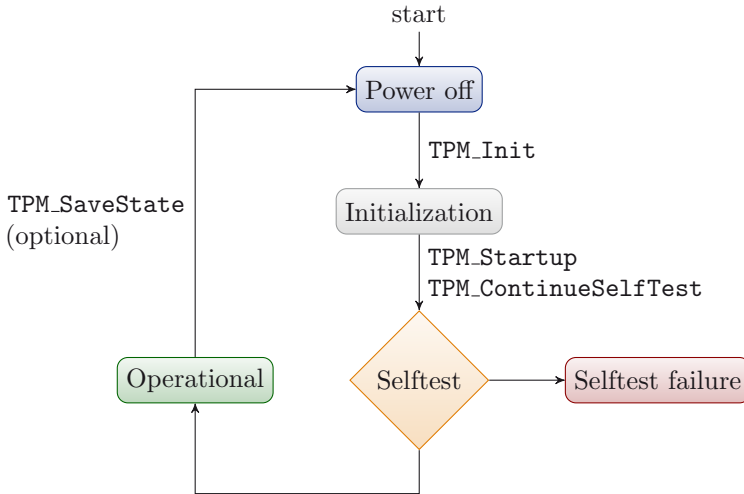
*Integrity measurement* is accomplished using the hash engine to obtain a digest of the characteristics representing the platform state. The hash engine facilitates the *integrity measurement* of the platform configuration and additionally supports the generation of Hash-based Message Authentication Codes (HMACs), which are essential for certain steps in the cryptographic protocols. Hash values at the beginning of a measured boot procedure (cf. Section 2.2.4) are generated by applying the Secure Hash Algorithm (SHA) engine.

Authentication is a fundamental component of *integrity reporting* and it is provided by an asymmetric engine. The *Key Generator* and the *Asymmetric Engine* are performing the RSA-based encryption / decryption and signature generation / verification schemes. The most important cryptographic engines of the cryptographic coprocessor are, RSA and SHA supplying means for authenticity verification and integrity reporting. The TPM additionally provides a secure storage for cryptographic key material and may include hardware-based accelerators for cryptographic services.

A common warrant for the freshness in cryptographic protocols is supplied by means of **NONCES** (Number used ONCE). A Random Number Generator (RNG) is provided with every TPM as the utilized protocols require a **NONCE** for the execution of commands. The RNG, supplied with a TPM, may be implemented in hardware or in software, as the TPM specification does not distinguish between those two variants. Additional external sources of entropy may be included to guarantee unpredictable behavior of the RNG, but the inclusion of these measures is decided by the manufacturer.

The operation modes of the TPM and further permanent configuration flags are stored in the Opt-In configuration, reflecting the state of the TPM. The mode of operation defines the subset of the commands available to the user. These modes may be set to disable certain commands, in particular handling the deletion of owner specific data.

The specification does not require any of the above mentioned algorithms to be implemented in hardware. However, it is practical to implement some as hard-wired modules to improve performance. A *Symmetric Engine* may



**Figure 2.2:** Initialization Procedure of a TPM

be provided as an optional component to accelerate symmetric encryption schemes.

The TCG differentiates between three Root-of-Trusts assembling the Trusted Building Block (TBB), namely the Core Root of Trust for Measurement (CRTM), the Root-of-Trust for Storage (RTS), and the Root-of-Trust for Reporting (RTR). In commodity computer systems the CRTM is realized as the initially executed part of the Basic Input Output System (BIOS). The CRTM represents the fundamental component for integrity measurements. The RTS and RTR are both implemented in the TPM itself and guarded by the Storage Root Key (SRK) and the Endorsment Key (EK), respectively. Securely storing the data being processed by the TPM is the purpose of the RTS. Reporting the system integrity information to an external requester is provided by the RTR (cf. Section 2.2.7).

The initialization procedure of current TPMs is visualized in Figure 2.2. It additionally details series of necessary commands for initialization to reach the operational state after a reboot.



## Trusted Platform Module Lifecycle

The Protection Profile (PP) for PC client specific TPMs [TCG11b] documents the activities related to a certain phase in the TPM life cycle. A simplified definition of the TPM life cycle is depicted in Figure 2.3. This life cycle covers all necessary steps to exploit the TPM functions. The first several phases of the life cycle concentrate on the generation of the EK key pair. There are in general three specific possibilities within the life cycle by whom the EK may be generated or downloaded, namely the manufacturer, the Original Equipment Manufacturer (OEM), or the enduser. The different impacts of these approaches are outlined in the following.

During manufacture, the EK key pair may be either downloaded to the TPM or generated by the TPM internally. This has the major advantage that the manufacturer is able to issue a EK credential certificate. A genuine TPM is then guaranteed by verification of this certificate. Additional conformance testing ensures the functional correctness of the device.

During the platform delivery phase, the OEM may also start the EK key pair generation process. Furthermore, the certification of this key pair supports authenticity checks in relation to the OEM. These first two possibilities of issuing an EK certificate are optional in the specification. However, these options provide the only measure to cryptographically protect the supply chain and effectively protect against substitution with malicious devices.

The EK and its corresponding certificate may also be generated at a later stage during the *take ownership* procedure. In the platform deployment phase, the owner of the platform initializes the TPM. In corporate environments, this initial phase is utilized to collect reference measurements of integrity values and adjust the TPM to comply with the company specific policies. Subsequently, the identity utilizing the TPM is initialized to support anonymous authentication during the platform identity registration phase.

During the platform operation phase most of the services of the TPM are enabled. The main purpose of platform authentication and data protection is carried out in this phase of the life cycle.

At the end-of-life of a TPM, issued certificates need to be revoked, data needs to be archived, and the ownership of the platform shall be cleared.

## Monotonic Counters

A monotonic counter is a useful primitive to prevent replay of data, without the requirement for a secure system clock. This type of counter is required

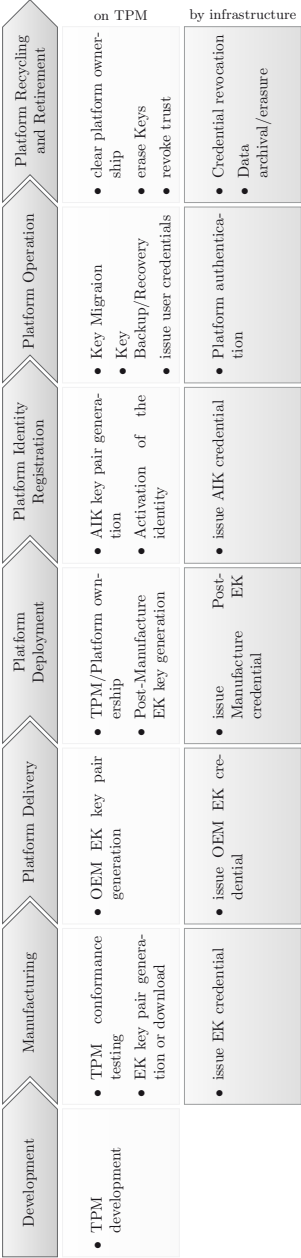


Figure 2.3: Simplified TPM Life Cycle [TCG11b]

Trustworthy Reconfigurable Systems  
Enhancing the Security Capabilities of Reconfigurable  
Hardware Architectures

Feller, Th.

2014, XX, 212 p. 35 illus., 15 illus. in color., Softcover

ISBN: 978-3-658-07004-5