Attestation in Trusted Computing: Challenges and Potential Solutions

A computing entity must have confidence in the identity and trustworthiness of another target platform before interacting with it. **Andrew Lee-Thorp** examines the attestation specification put forward by the Trusted Computing Group and assesses its fitness for purpose.

Royal Holloway University of London

SearchSecurity.co.UK

abstract

A trusted platform is a computing platform designed to provide trusted features as specified by the Trusted Computing Group (TCG). Trusted platforms sport a distinguishing feature known as (remote) attestation: a trusted platform can vouch for its own integrity. The purpose of attestation is to allow a third party to obtain confidence in the identity and trustworthiness of a target platform before it interacts with the target. In this report, Andrew Lee-Thorp examines the state of play in TCG attestation by asking the questions: how practical is the attestation specification and does it meet the needs of designs that propose to take advantage of trusted computing functionality?

1 INTRODUCTION

TRUSTED COMPUTING is a response to the security challenges facing computing platforms today [1]. Even software that has been correctly installed and configured may be affected by other software (for example malware) on the same platform or by software that has bypassed the operating system (for example a rootkit).

One response is to purchase a dedicated security solution, hardware and/or software, that allows the users of a service to establish confidence in a platform. This presents a number of problems. Firstly, such secure computing solutions tend to be expensive. Secondly, it is frequently the case that security equipment falls foul of government export restrictions on cryptography. Thirdly, whilst this may be a valid response for a large e-business it doesn't scale to the array of cheap, open platforms having an assortment of hardware architectures. Furthermore it still requires a client to have confidence in the owner of a server platform. A client platform cannot independently verify the identity and configuration of a server platform, nor can it establish that its data will only be used in the manner intended.

The TCG specification set aims to address

many of these problems whilst retaining some of the benefits of commodity computing platforms: low cost, flexibility and openness. Fundamental to trusted computing is a low-cost, tamper-evident computing chip (known as a Trusted Platform Module, or TPM for short) which is intended to be mounted on a computer motherboard.

Broadly speaking trusted platforms provide the following services [1]: protected storage of data, prevention of inappropriate access to data, identification, and support for trust mechanisms rooted in secure hardware.

Trusted platforms are capable of collecting and reporting (i.e. attesting to) evidence of behaviour. Endorsement of the trusted platform by third parties allows that evidence to be trusted. This capability is an enabler for more pervasive trusted inter-connectivity in which, for example, organisations can expose their computer systems and remain confident that data is used only in the way it is intended.

It turns out that TCG measurements of platform behaviour are intrinsically low-level, making it difficult for a relying party to translate this evidence into the rich policies that are meaningful to enterprises or ordinary users. Furthermore the application of TCG attestation is not straightforward. These factors present a major obstacle to the widespread adoption of attestation [2]. So how practical is TCG attestation and does it meet the needs of designs that propose to take advantage of trusted computing functionality? It is argued here that broadly speaking, TCG attestation falls short of meeting its stated goals in both specification and implementation aspects.

2 TCG ATTESTATION

A Trusted Platform Module (TPM) is a computing chip with a crypto coprocessor, secure memory, a computing engine and I/O components, attached to a computing platform (e.g. PC). Of particular relevance to this discussion is a set of special-purpose platform configuration registers (PCRs). PCRs are shielded hardware registers - 20-byte, dedicated internal storage units, separate from general-purpose, non-volatile memory - and with the sole purpose of recording the aggregate platform state. A protected capability on a PCR called extend is defined in such a way that the current value constitutes a trust chain (of events). A trust chain begins with a well-known initial state and comprises the sequence of events up to and including the event that brings a platform into its current state. The sequence of events is bootstrapped by a hardware root for trust for meas-

urement (RTM) which be may static (when the platform is initialised) or dynamic (in-flight booting into a predetermined state).

The basic idea (Figure 1, below) is to add some software known as a measurement agent (MA) to each layer of the platform stack. Each MA, starting with the core root of trust for measurement (CRTM) which is part of the trusted hardware subsystem, measures the local platform environment and the next MA (Step 1). It atomically extends a PCR with the new integrity metric and appends the measured value (i.e. information that describes the event) to the Storage Measurement Log (SML) (Step 2). Then control is handed to the next MA (Step 3). The chain of measure-record-handoff terminates with the final MA which is part of the software that is loaded and running once the platform has booted (typically the OS). The final MA continues to record integrity altering events.

FIGURE 1

AUTHENTICATED BOOT

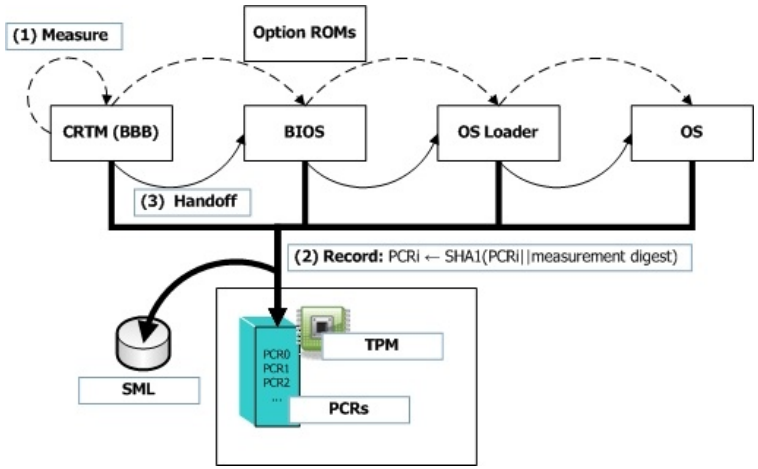


TABLE 1

USAGE ASSIGNMENTS FOR S-RTM PCRS

PCR Index	PCR Usage
0-3	CRTM, BIOS, and initialisation of hardware (e.g. CDROM)
4,5	OS Boot Loader and configuration
6	State Transition and Wake Events (e.g hibernation)
7	Host Platform Manufacturer Control (Reserved for OEM)
8-15	Defined for use by the static OS (OS specific)
16	Debug
23	Application support (OS specific)

A TPM must support at least sixteen PCRs. Their usage assignments are summarised in Table 1, above.

This process, referred to as authenticated boot, allows a platform to record evidence of platform behaviour which can later be presented to a remote party in a platform attestation. The evidence comes in two forms, namely the PCR values and the event history. This allows the rely-

ing party to obtain confidence in the trustworthiness of an attesting platform.

To satisfy these requirements, the attesting platform's TPM signs a fresh challenge and the PCR values (see Figure 2, below), using a special purpose private Attestation Identity Key (AIK). The challenger verifies the signature and event history before deciding whether to trust the attesting platform.

1. The relying party sends a message to the attesting platform in which it requests evidence of behaviour and authenticity. Two parameters in this request are noteworthy: the first is a nonce to guarantee freshness and the second is an array of PCR indices. The indices are a subset (or all) of the PCRs which provide a view (Table 1) of the platform configuration.

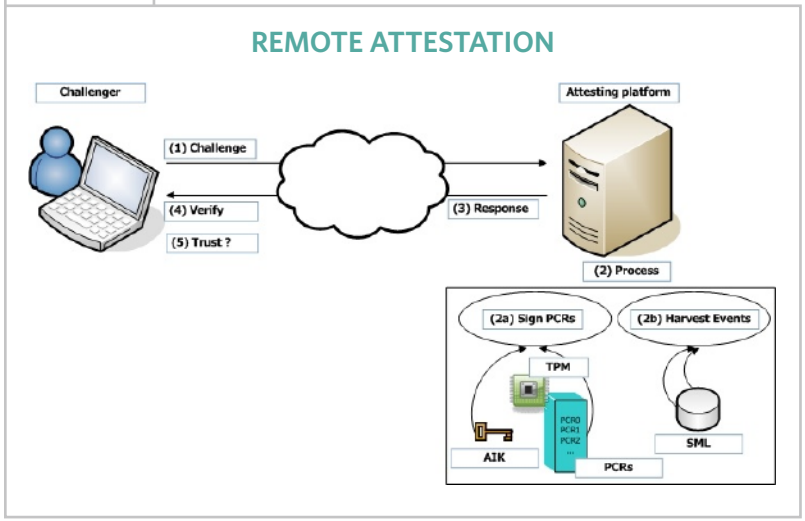
2. A platform agent gathers the evidence on the attesting platform. Firstly, the TPM signs the nonce and the selected PCR values using the private AIK. Secondly, the event history for the selected PCRs is retrieved.

3. The platform agent returns the signature, event history and platform credentials to the challenger.

4. The relying party verifies the signature (using the platform credentials) and validates the event history.

5. This establishes the identity and configuration of the attesting platform. If the relying party trusts the third party (this is the purpose of the platform credentials) that vouched for the attesting platform then it is able to decide whether to trust the platform. The relying party may now compare the attested configuration to a set of reference configurations that are deemed to be trustworthy.

FIGURE 2



This is best illustrated with an example. Suppose, firstly that an adversary attempts to install and execute trojan software on a platform. This event will have been recorded by the OS MA and

can be detected when the attested evidence is inspected by a relying party. To thwart this the adversary must replace the OS with an evil OS and an evil MA. This fact will have been noticed by the OS loader when it measured the OS and hence can also be detected by a relying party. In order to prevent this the adversary must subvert the OS loader but this fact will have been measured by the BIOS. It follows by induction that only the CRTM needs to be trusted.

The preceding discussion omitted some important details about the AIK and TPM key management in general. TCG specifies rules governing the purpose, usage and lifecycle of several special-purpose keys and these rules are enforced by the TPM. One such key is the attestation identity key (AIK). TCG makes a distinction between proving that a platform is a genuine Trusted Platform and proving that a platform is a particular Trusted Platform instance. For this reason and for the reason that a Trusted Platform may possess several AIKs (e.g. one for internet banking and one for peer-peer networking), AIKs are referred to as (pseudonymous) identities. The TPM associates authorisation and mobility attributes with keys that are used to enforce access control. Mobility is an important concept in the TCG specification. Key mobility falls into one of the following categories:

- Migratable keys are trusted only by their originator and can be moved to another platform.
- Certified Migratable keys (CMKs) are migratable keys that can be moved but remain secure.
- Non-migratable keys are known to, and used exclusively by, the TPM that owns them. Non-migratable keys are trusted by everyone as everyone can be sure that (i) they are created within a TPM, (ii) they never appear outside a TPM and (iii) that all operations using the private key are performed within the TPM.

3 INHERENT LIMITATIONS

The RTM is the root for a highly generalised trust chain. In the TCG model, integrity measurements are hashes of any software and its configuration that (potentially) alters the trust state of the platform. For this reason every element of the boot process is measured and recorded. A measurement is taken when software is first loaded, prior to execution - leading to the description “binary, load-time” attestation. The use of the term binary is overloaded here: since a set of integrity metrics defines a single platform configuration the resultant

policy is all or nothing.

The existing definition of the TCG trust chain has inherent limitations some of which are summarised here.

- Owing to the multitude of implicit interdependencies between trusted third parties, revoking a TPM identity (or identities if cascading revocation is required) is hard.
- A large semantic gap exists: users and applications require a high-level expression of policies that cannot be encoded using low-level binary measurements.
- The trust chain does not identify a software entity but a measured platform configuration. For example, it is not possible to tie the use or ownership of a secret to a particular application. Any software that is part of a configuration can access any secret that is bound to that configuration. This is in contrast with common practice in which knowledge of a secret (e.g. a private key) is used to identify the entity which (it is assumed) owns the secret.
- An attestation is a declaration, conveyed at a point in time, and says nothing about the running state. Opportunities for time-of-check-time-of-use attacks abound.
- The use of static RTM requires a challenger to verify a long trust chain.

4 USING ATTESTATION

The purpose of an attestation is to identify a platform as being a TCG trusted platform and then to decide based on the attested configuration whether the platform can be trusted for a particular purpose.

We illustrate some challenges by examining a popular theme: secure download and protected execution of a privileged application. In our simplified example, a server, Bob, wants to securely deploy and execute a high assurance application onto a trusted platform-enabled client, Alice. Bob wants to ensure that a specific download helper, h, and a secure execution environment (SEE) are available on Alice. The following steps and attestable elements have been proposed to achieve this objective [8].

1. Bob communicates a policy (a set of PCR values) to Alice. The policy represents a predicted platform state under which Alice may download and execute Bob's application. Part of this policy specifies a SEE, and the availibility of h to both perform the download and enforce an execution policy.
2. Alice creates a public key pair using TPM CreateWrapKey. Access to the private key component is conditional upon the current platform configuration being equal to Bob's policy.

3. Using an AIK, Alice's TPM attests to the mobility constraints and the state under which the new private key will be released.

4. Bob verifies Alice's key attestation, satisfies himself as to the suitability of the attested PCR metrics and then proceeds as follows. Bob generates a symmetric key, encrypts the application with the symmetric key and encrypts the symmetric key with Alice's new public key. Then Bob sends to Alice a bundle consisting of the encrypted application and the encrypted symmetric key.

5. Alice can only load the private decryption key (to decrypt Bob's bundle and execute the application) if her platform state is equal to the attested policy.

Let's analyse the resulting context.

- We first make the observation that Bob seeks to impose a usage policy on Alice. Bob must express this policy in TCG-speak, i.e. as a binary platform configuration. A separate trustworthy configuration is required for every combination of possible hardware and OS configuration, i.e. for any existing type of Alice. In order to manage this complexity Bob decides to reduce the scope of supported clients to a few popular configurations and exclude market players having a small user base.
- Having fixed the set of trustworthy configurations Bob discovers that his scheme is not future-proof. As soon as Alice applies the latest vendor patches (e.g. security updates) she can no longer run Bob's downloaded application as her modified platform configuration no longer matches Bob's policy.
- Alice is concerned that she has disclosed private information to meet Bob's requirements. At a minimum she has probably disclosed her hardware and OS configuration.
- Bob has assumed that h is executing inside the SEE. There is no information in a TCG attestation that explicitly binds one executing environment to another.
- Bob has assumed that only h can load the private key in order to decrypt the encrypted downloaded application. Since a TCG attestation identifies a platform configuration (not a single entity), any process within the attested configuration can load the TPM key to access the encrypted application. This means that Bob has to be very careful how he evaluates trustworthy configurations.

Bob can improve this scheme by generalising his requirements into a layered approach. Using three layered policies results in a package in which the application is encapsulated by three

layers of encryption which correspond to three sets of TPM wrapped public key pairs. The outer, middle and inner layers reflect the download, monitoring, and application execution policies respectively. The resulting context is slightly more favourable.

Firstly, fewer assumptions are made about the download helper which can now execute in an insecure environment. Secondly the opportunities for time-of-check-time-of-use attacks are restricted to application execution.

5 ATTESTATION IN PRACTICE

What can we learn from attestation reports that experiment with TCG hardware and specification? In this section we review the notion of attesting to properties [13] and an experience report applying TCG attestation to Linux [14].

Property-based attestation (PBA) PBA was independently proposed in [12, 13] principally to address perceived deficiencies of TCG attestation. Some of these were alluded to in the previous section, namely lack of privacy, the ability to exclude configurations and therefore restrict consumer choice, and poor scalability.

The underlying idea is that a platform only needs to attest to "properties" which can then be tested against the relying party's security requirements. Recall that in TCG attestation a relying party compares an attested binary configuration to a set of binary configurations deemed to be trustworthy. In order to support PBA, a so-called property certificate is issued by a trusted third party (e.g. evaluation facility). A property certificate is a statement to the effect that a platform with binary configuration C provides the security properties P_0, P_1, . . . , P_n. In this way multiple platform configurations could meet the same security properties. The exact definition of a "property" is somewhat nebulous. It could mean the absence of certain vulnerabilities or built-in measures to enforce privacy laws.

There are, however, two fundamental flaws, one technical and one commercial. To understand the former, recall that the challenger selects the PCR indices in an attestation.

The attesting platform must have a property certificate for each and every likely combination of PCR indices or the attestation request cannot be satisfied. Finally, the property certifier (it seems) derives no business benefit but significant cost and liability.

5.1 LINUX INTEGRITY MEASUREMENT

TCG binary, load-time attestation has been

implemented on Linux [14]. The implementation reports on the platform configuration once the OS has fully booted. The design, dubbed Integrity Measurement Architecture (IMA) by its creators, consists of three mechanisms: measurement, integrity challenge and integrity validation. The measurement mechanism consists of three parts: (i) a modified BIOS and GRUB bootloader to measure the initial kernel code, (ii) a kernel instrumented to measure changes to itself and (iii) a modified runtime system instrumented to take integrity measurements when user-level executable content is loaded, but before it is executed.

In an important departure from vanilla TCG-attestation the IMA does not report the complete trust chain. The system does not record new events that would present the same measurement value. An example of this is a web-server that has been started twice.

This was done to bound the size of the measurement list (and hence the length of the trust chain) to a reasonable value.

This leads to a dilemma. Since the trusted computing base (TCB) is ill-defined, the measurement scope must be the entire system. All entities must be measured irrespective of whether they actually impact the integrity of an application. This requires the relying parties to recognise all measured entities.

6 HARDWARE AND SOFTWARE THREATS

The goal of TCG attestation is to protect information from software attack. Nevertheless, the relative ease at which TCG hardware can be attacked should be a cause for concern. Kauer [11] demonstrated that by manipulating the Low Pin Count (LPC) Bus a hardware reset could sent to the TPM independent of the rest of the platform. If the BIOS is patchable (as it is on many machines) then the BIOS TPM driver can be modified to swap in a new CRTM.

Sparks [18] reproduced the TPM reset attack and also provided evidence that the CRT-based RSA engine in the TPM under investigation was susceptible to timing attacks on the RSA private key. Load-time TCG semantics are inadequate under most software threat assumptions [18, 5] by admitting a large array of time-of-check-time-of-use attacks.

Time-of-check-time-of-use is a class of attack that is not well handled by TCG load-time attestation. For example, consider an attack in which a buffer overflow is exploited that leads to unauthorised access to sensitive information. Since it

is not a software load and execution event this event will not be measured and as a result the system compromise will not be reported in an attestation.

7 CONCLUSION

7.1 State of Play

The principle behind the (TCG) trusted platform is to expand a chain of trust from a single trust foundation. Elements in the trust chain are events (typically software load events) that alter the trust status of a platform.

Before summarising, we note that the TCG model of attesting to a trust chain is by no means the only model for substantiating trust in a system. The interested reader is referred to [16, 12, 13, 10, 4, 17, 15] for other examples.

In examining the state of play in TCG attestation the following issues have been briefly treated.

1. How is attestation (to be) used?
2. What do attestation systems look like in practice and are these systems practical?

In TCG, access to secrets are contingent upon the platform configuration matching a configuration to which the secret is bound. In theory an event measurement system should record every event that changes the trust status of a platform. The size of the resultant trust chain means that S-RTM based load-time measurement simply doesn't scale. Next, consider events e_1, e_2 and e_3. There are nine permutations of e_1, e_2 and e_3. Each permutation (i.e. configuration) results in a distinct trust status. As the number of events increases the combinatorial explosion of potential configurations makes it impractical for the relying party to list all reference trusted configurations. Load-time event measurement is inadequate as it fails to handle TOCTOU attacks.

The presents a dilemma: measuring a greater number of events and event types will not scale but the existing granularity of measurement is insufficient.

7.2 Future Directions

In order to balance the need for higher assurance with the goal of practical verification in an open architecture there are two complementary approaches that preserve the existing TCG effort in binary attestation.

1. Extend the scope of the TCG architecture to encompass protection capabilities at the platform level and allow developers to express security requirements using new primitives that expose these capabilities (see e.g. [5]).

2. Exploit isolation capabilities of new processor hardware to reduce the size and scope of the attestable trusted computing base by reorganising applications into distinct parts that do and don't need to be trusted.

Extending TCG architecture One response to the above dilemma is to extend the scope of the TCG architecture. For example, a hardware modification can be made to allow the TPM to "listen" to unauthorised memory access [5]. Such a method itself would need to be attested.

Isolation and new application paradigms A low-level assurance of integrity (i.e. hashes of the software image) does not provide the means to encode the rich security policies appropriate to most applications. It is, however, entirely appropriate for low-level entities such as the BIOS, boot loader and device drivers that carry no behavioural assurance. At the same time virtualisation technologies exist and are highly efficient [3].

This suggests a divide and conquer approach: divide the architecture into an untrusted part (the ill-defined "messy") and trusted part (having well-defined security and assurance requirements) [9, 6]. The principle of division of privilege is well understood and has led to noticeable improvements in application security.

In this paradigm, applications have distinct privileged and unprivileged parts. The privileged part runs in an isolated compartment in a hypervisor. If the hypervisor is an isolation kernel [6] its security can be evaluated. With D-RTM, attesting to the binary integrity of the hypervisor becomes more practical and a layer (possibly within the hypervisor) can support higher-level attestation semantics. In the context of trusted computing, examples of this style of application deployment already exist [7]. ■

ABOUT THE AUTHOR

Andrew Lee-Thorp started life as a researcher and student in Oceanography before switching to his other main interest - computer science. His first introduction to security was the EMV ICC specification (aka chip and pin) which he implemented on the MULTOS smart card platform.

His principal interests are in operating system and application security as well as penetration testing and UNIX security.

REFERENCES

[1] Boris Balacheff, Liqun Chen, Siani Pearson, David Plaquin, and Graham Proudler. Trusted Computing Platforms: TCPA Technology in Context. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2003.

[2] Shane Balfe, Eimear Gallery, Chris J Mitchell, and Kenneth G Paterson. Challenges for Trusted Computing. IEEE Security and Privacy, 6(6):60–66, 2008.

[3] Paul Barham, Boris Dragovic, Keir Fraser, Steven Hand, Tim Harris, Alex Ho, Rolf Neugebauer, Ian Pratt, and Andrew Warfield. Xen and the art of virtualization. In SOSP '03: Proceedings of the nineteenth ACM symposium on Operating systems principles, New York, NY, USA, 2003. ACM.

[4] Stefan Berger, Ram´on C´aceres, Kenneth A. Goldman, Ronald Perez, Reiner Sailer, and Leendert van Doorn. vTPM: virtualizing the trusted platform module. In USENIX-SS'06: Proceedings of the 15th conference on USENIX Security Symposium, Berkeley, CA, USA, 2006. USENIX Association.

[5] Sergey Bratus, Nihal D'Cunha, Evan Sparks, and Sean W. Smith. TOCTOU, Traps, and Trusted Computing. In Trust '08: Proceedings of the 1st international conference on Trusted Computing and Trust in Information Technologies, Berlin, Heidelberg, 2008. Springer-Verlag.

[6] Paul England, Butler Lampson, John Manferdelli, Marcus Peinado, and Bryan Willman. A Trusted Open Platform. Computer, 36(7), 2003.

[7] Sebastian Ga jek, Ahmad-Reza Sadeghi, Christian Stuble, and Marcel Winandy. Compartmented Security for Browsers - Or How to Thwart a Phisher with Trusted Computing. In ARES '07: Proceedings of the The Second International Confer- ence on Availability, Reliability and Security, Washington, DC, USA, 2007. IEEE Computer Society.

[8] Eimear Gallery. Secure delivery of conditional access applications to mobile receivers. In Chris Mitchell, editor, Trusted Computing (Professional Applications of Computing) (Professional Applications of Computing), chapter 7. IEEE Press, Piscataway, NJ, USA, 2005.

[9] Tal Garfinkel, Ben Pfaff, Jim Chow, Mendel Rosenblum, and Dan Boneh. Flexible OS support and applications for Trusted Computing. In Peter Lipp, Ahmad-Reza Sadeghi, and Klaus-Michael Koch, editors, Proceedings of the 9th Workshop on Hot Topics in Operating Systems (HotOS IX), 2003.

[10] Tal Garfinkel, Ben Pfaff, Jim Chow, Mendel Rosenblum, and Dan Boneh. Terra: a virtual machine-based platform for trusted computing. In SOSP '03: Proceedings of the nineteenth ACM symposium on Operating systems principles, New York, NY, USA, 2003. ACM.

[11] Bernhard Kauer. Oslo: improving the security of trusted computing. In SS'07: Proceedings of 16th USENIX Security

Symposium on USENIX Security Symposium, Berkeley, CA, USA, 2007. USENIX Association.

[12] Jonathan Poritz, Matthias Schunter, Els Van Herreweghen, and Michael Waidner. Property Attestation : Scalable and Privacy-friendly Security Assessment of Peer Computers. Technical Report RZ 3548 (99559) 05-10-04, IBM Research GmbH, Zurich Research Laboratory, 8803 Ruschlikon, Switzerland, 2004.

[13] Ahmad-Reza Sadeghi and Christian St¨uble. Property-based attestation for computing platforms: caring about properties, not mechanisms. In NSPW '04: Proceedings of the 2004 workshop on New security paradigms, New York, NY, USA, 2004. ACM.

[14] Reiner Sailer, Xiaolan Zhang, Trent Jaeger, and Leendert van Doorn. Design and implementation of a TCG-based integrity measurement architecture. In SSYM'04: Proceedings of the 13th conference on USENIX Security Symposium, Berkeley, CA, USA, 2004. USENIX Association.

[15] Dries Schellekens, Brecht Wyseur, and Bart Preneel. Remote attestation on legacy operating systems with trusted platform modules. Sci. Comput. Program., 74(1-2), 2008.

[16] Elaine Shi, Adrian Perrig, and Leendert van Doorn. BIND: A Fine-grained Attestation Service for Secure Distributed Systems. In IEEE Symposium on Security and Privacy, 2005.

[17] Sean W. Smith. Outbound Authentication for Programmable Secure Coprocessors. In ESORICS '02: Proceedings of the 7th European Symposium on Research in Computer Security, London, UK, 2002. Springer-Verlag.

[18] Evan R. Sparks. A Security Assessment of Trusted Platform Modules. Technical Report TR2007-597, Dartmouth College, Computer Science, Hanover, NH, June 2007.