Chapter 6
From Hardware Security Tokens to Trusted Computing and Trusted Systems

Apostolos P. Fournaris and Georgios Keramidas

Abstract As security attacks are becoming an everyday real-life scenario, security engineers must invent more intricate countermeasures to deal with them. Infusion of strong security to a computer system by recruiting specialized hardware tokens has already an established foothold in the modern information technology (IT) world. However, these tokens nowadays must be appropriately adapted to ensure not only strong security but also trust. Modern security specialists believe that the ultimate security goal is not only to provide a strong security shield but also to guarantee in an undeniable way that a system is trusted (the system always performs its intended functionality).

In this chapter, we elaborate on the IT world's transition from security to trust and describe the trusted computing approach to provide trusted systems. Current trends are presented and tools like trusted virtualization approaches are analyzed. The trusted computing technology features are discussed and our critical view on what the trusted computing future will be like is offered.

6.1 Introduction

Information technology (IT) applications whether on smart phones, tablets, desktop, notebooks, netbooks, PCs have penetrated in our everyday life and handle a large number of our daily data transactions including personal, business, social, or governmental type of communications. A great deal of sensitive data are involved in those transactions and if these data fall in the wrong hand they can cause critical security damage. In parallel to this evolution, security threats have also become smarter, leading into significant risks. Such threats include [1]:

A. P. Fournaris (✉)
Electrical and Computer Engineering Department, University of Patras, Greece.
KNOSSOSnet Research Group, Informatics and MM Department,
Technological Educational Institute of Patras, Greece
e-mail: apofour@ieee.org

G. Keramidas
Electrical and Computer Engineering Department, University of Patras, Patras, Greece
e-mail: keramidas@ece.upatras.gr

- Vulnerable programs (coding bugs, buffer overflows, parsing errors)
- Malicious programs (spyware, Trojans)
- Misconfigured programs (security features not turned on)
- Social engineering (phishing/pharming attacks)
- Physical theft (laptops, smart phones)
- Electronic eavesdropping (capturing email)

Most of above threads are completely transparent to the computer system user as well as other associated systems (e.g., operators). Every transaction with a computer system can potentially lead to unauthorized security data leakage. In other words, during operation, the computer system is considered untrusted. The associated risk with such system usage is not always affordable, especially when it comes to transactions involving sensitive data. In such cases, the user must be confident that his system can be trusted. As a result, trust is a very important and desirable feature of modern computer systems and considerable effort has been invested in the development of trusted systems.

In this chapter, we offer a brief overview of the approaches followed over time to protect a computing system from security threads, starting from simple tokens, e.g., magnetic cards, and moving toward the design of a fully trusted system. Since software can always be tampered and manipulated, the realization of a trusted system can only be achieved through dedicated hardware components that offer untampered, secure areas, an attacker cannot reach or violate. So, we base our analysis on how hardware structures can be utilized to provide trust. Moreover, the notion of trust is thoroughly defined and trust models that lead to trusted systems are discussed. Our focus is directed around well-established trusted system hardware approaches (Trusted Computing Group (TCG), specifications) and how they can enhance software tools like virtualization to achieve trust.

The rest of the chapter is structured as follows. In Sect. 6.2, a brief analysis of the security enhancement history through hardware means is discussed. In Sect. 6.3, the notion of trust is defined, various trust establishment models are discussed, and how those models can be used for trusted system establishment is analyzed. In Sect. 6.4, dominant hardware-oriented trusted computing methodologies (based on the trusted platform module (TPM)) are presented. In Sect. 6.5, virtualization technology and its combination with hardware trusted computing are discussed. Section 6.6 concludes the chapter and suggests future directions on trusted system development.

6.2 Hardware Security Modules

IT strong security can be achieved by enhancing existing applications or developing new ones with appropriate built-in secure software or hardware fences. However, the vulnerability of software solutions on malicious manipulation that can bypass software security as well as the slow response of software solutions to security requests have convinced IT security experts that hardware solutions are more appropriate for high security demanding applications like those in financial, military, or governmental environments. This belief led to the development of a wide range of special purpose hardware token devices that act as security arbiters and/or user authentication tools (validating a user's identity by providing a token that only the user possesses). These tokens are based on dedicated hardware processing units consisting of physically tamper-resistant embedded cryptographic processors that communicate with the conventional general purpose system processor in order to offer a predefined set of cryptographic and security services [2].

The first commercial uses of cryptographic processors or hardware security modules (HSM) were made for financial transactions. In such applications HSMs enforced a policy on key usage along with a variety of key protection measures. Electronic payment systems use the HSMs for secure communication between the banks and the customers and for secure storage of all authentication information. The customer is provided with a cheap autonomous HSM (smart card) along with a personal identification number (PIN) for authentication. This smart card solution guarantees end-to-end security in the communication between the bank and its clients.

The introduction of Internet banking brought new dynamics in the field of financial transactions since the customer has no physical presence in a prearranged place to use the bank services. To access bank services ubiquitously through internet banking, the user-customer needs to build and maintain a secure, trusted environment, irrespective of his physical location. Banks currently address this challenge by providing to their customers tamper-resistant authentication, i.e., authorization devices (e.g., the RSA SecurID) that can generate time-dependent or random passwords based on unique registered key in the device. However, the customer is not provided with any safeguards of validating his internet banking access device (laptop, desktop, tablet). Customers must apply their own additional security measures (e.g., firewall, antivirus, antimalware software) in order to trust their access devices while the bank itself always considers these devices untrusted.

HSMs are widely used in military–government applications. Military cryptographic processors have been used from the Cold War era for encrypting sensitive communications and for authorizing people as well as for protecting high-importance military operations. Some of these technologies have been replicated in crisis management situations where civil protection agencies (police, fire brigade, and ambulance staff) need to communicate over secure channels. Proprietary secure communication channels have been used in such scenarios (TETRA, Tetrapol, etc.[3]) that strongly rely on dedicated HSMs in an attempt to create a secure communication environment over an untrusted infrastructure (wireless links, telephone network, etc.).

6.3 Trust

While security might be the dominant term when it comes to protection of sensitive data, trust is a much stronger concept that goes beyond confidentiality, availability, integrity, and nonrepudiation (the basic security pillars). Trust tries to formulate a good-faith relationship between computing machines as well as between their users. The realization of trusting an entity B by an entity A is based on the belief that B will always behave honorably, reliably, and securely under a specific context [4]. From IT perspective, trust is not only about securing the communication channel or authenticating the data sender but also on trusting that the sent information are legitimate, they do not include malicious codes (e.g., malicious, virus, or trojan code) and they will not harm the receiver in an unforeseen way. In other words, trust extends to the sender itself (not just its messages) by believing that he will obey to specific communication rules (dictated by the communication protocol or policy) and will not abuse communication by nonresponsiveness or selfish behavior.

6.3.1 Trust Establishment Models

Establishing trust in computer systems must involve both the user and the computing device at hand. This can be achieved by using a "hard evidence" approach where (1) the device always complies with specific rules thus becoming predictable and (2) the device user behavior conforms to prescribed rules favoring a behavior pattern (or series of patterns) that can be considered legitimate. The series of device rules constitute a specific trust policy which can be also extended to a user behavior policy. To achieve this, appropriate monitoring mechanisms are required so as to validate the computer system (device and user) compliance to the trust policy. The system is considered trusted when the policy is followed and the user behaves in a trusted manner. Under this view, however, it has to be considered that the enforcement of the trust policy to all involved parties necessitates a trust mechanism that is hard to implement especially when users are involved. Thus, alternative approaches must be adopted, which are based on "trust reputation evidence." Reputation monitoring mechanisms are focused on assessing the trust level of an entity (the computer user and device) based on the entity's interaction and behavior history within a given context. More specifically, this history, constituting the entity's reputation, is built from the reports and observations collected by other entities (a single entity acts as evaluator or a group of entities act as trusted third parties).

In a trust model dictated by the above directives (i.e., policy and reputation), the communication channel between a sender and a receiver is always considered secure using strong security mechanisms as well as penetration-resistant means. Both sender and receiver gain value in trusting each other which exceed the performance cost of communication. Value is derived when the involved parties have a need for communication. For example, when the receiver needs to use the communicated information to invest on computer resources or when the sender is obliged to respond to a receiver data request. The level of trust that an involved entity A has to another entity B has a communication cost which is translated into the risk generated when entity B behaves in a malicious, unforeseen way. High trust benefit (i.e., high value) comes when entity A trusts B, meaning that the cost associated with communication with B is smaller than the value of the communication outcome (e.g., the message transmission) [5].

To minimize the risk associated with trust relationships, each involved entity can adopt an isolation mechanism to increase the communication value. Isolation is based on the adoption of trust verification mechanisms during communication.

Instead of blindly trusting the sender, a receiver can follow approaches based on policy and/or reputation in order to protect it from malicious or incompetent senders. The approaches extend not only to message verification but also on noncompliance with a communication protocol or nonresponsiveness of protocol participants. As can be derived from the previous analysis, the trust verification and isolation mechanisms can be categorized in policy-based trust approaches and reputation-based trust approaches.

Policy-based trust establishment is focused on the enforcement of a specific policy capable of guaranteeing trust relationships between communicating participants. Strong isolation and high value for each such participant is maintained by collection, management, and verification of policy-related credentials. Such credentials are meant to be provided and verified by a trusted third party (TTP) authority. Collecting enough credentials by a receiver entity constitutes a proof that the sender can be trusted, thus the receiver no longer remains isolated from this sender. Gathered trust information about an entity can be considered such credentials. Policy-based trust establishment includes trust negotiation protocols with requests for trust credentials, TTP credential verification, and generation of trust assurances. During the execution of a negotiation protocol, an entity usually must provide private content information. Sensitive trust-associated data must be handled appropriately so as to protect the entity computer system's and user's privacy. So it is obvious that trust has a tight interdependence with strong security (although it serves different purposes) since trust negotiation as well as credential handling follows security-related approaches based on cryptographic primitives and well-known security protocols. However, trust policies have a well-specified "language" that is associated with the employed trust credentials. Trust relationship representation can have many forms depending on the trust standard that is adopted. The representation language involves software structures, like Web services [4], but requires a hardware base using HSM (trust anchor), acting as trust policy arbiter on the associated computer [6].

In the *reputation-based trust establishment*, the trust experience of a community of other entities is used in order to make trust decisions about a single entity. In this approach, when a receiver wants to know if he can trust a sender, then he "asks" the opinion of a third party to attest the sender's trust level. Based on the collected replies, the receiver infers if the sender is trusted or not [4, 5]. If the third party is a single trusted authority, then the system is decomposed into a policy-based one. Instead of obtaining trustworthiness-related information from a centralized trusted third party, the reputation-based model collected knowledge of many entities that have prior experience with the evaluated entity. The reliability of the above approach is maintained by a trust reputation-recommendation system. Such system relies on the opinion of a series of recommenders and evaluators of trust that are collecting and analyzing behavior patterns of involved entities. The larger the numbers of recommenders, the more reliable are the trust decisions that can be made. Of course, the recommenders themselves must also be trusted if their opinion is to be of any real value.

Reputation systems usually work in decentralized manner and can be applied to networks that favor data collection and distribution such as the IP networks (the

Internet), p2p networks, and grids [4]. Reputation collection is achieved using software tools (agents) roaming the network, collecting trust reputation data from network entities or other agents. Delivering such information to a requesting entity can lead to an appropriate trust decision. In this notion, reputation can be defined as a measure of trust that each entity maintains and shares with other entities thus forming a "Web Of Trust" (WOT). Transferring trust in a WOT approach is based on trust metrics and trust transitivity rules. Entities that have achieved a level of trust have a WOT link thus forming WOT graphs while entities that have no collected trust reputation data follow trust transitivity rules to form a link. Such rules can be expressed in a simplified matter as "if A entity trusts B entity and B entity trusts C entity then A can trust C." Thus traversing the created WOT trust graphs and collecting reputation data (through the trust metrics system) the various agents of WOT can make trust decisions on a system entities. It must be noted that reputation-based model is a fully software solution (a software service) located at the application stack of a computing system (on top of operating system (OS) kernels and hardware structures).

In general, the reputation-based model is primarily used in network security as a means of providing trust to nodes and network resources (network trust) while the policy-based model can more directly be applied to each computer system individually (computer system trust). The focus of this chapter is on ways of providing computer system trust.

6.3.2 Trusted Systems

By defining the notion of trust and formulating models to achieve it, a toolbox to build trusted systems is provided to computer security experts. A system can be considered trusted when its functionality is fully predictable or in other words when it always works the way it was designed to work. Breaking this rule will result into critical security problems, since trusted systems are relied upon serious security actions. Therefore, a trusted system is a system that can be trusted not to fail (if it fails the whole security policy collapses). Note that this should be discriminated from trustworthy systems which are systems that cannot fail (it is impossible to fail).

Since trustworthy systems are very difficult (if not impossible) to design, trust and not trustworthiness, is extended to a wide variety of systems covering even nontraditional security (military, business) applications. However, such systems consist of components that are not considered secure (nonsecure hard disk storage, networks, OSs, legacy components, etc.). Replacing those components with secure ones or redesigning them from scratch in an effort to guarantee trust is a very costly operation and is not usually followed. Affordable trusted system security can be achieved by following the policy-based model where trust metrics and rules are managed for all system's components and trusted applications, services, and resources are strongly isolated from untrusted ones. Similar to isolation from external environment mentioned in the previous subsection, computer system component isolation involves hardware/software codesign in an effort to create a trusted section

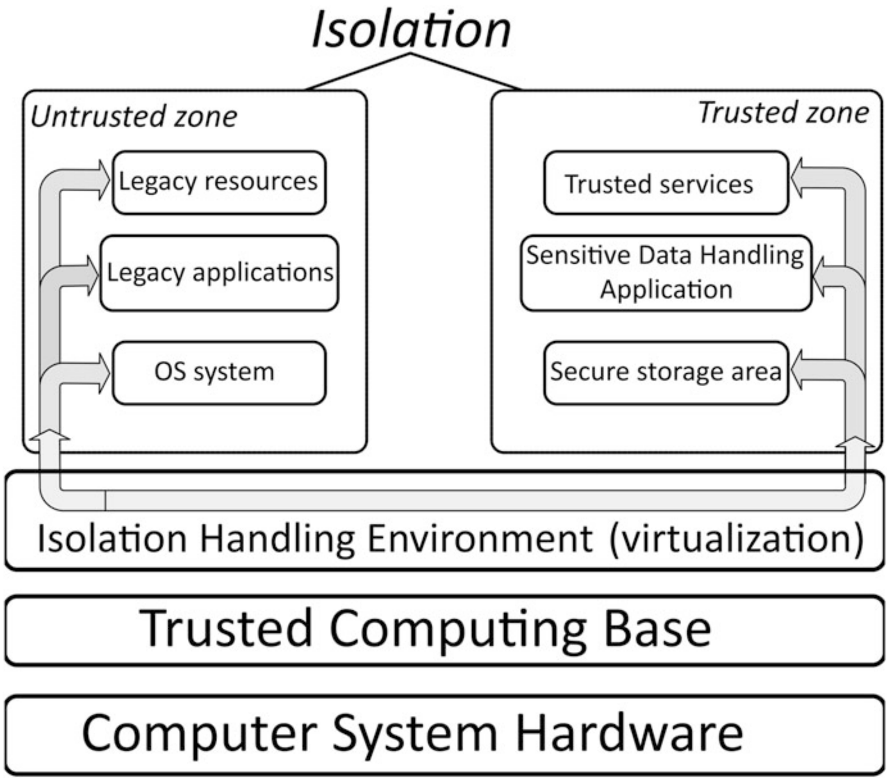

Fig. 6.1 Isolation in trusted systems

within the computer system that is fully reliable for critical applications. Trusted and nontrusted applications are strongly separated between the trusted and untrusted system's zones while their data exchange follows strict security rules (based on the system's policy). This approach can be viewed as an extension of the HSM concept where an HSM is attached externally to a computer system in an effort to provide strong security. In a trusted system, the HSM becomes an integral part of the system's hardware and low-level software, and is assigned the additional role of managing the system's isolation. A common practice on how to achieve this is described in Fig. 6.1.

As described in Fig. 6.1, the system's hardware does not communicate directly with the OS kernel but is rather managed by a trusted computing base (TCB). TCB is a collection of hardware, firmware, and/or software components critical for the reliable functionality of a computer system [7]. As the term suggests, TCB must be trusted since it has the highest OS privilege level and is responsible for the system's security policy enforcement. It includes security validation as well as domain separation mechanisms so as to fully control the information flow of the computer system, provide access control and resource management. The TCB assets and its sensitive data and services must be protected even from itself. To achieve this, the

TCB has special hardware and software protection mechanisms such as protected storage, protected memory, special memory managements units (MMU), TCB software security checkers, and validators in an effort to guarantee trust. On the other hand, for a practical implementation, the TCB must be small enough in order not to burden the system's functionality, to be manageable, and to be efficiently checked periodically from certification organizations, such as Common Criteria [8] for trust (formal security verification). It must be noted, however, that TCBs are considered "trusted" and not fully trustworthy. Security bugs related with the OS kernel (which in some cases is part of the TCB) cannot always be detected thus leading to TCB security risks compromising its trustworthiness.

The isolation between the trusted and untrusted zone is handled by a specialized level consisting of a collection of software tools that use the TCB services for splitting OS resources, services, and applications into trusted and untrusted ones and of controlling/managing all transactions between the two zones. This control is vital for the computer system trust. Trusted subjects (application services) and objects (resources) of the computer system are securely handled in the system's trusted zone and calls or data exchanges with untrusted subjects and objects are not done directly. All such activities must be evaluated and validated by the isolation handling environment acting as a security arbiter between zones. Virtualization is a widely used technology for such isolation handling. Each zone is executed in a different virtual machine environment in order to remain totally autonomous. Alternatively, microkernels [9] are used to achieve the same goal.

A similar solution is offered by ARM in its latest processors (ARMTrustZone technology). ARMTrustZone [10, 11] offers a trust zone environment within the processor structure and not the computer system in general, by splitting processor functionality between a secure and nonsecure operation section, thus making possible the realization of a secure and trusted anchor within its hardware core. So, a designer is able to use the secure part of the processor for implementing trusted services. However, when using the ARMTrustZone or software-based isolation handling environment, it is hard to determine whether such implementations provide fully shielded-secure locations or protected capabilities.

The above approaches are focused on the computer system side and not on the user side. They follow the policy-based trust model but do not involve the user's behavior. This fact can be a security risk for the system as a whole since a malicious insider user can still compromise the system by deactivating trusted services or hacking the software arbitrating the trust enforcement.

6.4 Trusted Computing

In order to cope with the growing need for highly reliable trust enforcement that involves both the computer system and its user, Trusted Computing Platform Alliance (TCPA), an industry alliance was formed in October 1999 in an effort to develop and standardize trusted platform technology. TCPA and its successor, the TCG consortium (established in 2003), are responsible for formalizing, applying, and extending the trusted computing ideas to established computer systems either by introducing new hardware and software modules or by proposing appropriate protocols and directives. Trusted Computing can be viewed as a collection of technologies capable of constantly monitoring the behavior of a given computer system and its user for uncovering a possible compromise and an unexpected behavior. The TCG's view on trusted computing focuses on protecting the system from malicious entities even from its own user. As a result of its work, the TCG consortium has devised a series of specifications for new structures within a computer system that can provide reliable trust establishment. TCG goal is to enforce trust on a system by prohibiting the execution of malicious code, by protecting sensitive data (mainly private keys), and by attesting the system's trust level to other entities. This is achieved by a constant evaluation of the computer system's security from boot time. Since software-based security validation is not fully protected from malicious code injection and hacking attacks, TCG solution is based on hardware protection mechanisms along with software tools to establish trust. Embodiment of this approach is the specification of an HSM structure denoted as Trusted Platform Module (TPM) that is capable of acting as trust anchor within a computer system [1, 12].

6.4.1 Trusted Platform Module

The TPM is a smart-card like HSM chip bound to the computer system (usually soldered on the system motherboard). It acts as a hardware trust anchor (extending the trust zone concept) in order to enforce a trust policy by providing secure storage, public key authentication functionality, integrity measurements for all computer system resources/services as well as trust attestation. TPM functionality plays a key role in at least 16 special purpose platform configuration registers (PCRs) designed to store trust configuration measurements associated with specific computer system resources. PCRs cannot be directly written, they can only store an extended version of their previous value (a hashing of their previous value). Collecting a new trust-related measurement about a specific computer resource (e.g., service, private key) associated with a specific PCR leads to an update of the PCR value (extend operation) based on Eq. 6.1, where $H^{(SHA-1)}()$ is the SHA-1 Hash function and $(+)$ refers to concatenation.

$$PCR \leftarrow H^{(SHA\text{-}1)}(PCR + trust\ \ measurement) \tag{6.1}$$

The extend operation inherits the benefits of hash functions. There are no two identical PCR values coming from different measurements. The operation also provides indication of the order in which measurements were taken and hashed (a hash chain approach). The number of measurements is unlimited since the outcome will always be of the same bit-length. Currently, the TPM 1.2 version uses SHA-1 function and the PCR length is 160 bits.

The TPM is characterized and identified by a set of asymmetric cryptography keys denoted as endorsement key (EK) along with a certificate from the manufacturer.

This set of keys are securely stored in the TPM during manufacturing, they are unique for each TPM chip and are very restrictively accessed (private key cannot be released outside the TPM chip, it is nonmigratable). The EK is associated with three credentials provided usually by the TPM manufacturer and a third-party testing laboratory attesting that the current TPM conforms to TCG specifications, that the TPM is genuine and that the host system is an instantiation of a TPM equipped platform. The EK is used when a user wants to prove TPM generated keys have been produced by a genuine TPM. Therefore, the EK is utilized for certificate decryption of other TPM-generated keys, and this can only take place upon the request of the owner of the TPM cooperating with a certificate authority (CA).

The TPM generates a variety of different asymmetric and symmetric cryptography keys in order to realize its various functions. Such keys are Attestation Integrity Keys (AIK), the storage root key (SRK), and consecutive storage keys stemming from the SRK. All such keys can be either migratable or nonmigratable. Migratable keys can be moved to other TPM chips if there is some problem with a host TPM. Nonmigratable keys cannot be moved to another TPM chip. In TPM 1.2 version, all asymmetric cryptography keys are 2048 bit RSA keys.

Communication with the TPM is typically handled by the TCG device driver library (TDDL) whose interface is described in the TCG software stack (TSS) specification [13]. This software library installed on the TPM host computer system communicates with a device driver inside the TPM kernel. The device driver is responsible for the actual transaction with the TPM hardware device and its functions. The TSS library has all the tools (services, commands) for an application architect to use the TPM for adding strong security features to a software application. TSS functionality can be divided into three logical components: the TDDL, the TCG core service (TCS), and the TCG service provider (TSP). The TDDL provides an API for interfacing the TPM functions. The TCS, being the main user of the TDDL, manages the TPM resources, converts API request to TPM byte streams in order to be recognized by the hardware, and provides system-level key storage (outsize the TPM) while synchronizing application-specific calls coming from the TSP. The TSP is the interface with which applications communicate to the TPM. It offers access to the TPM services transparently for the application, acting as a shared object or a dynamic linked library (dll).

In its current version (TPM 1.2), the TPM chip is equipped with all the necessary hardware components in order to support strong security features. Apart from the I/O interface, necessary for the TPM communication with the external world, inside the TPM there are a series of cryptographic hardware components including a true random generator unit, an asymmetric key cryptography digital signature and authentication-authorization unit, and a hash function unit. Currently, the TPM adopts the RSA algorithm with 2048 bit keys and 160 bit hashing through SHA-1 and HMAC algorithm. The TPM also supports the secure storage of sensitive values (such as asymmetric and symmetric cryptography keys or measurement states) in special storage elements. Those elements include nonvolatile and secure volatile memory as well as a series of at least 16 PCRs [12].

6.4.2 Trusted Computing Functions and Services

TCG has specified a number of innovative ideas capable of setting up a fully controllable and trusted computer system environment for the system's user. TCG also describes a mechanism of providing evidence of trust to third parties as well as authentication/authorization. In this section, the basic TPM-TSS functions for achieving the above concepts are described.

6.4.2.1 Authenticated—Secure Boot

Through the TCG TPM mechanisms, the trust state of a system can be reliably measured and recorded. To achieve that, every part of the computer system from hardware level to application level is measured from boot time. This authenticated boot sequence measurement provides the guarantee that the system is not compromised and can be trusted. The TPM PCRs play an important role in the above boot sequence. The outcome of each system's component measurement is stored in a PCR register, according to Eq. 6.1. A recording of this process is stored in a history file denoted as Stored Measurement Log (SML), which is maintained outside the TPM structure.

The authenticated boot measurement follows a daisy chain approach, meaning that each component is measured and compared with existing known good value on the SML. In that way, measurement integrity is retained and the boot sequence can be trusted, meaning that no system component has been tampered. For this approach to work correctly, the boot sequence control must be given to a trusted source. Thus at computer system power-on, control is given to a TPM small root of trust module (a subset of the BIOS) before loading the BIOS. The root of trust takes the first measurement (BIOS measurement), compresses it (creating a digest), and compares it with the stored value inside the history file (SML). If the two values match, then the BIOS component is considered trusted and is executed, a specific PCR (PCR_0) value is extended with the collected measurement and the next component is evaluated in a similar way. The history file's integrity can be guaranteed by comparing it with the PCR values (that cannot be tampered). An overview of the above procedure, denoted as TPM static root of trust, is presented in Fig. 6.2. Note, that every component of the computer system is measured and evaluated in the above fashion including all executable code (i.e., system applications).

Static root of trust has some drawbacks associated with the need for chain trust measurement of every computer system software structure (inclusivity problem). The size of the executable code to be checked can be overwhelming and in a complex system can render its security properties unverifiable (no scalability). Executing a software code using a configuration file or processing data that has not been already measured can break the trust chain and harm the system's trust especially if such code is tampered at the time interval between trust measurements (scalability and measurement time problem). For this reason, static root of trust is used in contained, well-structured computer systems with well-defined operations. Instead, in complex systems requiring hundreds or thousands of measurements in a static root of trust chain, the dynamic root of trust chain approach is used which can dramatically

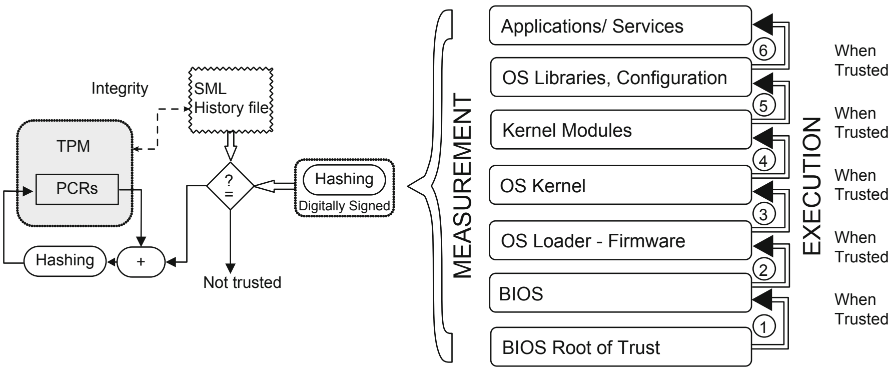

Fig. 6.2 Trusted platform module (TPM) static chain of trust overview

reduced measurement number. Dynamic root of trust (DRT) uses a sophisticated measurement mechanism in order to evaluate the computer system trust level that can be initiated at any point in time and can be repeated as often as necessary. DRT measurement is initiated by a specialized processor instruction creating a secure, attested execution environment [1, 14]. This environment is completely isolated from the rest of the system (e.g., Direct Memory Access is inhibited, interrupts, and virtual memory are disabled) and guarantees untampered execution of a secure loader that initiates the trust measurement process of any executed code. Each processor manufacturer uses a different approach on how to implement the DRT measurement instructions [15, 16].

6.4.2.2 Secure Storage

The TPM chip is capable of storing a wide variety of information in a secure way. Such information can be divided into asymmetric cryptography keys and symmetric cryptography keys or data. Secure storage follows a protected object hierarchy where higher-level keys are used for signing-protecting lower-level keys within this hierarchy. The root of the key hierarchy is the SRK which is an asymmetric key pair (2048 bit keys RSA keys) generated using the EK at the first TPM power-on. The private key of SRK must never leave the TPM and is therefore nonmigratable. Then, the SRK can be used along with the TPM random number generator in order to create storage keys for each piece of information needed to be securely stored. The storage keys are located into the key hierarchy and are protected by higher level keys. In addition, the TPM key generation mechanism can provide keys that are used only by the TPM that generated them (applies to private keys) and/or when the TPM's host platform is in a specified state. Securely storing keys or data is done using asymmetric key cryptography for confidentiality-integrity and can be associated with a 160-bit string of data in order to provide authentication before encryption.

The TPM can be used in order to provide secure sealing functionality. The encrypted information can be sealed so that they can only be decrypted by the TPM used for their encryption and only when the TPM Host platform is in a specified state. Practically, this can be done by performing asymmetric cryptography encryption/decryption using the SRK (that is unique for each TPM chip) and associating the encryption/decryption with specified values of some PCR.

6.4.2.3 Platform Remote Attestation

A very important TPM feature is the ability to provide TPM host system trust attestation reports to external third parties thus proving that the system can be trusted. The attestation operation should be unique for each TPM host computer system and undeniable. The TPM can be uniquely identified using the EK values. However, the need of protecting the TPM identity and its host's anonymity dictates that this approach cannot be used in practice. When the TPM host, denoted as Trusted Platform (TP), must provide attestation of its trust level to a third party, its anonymity must be retained so that its activities cannot be tracked. For this reason, the TPM generates a series of pseudonyms in the form of asymmetric cryptography key pairs, denoted as attestation identity keys (AIK), in association with the host system and a Certificate Authority (privacy certificate authority (PCA)) that provides AIK credentials. To achieve that, the TP provides the EK credentials and AIKs to the PCA, the PCA verifies that these credentials are legitimate thus the TP is genuine, and then generates an AIK credential by digitally signing (binding) the AIK public key with the description of the TP [1, 14]. In this process, the identity of the TP and the TPM (i.e., the EK) should not be revealed by the PCA. For this reason, the TCG has specified the Direct Anonymous Attestation protocol using provable security features and zero knowledge protocol cryptographic approaches to retain privacy [13, 17].

The attestation process, denoted as remote attestation, involves a specific AIK key pair, a TPM-specific state (denoted TPM quote) that provides a captured instant of the PCRs values and a series of nonce numbers. When an entity wants to have insurances about the trust level of the TP, it sends a request to the TP along with a nonce. In return the TPM sends back a digitally signed by the AIK private key, concatenation of the nonce and the TPM quote along with the appropriate AIK credential and a section of the SML. The requesting entity verifies the AIK credentials and digital signature as well as the nonce value that it originally sent to the TP and acquires the TPM quote. It then compares the TPM quote and the metrics provided from the SML with trusted known good values stored in trusted third-party database and if the values match then the TP is considered trusted.

6.4.2.4 Trusted System Realization with Trusted Computing Group Specifications

The trusted system concept, as described in Sect. 6.3.2, applies to the TCG specifications about trust. The hardware TPM structure provides an untampered, secure environment for supporting a TCB. Such TCB can be measured for trust and remain trusted at all times, thus, providing the basis for isolation. The TPM and TSS not only provide the means for evaluating trust for a computer system but also indirectly for the system user. They can guard the TP even from its own user since they do not allow malicious behavior (code injection, illegitimate software modifications) through the "secure boot" feature. The TCG approach can effectively enforce a TCG specified policy on the computer system and its user by hardware means and through measurement provide, in a way, a "reputation" collection system stored in the PCRs and SML. This system collects trust "reputation" locally about the various system components. This local reputation can be transmitted to other entities through remote attestation. This function requires that a TPM TP's "reputation" data (TP trust measurements) is being stored in trusted third parties within a computer network (e.g., using Internet) thus structuring and retaining "reputation" databases. This approach can be viewed as a form of static reputation collection mechanism following the reputation-based trust model. The TCG has specified ways of enhancing this direction through the TCG network connect (TNC) scheme which is beyond the scope of this chapter. Furthermore, several researchers have proposed ways of using TPMs for securing mobile agents roaming a network for data collection that also can be used for reputation collection [18–22].

TCG tries to provide a trust establishment solution that uses concepts from both the policy-based trust model and the reputation-based model, infusing them to an HSM (the TPM) characterizing the computer system (since it is soldiered to it) as well as creating all the HSM necessary supported software.

The TCG solution has guided the implementation by top hardware vendors (currently Intel and AMD only) of an isolated execution environment (IEE) (through processor instruction) to be used originally for DRT measurement. This environment offers isolation at hardware level and is inline with Fig. 6.1 concept. Setting up one independent IEE for each application or group of applications can provide strong isolation since the TPM and TSS can guarantee trusted communication between environments and the external world.

6.5 Virtualization Environment for Trust

Virtualization is a technology with high potentials in securing the computing world and providing trust. It provides an abstraction of one computer system level to another higher level. Virtualization can be found in many forms from network systems to storage or process virtualization. However, from security perspective, it is especially interesting to view these technology actors as reference monitor mediators of access to system resources and communication between abstraction environments within the system. These types of actors are referred as virtual machine monitors (VMM) or hypervisors while the abstraction environments are denoted as virtual machines (VM). A Hypervisor can be a small software code, positioned between the hardware and the OS kernel computer system levels in a similar way as the Isolation Handling Environment presented in Fig. 6.1. By introducing a VM environment where not only

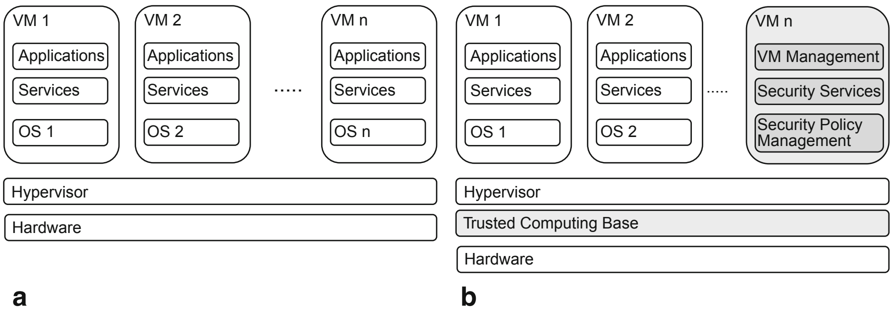

Fig. 6.3 Computer system architecture featuring hypervisors

programs but also OSs can run as if executed on hardware, hypervisors can achieve strong isolation. Thus, logical or network attacks on an OS application executed on top of a VM can be contained there and won't spread to the rest of the system. Further security advantages of hypervisor systems can be the ability to cryptographically protect data under a contained environment (cryptographic separation), the running of workloads at different time periods (temporal separation), and the assignment of specified hardware resources to different VMs (physical separation) [23]. Figure 6.3 describes generic approaches on security-oriented hypervisor-based systems.

Hypervisors can be used in cooperation with the TPM chip in order to take advantage of TCG trust functionality. Virtualization gives further dynamic to TPM trusted systems since it sponsors strong isolation beyond the TPM. On the other hand, TPM can be used to provide a secure hardware virtualization interface, practically forming a virtualization enabled TCB. However, the TPM cannot directly be used from VMs since it was designed for a single host platform. Only one such platform can have access to the TPM hardware structures and especially the PCRs. In a virtualization scenario, each VM can act as an independent platform and can potentially change the PCR of the single TPM attached to the host system. In that case, there is a serious security danger since VM trust states assigned to a PCR can be changed by a different VM. Several researchers have suggested solutions to this problem by adopting the concept of a software virtual TPM (vTPM) instance residing in each VM (as depicted in Fig. 6.4a) and communicating with the TPM structure in a manageable way through the hypervisor. Terra [24] was one of the first systems to introduce trusted computing to hypervisors (although without the use of TPMs) and has been followed by several other works [25–27] that employ the vTPM concept. Another approach introduced in sHype hypervisor [28] is presented in Fig. 6.4b. In this approach the vTPM is running on a dedicated VM and communicates through the hypervisor with a hardware TPM. The rest of the VMs have only vTPM drivers in order to communicate with the hypervisor trust interface. sHype also supports a dedicated VM for the system's security policy management and an access control mechanism inside the hypervisor [23].

Other approaches related to virtualization include the employment of microkernels such as the L4 [29] or seL4 [30, 31] or microvisors such as OKL4 [32] instead of

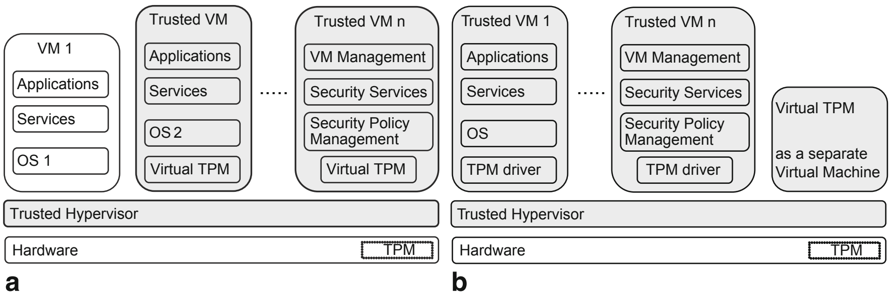

Fig. 6.4 Trusted platform module (TPM)-based hypervisor systems

hypervisors for achieving isolation. In such cases, the microkernel/microvisor must be an extension of the TCB in order to be trusted and executes security critical operations.

Virtualization has also been introduced in processor architecture level, as briefly mentioned in the previous section (DRT measurement), by extending the processor instruction set with special virtualization instructions in Intel or AMD processors through the trusted execution technology (TXT) [15, 33] and secure virtual machine (SVM) [16, 34] technology, respectively. The two approaches are implemented in a different way but both provide the means of creating VMs through totally hardware measures. The collaboration of the TXT or SVM technology with the TCG TPM enables DRT measurement and provides IEE as described in previous sections.

Using any of the above virtualization approaches, a system security designer can create trusted areas of virtual machines running on virtualizated hardware and direct sensitive applications and data toward those virtual machines. Extending this logic, trusted OS can be run on such VM and as long as access is controlled by a TCB program on the processor, the OS remains isolated and protected from the rest of the system's untrusted VMs. Trusted virtualization technology is still on an early stage, since several practical implementation problems still exist (hardware constrains, system real-time behavior, scheduling, access control rights). However, virtualization opens the road for unlimited hardware/software codesigned structures that through HSMs like the TCG TPM, can lead to very high trust-level systems.

6.6 Conclusions and Future Directions

In this chapter, the road toward designing hardware-based trusted system was described, beginning from monolithic implementations of HSM for specific applications and moving toward very dynamic solutions like the TPM offering sophisticated functionality to associated software. Parallel to this approach, virtualization was evolved over time into a very useful security enhancement tool. Thus, the merging of the hardware security world with software virtualization has resulted in very strong isolation mechanisms capable of providing high trust level. In the future, this approach is bound to be expanded and further adopted. TPM chips will migrate from the desktop, notebook, and server computer domain to the mobile world enabling us to, ultimately, trust our mobile devices (smart phones, tablets) for security-sensitive transactions. The TCG has moved toward this direction by providing specifications for a mobile TPM version (Mobile Trusted Module, MTM) but these specs have not led to a market product yet. Furthermore, the growing adoption of embedded computing systems in everyday devices has stemmed the need for strong security and trust. Trusted computing will play a very dynamic role in securing the embedded system world and TPM structures along with virtualization will, eventually, be infused into embedded systems.

On the other hand, the current TCG specifications (TPM v1.2) on the TPM have several shortcomings since they lack flexibility and diversity. Already, TCG is working on changing that by providing a TPM v2.0 (a preliminary draft was released in October 2012). From cryptographic perspective, designers need to put behind the aging cryptographic infrastructure of the existing TPM and adopt more efficient cryptographic schemes. Researchers point to Elliptic Curve cryptography (ECC) as the most suitable candidate for a public key scheme and to SHA-256 or the upcoming SHA-3 as the most suitable Hash function scheme [35, 36]. The presence of an ECC framework can provide additional TPM features such as ECC pairing-based cryptography (PBC) (using Weil pairing, Tate pairing, Eta pairing, Ate pairing, etc.) capable of supporting advanced security services such as short signatures, identity-based encryption and signature, identity-based authenticated key agreement, Tripartite Diffie-Hellman or self-blindable credentials [37]. The Direct Anonymous Attestation (DAA) mechanism, adopted by TCG, that is currently based on symmetric pairings can be more efficiently realized using ECC asymmetric pairings [38].

Finally, the wide adoption of multicore processors will have a profound impact on trusted computing and virtualization. Apart from high-speed implementations, dynamic use of multiple cores will enable better hardware protection patterns. Hypervisors will be able to assign whole VMs to specified cores and provide hardware virtualization on processor core level.

References

1. Challener, D., Yoder, K., Catherman, R., Safford, D., & Van Doorn, L. (2007). *A practical guide to trusted computing*. Boston: IBM press.
2. Anderson, R., Bond, M., Clulow, J., & Skorobogatov, S. (2006). Cryptographic processors-a survey. *Proceedings of the IEEE*, *vol. 94*, 2, pp. 357–369.
3. PracTel Inc. (2007). Tetra and tetrapol: Technology and market comparison. PracTel Inc.
4. Artz, D., & Gil, Y. (2007). A survey of trust in computer science and the semantic web. *Web Semantics: Science, Services and Agents on the World Wide Web, 5*(2), 58–71.
5. Gligor, V., & Wing, J. M. (2011). Towards a theory of trust in networks of humans and computers. *Security Protocols XIX. Springer*, pp. 223–242.
6. Aussel, R. B., & Sailer, J. D. A. (2011). Only hardware-assisted protection can deliver durable secure foundations. *IEEE Software*, *28*, 2, pp. 57–59.

7. Rushby, J. (1984). A trusted computing base for embedded systems. *Proceedings 7th DoD/NBS Computer Security Initiative Conference*, *Gaithersburg, MD*, Sep. 1984, pp. 294–311.
8. Criteria, C. Online. http://www.commoncriteriaportal.org.
9. Iqbal, A., Sadeque, N., & Mutia, R. I. (2009). An overview of microkernel, hypervisor and microvisor virtualization approaches for embedded systems. *Report, Department of Electrical and Information Technology, Lund University, Sweden*, *2110*.
10. Alves, T., & Felton, D. (2004). Trustzone: Integrated hardware and software security (ARM white paper). *Information Quarterly, 3*(4), 18–24.
11. Armtrustzone, A. R. M. http://www.arm.com/products/processors/technologies/trustzone.php.
12. Group, T. C. (2007). TCG TPM specification version 1.2. https:/www.trustedcomputinggroup.org/specs/TPM/.
13. Group, T. C. (2006). TCG software stack (tss) specification version 1.2. http://www.trustedcomputinggroup.org/resources.
14. Fisher, D. A., McCune, J. M., & Andrews, A. D. (2011). Trust and trusted computing platforms. DTIC Document, Tech. Rep., 2011.
15. Intel. (2011). Intel trusted execution technology (intel txt).
16. Devices, A. M. (2005). AMD, secure virtual machine architecture reference manual.
17. Chen, L., Morrissey, P., & Smart, N. (2008). Pairings in trusted computing. *Proceeding of Pairing-Based Cryptography Pairing*, pp. 1–17.
18. Shen, Z., & Wu, X. (2008). A Trusted Computing Technology Enabled Mobile Agent System. *Computer Science and Software Engineering, International Conference on*, *3*, pp. 567–570.
19. Wilhelm, U., Staamann, S., & Buttyan, L. (1998). On the Problem of Trust in Mobile Agent Systems. *Internet Society's Symposium on Network and Distributed System Security*.
20. Tan, H. K., & Moreau, L. (2001). Trust Relationships in a Mobile Agent System. *Mobile Agents, number 2240 in LNCS*, *Springer*, pp. 15–30.
21. Uwe, S. S., Wilhelm, G., & Buttyan, L. (1999). Introducing Trusted Third Parties to the Mobile Agent Paradigm. *Secure Internet Programming: Security Issues for Mobile and Distributed Objects*, *Springer-Verlag*, pp. 471–491.
22. Hein, D., & Toegl, R. (2009). An autonomous attestation token to secure mobile agents in disaster response. *The First International ICST Conference on Security and Privacy in Mobile Information and Communication Systems (MobiSec 2009)*. Torino, 2009. From HST to Trusted Computing and Trusted systems 19.
23. Perez, R., van Doorn, L., & Sailer, R. (2008). Virtualization and hardware-based security. *Security & Privacy, IEEE*, *6*, *5*, pp. 24–31.
24. Garfinkel, T., Pfaff, B., Chow, J., Rosenblum, M., & Boneh, D. (2003). Terra: A virtual machine-based platform for trusted computing. *ACM SIGOPS Operating Systems Review*, *37, 5. ACM*, pp. 193–206.
25. Berger, S., Caceres, R., Goldman, K., Perez, R., Sailer, R., & van Doorn, L. (2006). vTPM: Virtualizing the trusted platform module. *Proceedings of 15th Conf. on USENIX Security Symposium*, pp. 305–320.
26. Stumpf, F., Benz, M., Hermanowski, M., & Eckert, C. (2007). An approach to a trustworthy system architecture using virtualization. *Autonomic and Trusted Computing*, *Springer*, pp. 191–202.
27. Stumpf, F., & Eckert, C. (2008). Enhancing trusted platform modules with hardware-based virtualization techniques. *Emerging Security Information, Systems and Technologies, 2008. SECURWARE' 08. Second International Conference on IEEE*, pp. 1–9.
28. Sailer, R., Valdez, E., Jaeger, T., Perez, R., Van Doorn, L., Griffin, J. L., & Berger, S. (2005). sHype: Secure hypervisor approach to trusted virtualized systems. *Techn. Rep. RC23511*.
29. Härtig, H., Hohmuth, M., Liedtke, J., Wolter, J., & Schönberg, S. (1997). The performance of μ-kernel-based systems. *ACM SIGOPS Operating Systems Review*, *31, 5, ACM*, pp. 66–77.
30. Heiser, G. (2008). The role of virtualization in embedded systems. *Proceedings of the 1st workshop on Isolation and integration in embedded systems*, *ACM*, pp. 11–16.
31. Heiser, G., Andronick, J., Elphinstone, K., Klein, G., Kuz, I., & Ryzhyk, L. (2010). The road to trustworthy systems. *Proceedings of the fifth ACM workshop on Scalable trusted computing*, *ACM*, pp. 3–10.

32. Heiser, G., & Leslie, B. (2010). The okl4 microvisor: Convergence point of microkernels and hypervisors. *Proceedings of the first ACM asia-pacific workshop on Workshop on systems*, *ACM*, pp. 19–24.
33. Uhlig, R., Neiger, G., Rodgers, D., Santoni, A. L., Martins, F. C., Anderson, A. V., Bennett, S. M., Kagi, A., Leung, F. H., & Smith, L. (2005). Intel virtualization technology. *Computer*, *38*, *5*, pp. 48–56.
34. Strongin, G. (2005). Trusted computing using amd pacifica and presidio secure virtual machine technology. *Information Security Tech. Report*, *10*, *2*, pp. 120–132.
35. Zhang, X., Zhou, M., Zhuang, J., & Li, J. (2007). Implementation of ECC-Based Trusted Platform Module. *Machine Learning and Cybernetics, 2007 International Conference on*, *4, August, IEEE*, pp. 2168–2173.
36. Fournaris, A. (2012). Toward flexible security and trust hardware structures for mobile-portable systems. *Latin America Transactions, IEEE (Revista IEEE America Latina)*, *10*, *3*, pp. 1719–1722.
37. Barreto, P. S., Kim, H. Y., Lynn, B., & Scott, M. (2002). Efficient algorithms for pairing-based cryptosystems. *Advances in cryptologyCRYPTO 2002*. *Springer*, pp. 354–369.
38. Brickell, E., Chen, L., & Li, J. (2008). A new direct anonymous attestation scheme from bilinear maps. *Trusted Computing-Challenges and Applications, Springer,* pp. 166–178.