# Challenges for Trusted Computing

Shane Balfe, Eimear Gallery, Chris J. Mitchell and Kenneth G. Paterson
Information Security Group,
Royal Holloway, University of London
{s.balfe,e.m.gallery,c.mitchell,kenny.paterson}@rhul.ac.uk

April 29, 2008

## 1 Introduction

A trusted platform 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"[1]. 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 sensitive data against software attack. Based on these capabilities, Trusted Computing has been proposed as a means of enhancing the security of numerous applications. For example, it has been promoted as an adjunct to the digital signature process, to enable secure software download, to support secure single sign-on solutions, to secure peer-to-peer networks, to improve the security and privacy of biometric user authentication, to harden mobile devices, and to facilitate identity management. A number of authors have also considered trusting computing's applicability to the agent paradigm, grid security, e-commerce transaction security, and to defend against the ever-growing threat posed by crimeware.

Despite its many potential beneficial applications, Trusted Computing is not without its detractors. Privacy concerns relating to trusted platforms have been raised. The extent to which Trusted Computing could be used to enable and enforce digital rights management, and, more generally, the possible expropriation of platform owner control, are contentious issues. Concerns have also been expressed 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 certain vendors.

[1] www.trustedcomputinggroup.org

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. 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.

## 2 Trusted Computing Technologies

Trusted computing relies on the successful integration and interoperation of a number of technologies:

- Trusted Platform Modules (TPMs), as specified by the TCG, are microcontrollers with cryptographic coprocessor capabilities. A TPM provides a platform with the following features: special purpose registers called Platform Configuration Registers (PCRs) in which information characterising the host platform's configuration can be stored; a means of reporting the platform's current configuration 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.

- A platform's Root of Trust for Measurement (RTM) enables its configuration to be reliably recorded.

- Isolation technologies, such as Microsoft's Next Generation Secure Computing Base (NGSCB) or Citrix's XEN, take advantage of CPU and chipset extensions incorporated in a new generation of processor hardware, including Intel's TXT and AMD's AMD-V. They enable the unhindered execution of software through the provision of assured memory space separation between processes.

The ability to reason about platform behaviour and to protect senstive data are both reliant upon the concept of an *integrity measurement*, namely the cryptographic digest (or hash) of a platform software component[2]. However, in isolation, individual measurements of software components may be of little interest. To reason *effectively* about the behaviour of a particular platform component, the entire sequence of events that culminated in the execution of that component must be measured. For example, during an *authenticated boot process*, initiated by the RTM, a platform's entire configuration is reliably captured and stored. During this process, the integrity of

[2]For example, the integrity measurement of a program could be calculated by computing the cryptographic digest or hash of its instruction sequence, its initial state (i.e. the executable file) and its input.

a pre-defined set of platform components is measured. These measurements are condensed to form a set of *integrity metrics* which are then stored in the TPM's PCRs. These integrity metrics can be communicated to external entities for examination and verification, via a process called *attestation*.

In order to protect sensitive data against software attack, the data can be associated with a set of integrity metrics representing a particular platform configuration and/or a password, and then encrypted. A TPM then ensures that protected data can only be decrypted and released for use if the correct password is input and/or the current configuration of the platform matches the integrity metrics *sealed* with the data. The protection of sensitive data is further enhanced through the deployment of *isolated execution environments*.

# 3 Public Key Infrastructure and Trusted Computing

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 [2, 4], and high-profile system and protocol failures blamed on inappropriate deployment of PKI abound, with SET providing one of the most prominent examples. Put simply, providing a PKI is hard.

The majority of Trusted Computing services depend fundamentally on the deployment and successful inter-operation of certain 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.

For a platform to be considered trusted, it must first obtain the following *core credentials* from an endorsement CA, a platform CA, and one or more conformance CAs, respectively.

**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. The endorsement CA is typically the TPM manufacturer, with the binding taking the form of a digital signature created using a signing key of the manufacturer.

**A platform credential:** A platform credential asserts that a TPM has been correctly incorporated into a design conforming to the TCG specifications. The platform CA is typically the platform manufacturer. In order to create a platform credential, the platform CA must examine

the endorsement credential, the conformance credentials relevant to the trusted platform, and the platform to be certified.

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

Together, CAs of these three types are responsible for issuing the core trusted platform credentials. However, in order to address privacy concerns resulting from routine use of an EK, the TCG introduced the ability for a TPM to generate and use an arbitrary number of pseudonyms, in the form of *Attestation Identity Key* (AIK) pairs. In order for a relying party to have assurance that an AIK represents a trusted platform, a platform must obtain an AIK certificate 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), verifies a trusted platform's core credential set and provides assurance that an AIK is bound to a genuine trusted platform in the form of an *AIK credential*. However, this approach has attracted a certain amount of criticism, as a P-CA is capable of linking all the AIK credentials it issues to a specific platform via the EK, putting the P-CA in a position where it is able to defeat the anonymity protection provided by the use of AIKs.

The second approach, Direct Anonymous Attestation (DAA), was introduced to counteract this criticism. DAA requires a *DAA CA*, which can produce an anonymous *DAA credential* for a trusted platform, which in turn can be used by the platform to sign AIK credentials. Using this approach, trusted platforms can generate and use AIKs which cannot be easily linked to a particular EK by any third party.

Yet another class of CA has been introduced to attest to the usage, mobility and authorisation constraints associated with private keys held by a TPM. A *Subject Key Attestation Evidence (SKAE) CA* is responsible for issuing X.509 certificates which allow a verifier to ascertain that an operation involving a private key can only be performed within a TCG-compliant TPM environment. Such a certificate can be used as a means of coping with difficulties in integrating TPM-controlled keys with standard security protocols. Recently, further PKI-related authorities (notably Migration Authorities and Migration Selection Authorities) have been introduced to address issues with key migration between TPM-enabled platforms.

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 CAs 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 CAs 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 Certification 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 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 [4]. In the absence of CPs and CPSs, implicit cross-certification may exist between CAs, which, as noted in [2], 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, a TPM manufacturer generates the EK credential. However, in *post-manufacturing* generation, 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.

In summary, 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.

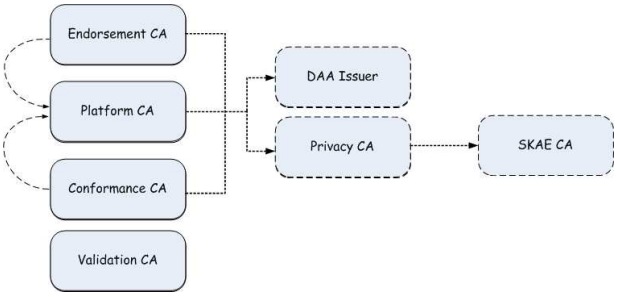


Figure 1: Trusted Computing PKI Components

# 4 Credential 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 certificates 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, by design, a DAA CA cannot link a platform's EK 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 [8] 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 evidence upon which a remote verifier relies to assess a platform. Once this

evidence is destroyed, the PCRs can in theory be repopulated with whatever data the platform owner wishes, allowing the owner to misrepresent their platform's current configuration in a manner that is convincing to a remote verifier. The simple attack of [8] 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 CAs and SKAE CAs. It has been suggested [1] 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, this approach to detection itself 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, this information needs to be globally propagated 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 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 damaging a platform's ability to interact with other parts of the infrastructure. As a result, CAs may be 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 is needed. It is currently unclear what form such a mechanism might take.

# 5 Attestation Evidence Gathering and Verification

The exact parameters to be considered when performing integrity measurements on platform components have yet to be standardised. 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 the elements required by the verifier of a platform component. This is especially true when one considers the complications introduced by software which relies on dynamically linked libraries. 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 increasing. As a result, each PCR will have to store multiple measurements. As the number of platform 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 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 will still arise from platform component updates and patching. Given current software development practices, frequent patching of 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 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 do so will result in the key or data being inaccessible after the patch has been applied.

Property Based Attestation [6] 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.

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. 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. Such an evaluation procedure may contribute to the marginalisation of minority platforms, since the cost of establishing that a given platform state matches some desirable property may be so great that only a few well-funded organisations are able to obtain such a result. Also, 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. With this approach, 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.

A final potential limitation of platform attestation is that of user observable verification. McCune *et al.* [3] 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.

# 6 Backward Compatibility

As a consequence of the piecemeal roll-out of Trusted Computing technologies, current trusted platforms do not come equipped with fully-integrated

RTMs, isolation technologies, processors or chipset extensions. Instead, current trusted platforms include only a TPM 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 RTMs, 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 configuration cannot be reliably measured. In order to later enable these features on an already deployed platform, measurement functionality (in the form of a RTM 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 hardware issues may be less serious because of more rapid retirement of platforms. Moreover, software-based isolation environments can be provided through the installation of additional software on 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.

# 7 Usability

Prevailing wisdom suggests that it is prudent to hide the complexities of security technology from end-users. 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. In one example [7], 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 [9].

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 from 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 the design and specification of technical aspects of Trusted Computing by the TCG, so far it seems that 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.

## 8 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 platform are 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 [5]. 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 between different trusted platforms.

## 9 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 here. 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 in particular, well-defined 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] 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, 25–29 October 2004. ACM Press, New York, NY, USA.

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

[3] J. McCune, A. Perrig, A. Seshadri, and L. van Doorn. Turtles All The Way Down: Research Challenges in User-Based Attestation. In *Proceedings of the 2nd USENIX Workshop on Hot Topics in Security (HotSec 2007)*, Boston, MA, USA, 6–10 August 2007. USENIX Association, Berkeley, CA, USA.

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

[5] 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, VA, USA, 3 November 2006. ACM, New York, NY, USA.

[6] 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, 20–23 September 2004. ACM, New York, NY, USA.

[7] 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.

[8] E. Sparks. A Security Assessment of Trusted Platform Modules. Technical Report TR-2007-597, Department of Computer Science, Dartmouth College, June 2007.

[9] 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 (USENIX 1999)*, pages 169–183, Washington DC, USA, 23–26 August 1999. USENIX Association, Berkeley, CA, USA.