Challenges for Trusted Computing

S. Balfe, E. Gallery, C.J. Mitchell and K.G. Paterson

Technical Report
RHUL–MA–2008–14
26 February 2008

Abstract

This article identifies and discusses some of the key challenges that need to be addressed if the vision of Trusted Computing is to become reality. Topics addressed include issues with setting up and maintaining the PKI required to support the full set of Trusted Computing functionality, the practical use and verification of attestation evidence, and backwards compatibility, usability and compliance issues.

1 Introduction

A trusted platform, as discussed in this article, refers to a platform of the type championed by the Trusted Computing Group (TCG). That is, a trusted platform is one which will behave in a particular manner for a specific purpose. Trusted computing refers to the collection of interrelated and interoperating technologies, which, when combined, help to establish a more secure operating environment on commodity platforms. A fully-realised trusted computing platform will allow users to reason about the behaviour of a platform, as well as providing standardised mechanisms to protect user private data against software attack [36]. A trusted platform should be able to reliably gather and provide evidence of its current operating state, or any subcomponent thereof. Any divergence from an intended operating state can be reported to interested parties, allowing them to make informed decisions as to whether to continue to interact with the platform in question.

Trusted Computing functionality has been proposed to enhance the security of numerous applications. For example, it has been promoted as an adjunct to the digital signature process [6, 49], in enabling secure software download [19], in support of secure single sign-on solutions [29], in securing peer-to-peer networks [8, 23, 45, 43], in improving the security and privacy of a biometric user authentication process [14], in hardening mobile devices [18] and to facilitate identity management [27, 28]. A number of authors have also considered trusting computing's applicability to the agent paradigm [15, 30, 31, 35], grid security [25, 26], e-commerce transaction security [10, 9], and to defend against the ever-growing threat posed by crimeware [7]. Further applications are described in [5, 20, 24, 62].

Despite its many beneficial applications, Trusted Computing is not without its detractors; see for example [2, 3, 4, 38, 39, 40, 46, 51, 63]. Privacy concerns relating to trusted platforms were raised in [37]. The extent to which Trusted Computing could be used to enable and enforce Digital Rights Management (DRM), and, more generally, the possible expropriation of platform owner control, have been contentious issues [46, 60]. Anderson [2] expresses the view that Trusted Computing could be used to support censorship, stifle competition between software vendors, facilitate software lock-in and hinder the deployment and use of open source software, thereby potentially enabling market monopolisation by select vendors.

Our aim in this article is not to engage in this debate, but rather to highlight some of the key challenges that we believe need to be addressed in order to accelerate the widespread adoption of Trusted Computing. Many of the challenges we identify are not purely technical in nature, but rather involve a mixture of technical, policy and management aspects. This could make this article somewhat different in flavour from the majority of current research in Trusted Computing. We trust that our work will bring some useful diversity into the field and that our approach will add to, rather than subtract from, the value of the article for readers.

This article is structured as follows. In order to make the article self-contained, Section 2 provides an overview of Trusted Computing technology. In Section 3, we discuss some of the challenges that we believe may hinder the widespread adoption and use of Trusted Computing. In particular, we focus on: the Public Key Infrastructure (PKI) that is associated with the deployment of Trusted Computing; certificate and Trusted Platform Module (TPM) revocation as well as the impact of hardware attacks on platforms; problems with attestation evidence gathering and verification; backward compatibility requirements; usability issues; and challenges arising from non-compliance with the TCG's technical specifications. We conclude with Section 4.

2 Trusted Computing Concepts

The provision of the full range of Trusted Computing functionality is dependent upon the integration of a number of additional hardware and software components into a computing platform:

A Trusted Platform Module (TPM): The TPM forms the core of a Trusted Computing platform [56, 57, 58]. A TPM is a microcontroller with cryptographic coprocessor capabilities that provides a platform with the following features: a number of special purpose registers called Platform Configuration Registers (PCRs), into which cryptographic digests representing platform state altering events can be recorded; a means of reporting the platform's current state to remote entities; secure volatile and non-volatile memory; random number generation; a SHA-1 hashing engine; and asymmetric key generation, encryption and digital signature capabilities.

Core Root of Trust for Measurement (CRTM): The CRTM should form an immutable portion of a host platform's initialisation code and is responsible for measuring the host platform's state, that is, the collection of operating system and application software components running on the machine. The process of measuring platform state may be much more complicated than simply taking a single hash of a monolithic piece of code, however. For example, for a Personal Computer (PC) client [55], the CRTM code would measure a platform's BIOS and store a hashed representation of the BIOS code in one (or more) of the PCRs within a TPM. The CRTM would then hand execution control to the BIOS, which in turn would measure, store, and transfer control to the next component in a host platform's boot sequence. This process would then continue until all components are measured. Through this succession of measuring, transfer of control and execution, a transitive trust chain from the CRTM to the host Operating System (OS) can be formed. Attestations to the measurements recorded to the TPM's PCRs that make up this chain can subsequently be provided to interested parties. (We discuss the different flavours of Trusted Computing attestation in more detail below.)

Isolation technologies: The introduction of isolation technologies, such as Microsoft's Next Generation Secure Computing Base (NGSCB) [32, 33] or XEN Source's XEN [11], were designed to take advantage of CPU and chipset extensions incorporated in a new generation of processor hardware [22]. Together, these new initiatives will enable a platform to be partitioned into isolated execution environments. An isolated execution environment, independent of how it is implemented, should provide the following services to hosted software [33]:

- protection of the software from external interference;
- observation of the computations and data of a program running within an isolated environment only via controlled inter-process communication;
- secure communication between programs running in independent execution environments; and
- a trusted channel between input/output devices and a program running in an isolated environment.

We note that current deployments of Trusted Computing technologies have tended to focus more on the TPM than on the CRTM or isolation technologies. This naturally limits the trustworthiness of the platform. Even so, a number of proposals [8, 9, 44] demonstrate that even a limited deployment of Trusted Computing can bring security benefits.

2.1 Trusted Computing Credentials and Hardware Attestation

In addition to the integration of additional hardware and software components, certification and accreditation play an important role in building a trusted platform. In this regard, there are a number of interrelated credentials that must be present before a platform can be considered to be a *trusted* platform:

An endorsement credential: Each TPM is associated with a unique asymmetric encryption key pair called an *Endorsement Key* (EK) pair. An endorsement credential binds the public component of this key pair to a TPM description and vouches that a TPM is genuine. This credential is typically generated by the TPM manufacturer, with the binding taking the form of a digital signature created using a signing key of the manufacturer.

One or more conformance credentials: Conformance credentials vouch that a particular type of TPM and associated components (such as a CRTM and the connection of the CRTM and TPM to a motherboard) conform to the TCG specifications. These credentials are issued by entities with sufficient credibility to evaluate platforms containing TPMs, typically conformance testing facilities.

A platform credential: A platform credential proves that a TPM has been correctly incorporated into a design which conforms to the TCG specifications. This credential is typically generated by a platform manufacturer. In order to create a platform credential, the platform manufacturer must examine the endorsement credential, the conformance credentials relevant to the trusted platform, and the platform to be certified.

2.2 Identity Attestation

Since an EK is unique to a platform, it may act as a "super-cookie" in identifying subsequent platform actions across multiple domains. To help allay concerns that an EK may become associated with personally identifiable information, the TCG [56] introduced the ability for a TPM to generate and use an arbitrary number of pseudonyms, in the form of *Attestation Identity Key* (AIK) key pairs. Unlike an EK, AIKs serve to identify a platform as trusted without uniquely identifying it as a *particular* trusted platform. However, in order for a relying party to have assurance that an AIK represents a trusted platform, a platform must obtain AIK certification from a mutually trusted third party. Two approaches to AIK certification have been proposed by the TCG.

In the first approach, a trusted third party, referred to as a *Privacy-Certification Authority* (P-CA), provides assurance that an AIK is bound to a trusted platform in the form of an *AIK credential*. When a platform requests an AIK credential from a P-CA, it must supply an AIK public key as well as its endorsement, conformance and platform credentials. The P-CA verifies these credentials, thereby obtaining assurance that the trusted platform is genuine, and creates an AIK credential attesting to this fact. This credential takes the form of a digital signature on the AIK public key. However, this approach has attracted a certain amount of criticism. A P-CA is capable of linking all AIK credentials issued to a specific platform via the platform's EK, putting the P-CA in a position where it is able to defeat the anonymity protection provided by the use of AIKs. Moreover, it is unclear what business model might support the development of commercial P-CAs.

Direct Anonymous Attestation (DAA) has been proposed as an (optional) alternative to the P-CA model, and offers strong anonymity guarantees through advanced cryptographic techniques. These prevent the corresponding issuing authority from being able to link AIK credentials with a single platform identity (EK). A DAA Issuer produces a certificate on a blinded TPM secret. The TPM-host platform later uses this certificate and TPM secret to produce a special type of digital signature (called a signature proof of knowledge) on an AIK public key corresponding to the TPM secret. A verifier of this signature does not actually get to see the certificate, but instead receives an assurance (via a zero-knowledge protocol) that the platform possesses such a certificate on the DAA public key corresponding to the TPM secret.

2.3 Object Attestation

2.3.1 State Attestation

A platform's state is represented by a set of integrity measurements. When a platform component (i.e. a piece of software executing on a platform) is "measured", a hash of the component is recorded to one of a set of PCRs within the host platform's TPM (where the PCR to which a particular component is recorded is platform-specific). Subsequent measurements to this register are recorded by overwriting its current value with a hash of the concatenation of the new measurement and the existing contents of the register. In this way, the cumulative contents of a TPM's registers reflect the current software state of the host platform, as well as the states through which the platform has transitioned.

Platform attestation is the process by which a platform can reliably report evidence of both its identity (as a valid trusted platform) and its current state. A TPM signs the contents of (one or more of) the TPM's PCRs which reflect (all or part of) the current host platform's state. The private component of an AIK key pair is used to create the signature. This signed bundle is communicated to an external entity in conjunction with a corresponding AIK public key credential and a record of the platform components which have been measured. The receiving entity validates the AIK credential, verifies the AIK signature, and compares the signed PCR values against a set of reference integrity measurements. The TCG envisages that reference values and associated credentials will typically be created during software development [59].

2.3.2 Data Attestation

Sealing is the process by which sensitive data can be associated with a set of PCR values representing a particular platform configuration, and only released to the platform for use when the current state of the platform is such that the PCR values match those specified at the time of sealing. The sealing and unsealing processes are implemented using encryption and decryption, in tandem with appropriate key management procedures.

Data can also be associated with a string of 20 bytes of authorisation data before being sealed. When unsealing is requested, the authorisation data must then also be submitted to the TPM. The submitted authorisation data is then compared to the authorisation data in the unsealed string, and the object is only released if the values match.

2.3.3 Key Attestation

A TPM can generate an unlimited number of asymmetric key pairs. For each of these pairs, private key usage and mobility can be constrained. A key pair can be generated so that use of its private component is contingent upon the presence of a predefined platform state (as reflected in the TPM's PCRs). Additionally, a private key can be marked as either being migratable or non-migratable. Migratable private keys can be moved from a TPM, whilst non-migratable keys are inextricably bound to a single TPM.

Private keys, like data, can also be associated with a string of 20 bytes of authorisation data. When private key usage is required, the authorisation data must be submitted to the TPM. The submitted authorisation data is then compared to the authorisation data associated with the key, and key use is only permitted if the values match.

To attest to the usage, mobility and authorisation constraints associated with a private key held by a TPM, the TCG has specified the Subject Key Attestation Evidence (SKAE) X.509 extension [53]. SKAE provides a mechanism to allow a verifier to ascertain that an operation involving a private key was performed within a TCG-compliant TPM environment. After obtaining an AIK credential (following the P-CA method outlined in Section 2.2), a platform can sign a data structure containing the public component of a non-migratable TPM key pair and a description of usage attributes of the corresponding private key. An SKAE Certification Authority (CA), after verifying the signed TPM data structure, can then create a new X.509 certificate for the public key. This certificate incorporates an extension field attesting to the TPM-related security properties of the certified key. Such a certificate can then be used as an aid to authentication in protocols such as SSL/TLS [16] or IKEv2 [1].

3 Challenges for Trusted Computing

3.1 Public Key Infrastructure and Trusted Computing

As we saw in Section 2.1, for a platform to be considered trusted, it must first obtain certificates from an endorsement CA, a platform CA, and one or more conformance CAs. Together, these CAs are responsible for issuing the core trusted platform credentials. However, in order to address privacy concerns resulting from overuse of an EK (as contained in an endorsement credential), P-CAs, and later DAA Issuers, were introduced. Subsequently, SKAE CAs were proposed as a means of coping with difficulties in integrating TPM-controlled keys with standard security protocols. Recently, further-PKI-related authorities (notably Migration Authorities (MAs) and Migration Selection Authorities (MSAs) [52]) have been introduced to address issues concerning key migration between TPM-enabled platforms.

Thus the majority of Trusted Computing services depend fundamentally on the deployment and successful inter-operation of a number of PKI elements. We refer to this collection of components as a *Trusted Computing PKI (TC-PKI)*, although it is actually a larger and more complex "eco-system" of elements than would normally be contained in a single PKI. Figure 1 depicts the main types of CA in a TC-PKI. Yet the development of *any* functional PKI requires a sophisticated combination of organisational, policy-oriented, procedural, and legislative approaches. Indeed the challenges and pitfalls of PKI deployment are well-documented [21, 34], and high-profile system and protocol failures that have been blamed on inappropriate deployment of PKI abound, with SET [47] providing one of the most prominent examples. Put simply, providing a PKI is hard.

A TC-PKI not only involves a plurality of CAs, but also a series of implicit dependencies amongst these CAs. In a TC-PKI, a platform CA relies on the due diligence of an endorsement CA and one or more conformance CAs in accrediting components of a trusted platform. Similarly, both privacy CAs and DAA Issuers rely on platform CAs, endorsement CAs and one or more conformance CAs. Furthermore, SKAE CAs rely on the due diligence of Privacy CAs or DAA Issuers in evaluating the accreditation evidence provided by a trusted platform.

Traditionally, Certificate Policies (CPs) (which specify what a certificate should be used for, and the liability assumed by the CA for this use) and Certificate Practice Statements (CPSs) (which specify the practices that a CA employs to manage the certificates it issues) are deployed by CAs in order to define and limit their liabilities with respect to relying parties. CPs and CPSs are, in fact, an essential component in building a successful PKI, since they give a relying party (which could be an end-user or another CA) a means to manage the business risk in pursuing a particular PKI-related course of action. In the past, uncertainty as to where liability lies has driven up the cost of many PKI implementations [34]. In the absence of CPs and CPSs, implicit cross-certification may exist between CAs, which, as noted in [21], implies that CAs are equally trusted. In such a setting the security of a certificate is reduced to that of the least trustworthy CA. Unfortunately, CPs and CPSs are notoriously difficult and costly to create, and so their production may act as a barrier to entities wishing to provide CA services.

In the setting of a TC-PKI, such policy statements must be produced by every CA upon which another CA may depend. Currently, these dependencies are only informally defined, and, as a result, there is no clear indication of where any liability will lie. Further, at the time of writing, we are not aware of any TC-specific CPs or CPSs having been created. The picture is further complicated by the fact that all the CAs in a TC-PKI rely (at least to some extent) on the endorsement CA. Therefore, the point in a TPM's life-cycle at which an EK credential is acquired impacts on a platform's ability to obtain platform, AIK, DAA and SKAE credentials. In *early normative* EK credential acquisition, as defined by the TCG [54], a TPM manufacturer generates the EK credential. However, in *post-manufacturing* generation, as defined by the TCG [54], a platform owner is responsible for generating the EK credential. In this instance, the certifying body may not be recognised by other CAs and, as a result, the certified TPM host platform may not be able to obtain further credentials from entities outside the domain of its EK credential issuer. In practice, this may not be an issue, as it seems likely that non-manufacturer supplied EK credentials will not be widely used.

To summarise: Trusted Computing relies on an as yet largely unavailable and unspecified PKI in which multiple CAs (possibly existing in different organisational, procedural and/or jurisdictional domains) are expected to inter-operate. This may pose a significant challenge to the future success of this technology.

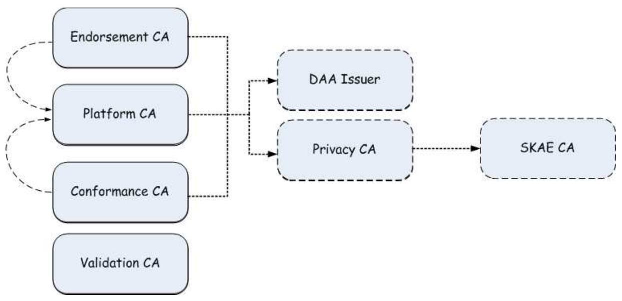

Figure 1: Trusted Computing PKI Components

3.2 Certificate and TPM Revocation

The revocation of credentials within a TC-PKI may introduce further problems. Given the complex dependencies between many of the TC-PKI credentials, the compromise of an individual key and the subsequent revocation of its associated public key certificate will result in a cascading revocation of all dependent TPM credentials. For example, in the event of endorsement key revocation, every AIK associated with the revoked EK must also be revoked. In addition, all SKAE credentials associated with the newly revoked AIKs must also be revoked. This implies that multiple CAs, potentially in independent domains, must be contacted in a timely manner and informed about a revocation decision. This may be a time-consuming and costly endeavour. Further complexity is introduced when attempting to revoke a DAA credential associated with a compromised Endorsement Key pair, because a DAA Issuer cannot link a platform's Endorsement Key pair with a DAA credential.

We next consider revocation of a TPM itself (rather than revocation of its credentials). For cost reasons, the level of tamper-resistance provided by

TPMs is likely to be limited. Moreover, the objective of the mechanisms specified by the TCG is the prevention of information asset compromise through software attack. That is, the software security of the platform is predicated upon the notion that the TPM will maintain an accurate and reliable record of all platform events. Such a focus means that the security of the underlying hardware is assumed and that there is no purely technical driver to promote the development of tamper-resistant TPMs.

Yet it is clear from the example of widespread gaming console modification that, given sufficient incentive, users will actively circumvent hardware-enforced security. In this context, recent demonstrations of a relatively unsophisticated hardware attack [50] through which a TPM's PCRs can be reset without rebooting a platform would appear to pose a significant challenge to Trusted Computing. The ability to reset PCRs effectively destroys the transitive trust chain upon which a remote verifier relies to assess a platform. Once this trust chain is broken, the PCRs can be repopulated with whatever data the platform owner wishes, allowing the owner to misrepresent their platform's current state in a manner that is convincing to a remote verifier. The simple attack of [50] underlines the need for any verifier to consider the "quality" of the platform when assessing the state of a trusted platform. That is, an attestation from a platform incorporating a well-designed TPM from a known manufacturer should be considered more convincing than an attestation from a platform incorporating a TPM from an unknown or disreputable supplier.

Given the above discussion, it is reasonable to assume that, before long, TPMs will be compromised and all credentials and keys extracted. These could then be used to emulate a TPM in software in a way that is indistinguishable from the true hardware TPM. The process by which a compromised TPM is detected will largely be reliant on that TPM's interactions with P-CAs, DAA Issuers and SKAE CAs. It has been suggested in [13] that TPM compromise could manifest itself through an excessive number of certification requests originating from a single TPM host platform (where "excessive" is to be determined by a risk-management policy). However, such a naive approach to detection introduces a number of challenges:

- CAs may specify different thresholds for determining what is meant by "excessive", potentially leading to a high number of false positives for CAs with low thresholds.

- Once a compromised TPM has been detected, there is a need to globally propagate this information to prevent the compromised TPM host platform from being (mis)used elsewhere. This requires the establishment of a global revocation infrastructure. Such an infrastructure could be implemented using Certificate Revocation Lists (CRLs) or through an On-line Certificate Status Protocol (OCSP). Neither option, however, is ideal. In the case of CRLs, there are concerns regarding CRL discovery and the timely issuance of revocation information. In the case of OCSP, in order to make its deployment economically viable, CAs typically charge for each revocation check. It is unclear who would pay for such a service in a TC-PKI. In the case of an OCSP request for an SKAE certificate, the verifier would need to contact the SKAE CA, which would need to contact the AIK CA, which in turn would need to contact the platform, endorsement and conformance CAs.

- A CA must consider potential legal issues that might result from the wrongful issuance of revocation statements negatively impacting on a platform's ability to interact with other parts of the infrastructure. As a result of such considerations, CAs may become reluctant to announce suspected compromises.

- To alleviate the risk of a malicious P-CA issuing falsified revocation statements, a means by which the credibility of CAs in issuing such statements can be assessed must be provided. It is currently unclear what form such a mechanism might take.

3.3 Attestation Evidence Gathering and Verification

The exact parameters to be considered when performing integrity measurements on platform components have yet to be standardised[1]. At a minimum, the parameters must be chosen so that each software component's integrity measurement can be uniquely identified. These measurements must also remain consistent to allow ease of verification. However, in the absence of standardisation, platform integrity measurements may fail to capture all elements required by the verifier of a platform component. This is especially true when one considers the complications introduced with respect to software which relies on dynamically linked libraries (DLLs). In this case, a proportion of the platform component's code base may not be measured, as it will not be loaded by the application prior to execution.

Moreover, given the extensible nature of modern computing systems, the number of components that might need to be measured by a TPM is rapidly

[1]For example, the authors of [33] have proposed that a platform component's integrity measurement can be calculated from its instruction sequence, initial state (i.e. the executable file) and input.

increasing. This implies that each register will have to store multiple measurements. As the number of a platform's components increases, so does the complexity of third party verification of attestation statements. It also becomes difficult for a challenger to verify a single component running on a platform.

The introduction of isolation technologies, as described in Section 2, enables a platform to be partitioned into isolated execution environments, thereby (potentially) simplifying attestation statement verification. In this case, a challenger of the platform may be satisfied to verify measurements pertaining to rudimentary platform components, such as the boot software, the isolation layer and software components running in an isolated execution environment rather than verify all software running on the platform. This may ease the platform attestation problem in some situations.

However, even assuming the number of platform component integrity measurements that a challenger must verify is limited, problems relating to platform component updates and patching will still arise. Given current software development practices, frequent patching to OS components and applications can be expected to be the norm for the foreseeable future. But even the order in which patches are applied can result in a "combinatorial explosion" of distinct configurations for a single application, each configuration requiring a distinct reference value for attestation purposes. Frequent patching may also lead to problems with respect to sealed data. If an update or patch is applied to a software component to which a key or data is sealed, this key or data must be unsealed and resealed to the updated software component measurements. Failure to reseal to the updated component measurements will result in the key or data being inaccessible after the patch has been applied.

Property Based Attestation [42] has been proposed to address the problem of managing attestation in the presence of a multitude of possible configurations and system updates. This approach introduces an additional layer of indirection into the attestation and sealing processes. Instead of expecting a verifier to determine if a particular set of PCR values represent a trustworthy software state, a platform's state is certified (by a trusted third party) as satisfying certain properties. A platform is then capable of attesting that its current configuration possesses such a property, allowing a verifier to infer whether a platform is trustworthy or not without knowing which particular software is running. Property Based Attestation also allows data or keys to be sealed to properties. As long as the properties of the updated platform configuration match those of the prior configuration, problems related to patching may be reduced. Sadeghi and C. Stüble suggest that this approach could be realised either through a software-based "Trusted Attestation Service" or through modifications to the TPM hardware.

Unfortunately, Property Based Attestation only succeeds in shifting the problems with attestation to an entity other than the verifier, with all of the original problems persisting for the entity that needs to verify a PCR-based attestation. Nevertheless, the number of entities needing to verify such complex attestations could be significantly reduced, and these entities could be given additional resources to enable them to complete their task. Moreover, a software component satisfying a particular property is by no means guaranteed to still satisfy that property after it has been patched, without rerunning the (potentially expensive) evaluation procedure. This evaluation procedure may contribute to the marginalisation of minority platforms by "altering the economics of interoperability" [38]. The cost of establishing that a given platform state matches some desirable property may be so great that only a few well-funded organisations may be able to obtain such a result. Additionally, exactly what properties can be satisfied using such an approach remains an open question. More positively, Property Based Attestation at least shifts the problems to an expert specialising in the particular business of attestation.

A final issue which must be considered with respect to the successful implementation of platform attestation is that of user observable verification. McCune *et al.* [17] describe a scenario in which a user's platform has become infected with malware. Despite the fact that this infection can be detected by an external entity during an attestation process, the external entity has no way of reliably informing the end user that they have failed their attestation. Malware may simply modify the user's display, resulting in the user believing their platform to be in an acceptable state, and, because of this, going on to disclose sensitive information to the malware.

3.4 Backward Compatibility

As a consequence of the piecemeal roll-out of Trusted Computing technologies, current trusted platforms do not come equipped with CRTMs, isolation technologies, processors or chipset extensions. Instead, current trusted platforms include only a TPM meeting the relevant TCG specifications, and, with the exception of Infineon TPMs, do not even include endorsement credentials. To the best of our knowledge, all currently available platforms lack both conformance credentials and platform credentials. This situation has the potential to create an awkward backward compatibility issue as and when fully-deployed TC-PKIs become available. In particular, the absence of these credentials will make it difficult, if not impossible, for a platform to later acquire AIK credentials without operating at reduced assurance levels.

The absence of CRTMs, isolation technologies, processors and chipset extensions from current TPM-enabled platforms makes the use of much of the TPM Trusted Computing functionality described in Section 2 essentially unreliable. Techniques such as sealing and attestation are unworkable if the host platform's state cannot be reliably measured. In order to later enable these features on an already deployed platform, measurement functionality (in the form of a CRTM and modified operating system) would need to be integrated into the platform. This would require the installation of a new OS and the BIOS to be flashed, tasks that would prove difficult for the average user. On the other hand, this may be feasible in a corporate environment with centralised administrative control of platforms. Indeed, in such deployments, legacy hardwares issue may be less of an issue because of more rapid retirement of platforms. Moreover, software-based isolation environments can be provided through the installation of additional software onto already deployed platforms. Nevertheless, hardware-based isolation, enabled through the processor and chipset extensions, cannot be retrofitted to platforms already in the field. As a result, first generation trusted platforms can never be adequately upgraded to provide all the services associated with a trusted platform.

3.5 Usability

Prevailing wisdom suggests that it is prudent to hide the complexities of security technology from end-users [34]. In the past, applications that have relied on a PKI have failed in cases where security functions have been too unwieldy to be usable by non-experts [12]. In one example [48], the PKI experience was considered so painful by some users that they refused to use the technology if it involved handling certificates. The design of suitable user-interfaces that can communicate rich security information whilst remaining usable has historically been very difficult to achieve [61].

By contrast, using a TPM currently requires a detailed understanding of how the underlying technology works. For example, the very act of enabling a TPM prior to its use is a non-trivial task requiring a user to understand and edit BIOS settings. Once enabled, a user is further confronted with setting a TPM owner password, selecting key types fit for purpose, and enrolling certain keys within a PKI. Further problems may arise with respect to password use and management. In addition to setting a password for TPM ownership, unique passwords may also be associated with protected data or keys in a TPM. While the deployment of numerous passwords may be viewed as a sound security decision, management of such passwords so that access is not jeopardised may prove problematic.

These usability issues are a reflection of the general immaturity of Trusted Computing technology and the associated marketplace. Whilst a huge effort has been put into design and specification of technical aspects of Trusted Computing by the TCG, so far less work seems to have been done to address user-centric issues. We may hope for user-friendly configuration and management tools in future, although even these may not be sufficient to make Trusted Computing accessible to the masses.

3.6 Non-Compliance and Inter-operability

Through the provision of a set of open standards, Trusted Computing specifies security interfaces which allow heterogeneous devices to interact. Unfortunately, many of the additional technological building blocks required to instantiate a trusted platformare not standardised, nor does the TCG dictate implementation specifics to its adopters. As a result, a number of currently available TPMs do not comply with the TPM specifications [41]. The current absence of conformance testing facilities implies that the production of non-compliant TPMs may very well continue for the foreseeable future. In turn, discrepancies in implementation between TPM manufacturers may limit future inter-operability of different trusted platforms.

4 Conclusions

Trusted Computing is undoubtedly a powerful technology, with a huge range of possible applications. Nevertheless, there remain a number of significant obstacles to its widespread use, as we have discussed above. Addressing these challenges is therefore a high priority for future research.

Perhaps the most significant of these obstacles is the deployment and management of the PKI necessary to enable general use of the security services supported by Trusted Computing. These issues are in many ways similar to those which prevented the establishment of a global general-purpose PKI. Nevertheless, deploying domain and company-specific PKIs to support Trusted Computing well-defined and limited environments would appear relatively straightforward, since the majority of the problems simply disappear — this again reflects the experience of deploying conventional PKIs, which have been used very successfully in specific domains.

We have also examined problems arising with the use and interpretation of evidence generated using Trusted Computing functionality. This problem arises in particular because of the number of different components (and versions of components). As with the PKI issues, many of the problems are particularly serious when one considers universal use of Trusted Computing — the issues are likely to be much less serious in a closed/managed environment, e.g. as established within a large organisation, notably because the number of components will be significantly less, and there are likely to be more resources available to evaluate the components.

In conclusion, many challenges to the successful large-scale use of Trusted Computing remain. Nevertheless, these challenges are likely to be much less serious for a very important class of users, namely corporate IT. Providing the full benefits of Trusted Computing to the widest possible audience is a major challenge for future research.

References

[1] Internet Key Exchange (IKEv2) Protocol. RFC 4306, 2005.

[2] R. Anderson. Cryptography and Competition Policy: Issues with 'Trusted Computing'. In *Proceedings of the 22nd Annual Symposium on Principles of Distributed Computing (PODC 2003)*, pages 3–10, Boston, Massachusetts, USA, 2003. ACM Press, New York, USA.

[3] R. Anderson. 'Trusted Computing' Frequently Asked Questions - Version 1.1. `http://www.cl.cam.ac.uk/~rja14/tcpa-faq.html`, August 2003.

[4] B. Arbaugh. Improving the TCPA Specification. *IEEE Computer*, 35(8):77–79, August 2002.

[5] B. Balacheff, L. Chen, S. Pearson, D. Plaquin, and G. Proudler. *Trusted Computing Platforms: TCPA Technology in Context.* Prentice Hall, Upper Saddle River, New Jersey, USA, 2003.

[6] B. Balacheff, L. Chen, S. Pearson, G. Proudler, and D. Chan. Computing Platform Security in Cyberspace. *Information Security Technical Report*, 5(1):54–63, 2000.

[7] S. Balfe, E. Gallery, C. J. Mitchell, and K. G. Paterson. Crimeware and Trusted Computing. In M. Jakobsson and Z. Ramzan, editors, *Crimeware.* Addison-Wesley, 2008.

[8] S. Balfe, A. D. Lakhani, and K. G. Paterson. Securing Peer-to-Peer Networks using Trusted Computing. In C. J. Mitchell, editor, *Trusted Computing*, chapter 10, pages 271–298. The Institute of Electrical Engineers (IEE), London, UK, 2005.

[9] S. Balfe and K. G. Paterson. Augmenting Internet-based Card Not Present Transactions with Trusted Computing: An Analysis. Technical Report RHUL-MA-2006-9, Department of Mathematics, Royal Holloway, University of London, London, UK, 2005. `http://www.rhul.ac.uk/mathematics/techreports`.

[10] S. Balfe and K. G. Paterson. e-EMV: Emulating EMV for Internet Payments using Trusted Computing Technology. Technical Report RHUL-MA-2006-10, Department of Mathematics, Royal Holloway, University of London, London, UK, 2006. `http://www.rhul.ac.uk/mathematics/techreports`.

[11] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauery, I. Pratt, and A. Warfield. XEN and the Art of Virtualization. In *Proceedings of the 19th ACM Symposium on Operating Systems Principles (SOSP 2003)*, pages 164–177, Bolton Landing, New York, USA, 19–22 October 2003. ACM Press, New York, USA.

[12] B. Beckles, V. Welch, and J. Basney. Mechanisms for Increasing the Usability of Grid Security. *International Journal of Man-Machine Studies*, 63(1-2):74–101, 2005.

[13] E. Brickell, J. Camenisch, and L. Chen. Direct Anonymous Attestation. In *Proceedings of the 11th ACM Conference on Computer and Communications Security (CCS 2004)*, pages 132–145, Washington DC, USA, 2004. ACM Press, New York, USA.

[14] L. Chen, S. Pearson, and A. Vamvakas. On Enhancing Biometric Authentication with Data Protection. In R. J. Howlett and L. C. Jain, editors, *Proceedings of the 4th International Conference on Knowledge-Based Intelligent Engineering Systems and Allied Technologies*, volume 1, pages 249–252, Brighton, Sussex, UK, 30th August – 1st September 2000. IEEE.

[15] S. Crane. Privacy Preserving Trust Agents. Technical Report HPL-2004-197, HP Labs, Bristol, UK, 11 November 2004.

[16] T. Dierks and C. Allen. The TLS Protocol. RFC 2246, 1999.

[17] J. McCun eand A. Perrig, A. Seshadri, and Leendert van Doorn. Turtles All The Way Down: Research Challenges in User-Based Attestation. In *Proceedings of 2nd USENIX Workshop on Hot Topics in Security (HotSec 2007)*, August 2007.

[18] E. Gallery and C. J. Mitchell. Trusted Mobile Platforms. In A. Aldini and R. Gorrieri, editors, *Foundations of Security Analysis and Design IV (FOSAD 2007)*, volume 4677 of *Lecture Notes in Computer Science*, pages 282–323. Springer-Verlag, Berlin, Germany, September 2007.

[19] E. Gallery and A. Tomlinson. Secure Delivery of Conditional Access Applications to Mobile Receivers. In C. J. Mitchell, editor, *Trusted Computing*, IEE Professional Applications of Computing Series 6, chapter 7, pages 195–238. The Institute of Electrical Engineers (IEE), London, UK, 2005.

[20] T. Garfinkel, M. Rosenblum, and D. Boneh. Flexible OS Support and Applications for Trusted Computing. In *Proceedings of the 9th USENIX Workshop on Hot Topics on Operating Systems (HotOS-IX)*, pages 145–150, Kauai, Hawaii, USA, 18–21 May 2003. USENIX, The Advanced Computing Systems Association, Berkeley, California, USA.

[21] P. Gutmann. PKI: It's Not Dead, Just Resting. *Computer*, 35(8):41–49, 2002.

[22] Intel. LaGrande Technology Architectural Overview. Technical Report 252491-001, Intel Corporation, September 2003.

[23] M. Kinateder and S. Pearson. A Privacy-Enhanced Peer-to-Peer Reputation System. In K. Bauknecht, A. Min Tjoa, and G. Quirchmayr, editors, *Proceedings of the 4th International Conference on E-Commerce and Web Technologies*, volume 2738 of *Lecture Notes in Computer Science*, pages 206–216, Prague, Czech Republic, 2–5 September 2003. Springer-Verlag, Berlin-Heidelberg.

[24] D. Kuhlmann, R. Landfermann, H. Ramasamy, M. Schunter, G. Ramunno, and D. Vernizzi. An Open Trusted Computing Architecture — Secure Virtual Machines Enabling User-Defined Policy Enforcement. www.opentc.net, June 2006.

[25] H. Löhr, H. V. Ramasamy, A-R. Sadeghi, S. Schulz, M. Schunter, and C. Stüble. Enhancing Grid Security Using Trusted Virtualization. In *Proceedings of the 4th International Conference on Autonomic and Trusted Computing (ATC 2007)*, volume 4610 of *Lecture Notes in Computer Science (LNCS)*, pages 372–384, Hong Kong, China, 11–13 July 2007. Springer-Verlag, Berlin-Heidelberg.

[26] W. Mao, F. Yan, and C. Chen. Daonity: Grid Security with Behaviour Conformity from Trusted Computing. In *Proceedings of the 1st ACM*

workshop on Scalable Trusted Computing (STC 2006), pages 43–46, Alexandria, Virginia, USA, 3 November 2006.

[27] M. C. Mont, S. Pearson, and P. Bramhall. Towards Accountable Management of Identity and Privacy: Sticky Policies and Enforceable Tracing Services. In *Proceedings of the 14th International Workshop on Database and Expert Systems Applications (DEXA 2003)*, pages 377–382, Prague, Czech Republic, 1–5 September 2003. IEEE Computer Society.

[28] M. C. Mont, S. Pearson, and P. Bramhall. Towards Accountable Management of Privacy and Identity Information. In E. Snekkenes and D. Gollmann, editors, *Proceedings of the 8th European Symposium on Research in Computer Security (ESORICS 2003)*, volume 2808 of *Lecture Notes in Computer Science*, pages 146–161, Gjøvik, Norway, 13-15 October 2003. Springer-Verlag, Berlin.

[29] A. Pashalidis and C. J. Mitchell. Single Sign-on using Trusted Platforms. In C. Boyd and W. Mao, editors, *Proceedings of the 6th International Conference on Information Security (ISC 2003)*, volume 2851 of *Lecture Notes in Computer Science*, pages 54–68, Bristol, UK, 1–3 October 2003. Springer-Verlag, Berlin-Heidelberg.

[30] S. Pearson. Trusted Agents that Enhance User Privacy by Self-Profiling. Technical Report HPL-2002-196, HP Labs, Bristol, UK, 15 July 2002.

[31] S. Pearson. How Trusted Computers can Enhance for Privacy Preserving Mobile Applications. In *Proceedings of the 1st International IEEE WoWMoM Workshop on Trust, Security and Privacy for Ubiquitous Computing (WOWMOM 2005)*, pages 609–613, Taormina, Sicily, Italy, 13–16 June 2005. IEEE Computer Society, Washington, DC, USA.

[32] M. Peinado, Y. Chen, P. England, and J. Manferdelli. NGSCB: A Trusted Open System. In H. Wang, J. Pieprzyk, and V. Varadharajan, editors, *Proceedings of 9th Australasian Conference on Information Security and Privacy (ACISP 2004)*, volume 3108 of *Lecture Notes in Computer Science (LNCS)*, pages 86–97, Sydney, Austrailia, 13–15 July 2004. Springer-Verlag, Berlin-Heidelberg, Germany.

[33] M. Peinado, P. England, and Y. Chen. An Overview of NGSCB. In C. J. Mitchell, editor, *Trusted Computing*, IEE Professional Applications of Computing Series 6, chapter 7, pages 115–141. The Institute of Electrical Engineers (IEE), London, UK, April 2005.

[34] G. Price. PKI—An Insider's View (Extended Abstract). Technical Report RHUL-MA-2005-8, Department of Mathematics, Royal Holloway, University of London, Surrey, England, UK, June 2005.

[35] A. Pridgen and C. Julien. A Secure Modular Mobile Agent System. In *Proceedings of the 2006 International Workshop on Software Engineering for Large-Scale Multi-Agent Systems (SELMAS 2006)*, pages 67–74, Shanghai, China, 22–23 May 2006. ACM Press, New York, USA.

[36] G. J. Proudler. Concepts of Trusted Computing. In C. J. Mitchell, editor, *Trusted Computing*, IEE Professional Applications of Computing Series 6, chapter 2, pages 11–27. The Institute of Electrical Engineers (IEE), London, UK, April 2005.

[37] J. Reid, J. M. Gonzalez Nieto, and E. Dawson. Privacy and Trusted Computing. In *Proceedings of the 14th International Workshop on Database and Expert Systems Applications (DEXA 2003)*, pages 383–388, Prague, Czech Republic, 1–5 September 2003. IEEE Computer Society.

[38] Electronic Frontier Foundation S. Schoen. Comments on LT Policy on Owner/User Choice and Control 0.8. `http://www.eff.org/Infrastructure/trusted_computing/eff_comments_lt_policy.pdf`, December 2003.

[39] Electronic Frontier Foundation S. Schoen. Give TCPA an Owner Override. `http://www.linuxjournal.com/article/7055`, December 2003.

[40] Electronic Frontier Foundation S. Schoen. Comments on TCG Design, Implementation and Usage Principles 0.95. `http://www.eff.org/Infrastructure/trusted_computing/20041004\_eff\_comments\_tcg\_principles.pdf`, October 2004.

[41] A-R. Sadeghi, M. Selhorst, C. Stüble, C. Wachsmann, and M. Winandy. TCG inside?: a note on TPM specification compliance. In *Proceedings of the 1st ACM workshop on Scalable trusted computing (STC 2006)*, pages 47–56, Alexandria, Virginia, USA, 2006. ACM, New York, NY, USA.

[42] A-R. Sadeghi and C. Stüble. Property-based attestation for computing platforms: caring about properties, not mechanisms. In C.F. Hempelmann, editor, *Proceedings of the 2004 workshop on New security paradigms (NSPW 2004)*, pages 67–77, Nova Scotia, Canada, 2004. ACM, New York, NY, USA.

[43] R. Sandhu and X. Zhang. Peer-to-peer access control architecture using trusted computing technology. In E. Ferrari and G-J. Ahn, editors, *Proceedings of the 10th ACM symposium on Access control models and technologies (SACMAT 2005)*, pages 147–158, 1–3 June 2005.

[44] L. F. G. Sarmenta, M. van Dijk, C. W. O'Donnell, J. Rhodes, and S. Devadas. Virtual monotonic counters and count-limited objects using a TPM without a trusted OS. In *Proceedings of the 1st ACM workshop on Scalable trusted computing (STC 2006)*, pages 47–56, Alexandria, Virginia, USA, 2006. ACM, New York, NY, USA.

[45] S. E. Schechter, R. A. Greenstadt, and M. D. Smith. Trusted Computing, Peer-to-Peer Distribution, and the Economics of Pirated Entertainment. In *Proceedings of the 2nd Annual Workshop on Economics and Information Security*, 2003.

[46] S. Schoen. Trusted Computing: Promise and Risk. Whitepaper, Electonic Frontier Foundation, October 2003.

[47] SETCo. SET Secure Electronic Transaction 1.0 specification — the formal protocol definition. `http://www.setco.org/set_specifications.html`, May 1997.

[48] R. O. Sinnott. Development of Usable Grid Services for the Biomedical Community. In *Useability in e-Science Workshop: An International Workshop on Interrogating Usability Issues in New scientific Practice, within the Lab and within Society (NeSC 2006)*, Edinburgh, Scotland, UK, 26–27 January 2006.

[49] A. Spalka, A. B. Cremers, and H. Langweg. Protecting the Creation of Digital Signatures with Trusted Computing Platform Technology against Attacks by Trojan Horse Programs. In M. Dupuy and P. Paradinas, editors, *Proceedings of the 16th Annual Working Conference on Information Security (IFIP/Sec'01) of Trusted Information: The New Decade Challenge*, volume 193 of *IFIP Conference Proceedings*, pages 403–419, Paris, France, 11–13 June 2001. Kluwer Academic Publishers, Boston, Massachusetts, USA.

[50] E. Sparks. A Security Assessment of Trusted Platform Modules. Technical Report TR-2007-597, Department of Computer Science, Dartmouth, Hanover, New Hampsire, USA, June 2007.

[51] R. Stallman. *Free Software, Free Society: Selected Essays of Richard M. Stallman*, chapter 17 – Can You Trust Your Computer?, pages 115–119. GNU Press, Boston, Massachusetts, USA, 2002.

[52] TCG. Interoperability Specification for Backup and Migration Services. TCG specification version 1.0 revision 1.0, The Trusted Computing Group (TCG), Portland, Oregon, USA, June 2005.

[53] TCG. Subject Key Attestation Evidence Extension. TCG specification version 1.0 revision 7, The Trusted Computing Group (TCG), Portland, Oregon, USA, June 2005.

[54] TCG. TCG Infrastructure Working Group Reference Architecture for Interoperability (Part I). TCG specification version 1.0 revision 1, The Trusted Computing Group (TCG), Portland, Oregon, USA, June 2005.

[55] TCG. TCG PC Client Specific Implementation Specification For Conventional BIOS. TCG specification version 1.20 final, The Trusted Computing Group (TCG), Portland, Oregon, USA, June 2005.

[56] TCG. TPM Main, Part 1: Design Principles. TCG Specification Version 1.2 Revision 94, The Trusted Computing Group (TCG), Portland, Oregon, USA, March 2006.

[57] TCG. TPM Main, Part 2: TPM Data Structures. TCG Specification Version 1.2 Revision 94, The Trusted Computing Group (TCG), Portland, Oregon, USA, March 2006.

[58] TCG. TPM Main, Part 3: Commands. TCG Specification Version 1.2 Revision 94, The Trusted Computing Group (TCG), Portland, Oregon, USA, March 2006.

[59] TCG. TCG Specification Architecture Overview. TCG specification revision 1.4, The Trusted Computing Group (TCG), Portland, Oregon, USA, August 2007.

[60] F. von Lohmann. Meditations on trusted computing. Electronic Frontier Foundation Article, 2003.

[61] A. Whitten and J. D. Tygar. Why Johnny Can't Encrypt: A Usability Evaluation of PGP 5.0. In *Proceedings of the 8th Conference on USENIX Security Symposium (SSYM 1999)*, pages 14–14, Washington, District of Columbia, USA, 1999. USENIX Association, Berkeley, California, USA.

[62] Z. Yan and Z. Cofta. A Method for Trust Sustainability Among Trusted Computing Platforms. In S. Katsikas, J. Lopez, and G. Pernul, editors, *Proceedings of the 1st International Conference on Trust and Privacy in Digital Business (TrustBus 2004)*, volume 3184 of *Lecture Notes in Computer Science (LNCS)*, pages 11–19, Zaragoza, Spain, 30 August–1 September 2004. Springer-Verlag, Berlin-Heidelberg, Germany.

[63] M. Yung. Trusted Computing Platforms: The Good, the Bad, and the Ugly. In R. N. Wright, editor, *Proceedings of the 7th International Conference of Financial Cryptography (FC 2003)*, volume 2742 of *Lecture Notes in Computer Science (LNCS)*, pages 250–254, Guadeloupe, Frence West Indies, 27–30 January 2003.