Daonity — Grid Security with Behavior Conformity from Trusted Computing

The Daonity Team*

May 6, 2006

Abstract

A central security requirement for Grid computing, or more generally federated computing, can be referred to as behavior conformity. This is a strong assurance for the system that a remote principal (user, computing platform or instrument) will be acting in conformity with the rules defined by the policies of the federated computing. However, as will be analyzed and discussed in this paper, Grid security practice at present, e.g., Grid Security Infrastructure (GSI) for a standard Grid middleware Globus Toolkit, has little means for this requirement to be met and consequently falls short of satisfactory solutions to a number of Grid computing problems.

Trusted Computing (TC) technology developed by Trusted Computing Group (TCG) forms an important industrial initiative for improving computer security by means of a hardware supported security architecture. For a federated computing system, the TC technology can not only improve security in a conventional sense (such as stronger protection on cryptographic key material), but also allow conformed behavior of principal(s) in a remote environment to be measured by the rest of the confederation. We consider that the TC technology can provide practical and readily available solutions to meeting behavior conformity requirements needed by Grid computing.

In the main part of this paper we report Daonity system. This is a TC-technology enabled Grid security system which we have designed for improving GSI. We shall see a number of TC innovations applicable to GSI. These include: (i) security suitable for constructing a dynamic virtual organization of an unbounded resource supply, (ii) construction of property based virtual organization with conformed quality of services, (iii) supporting sharing of security resource, and (iv) stronger protection of the Grid authorization mechanism.

Keywords Grid Computing, Grid Security Infrastructure (GSI), Behavior Conformity, Trusted Computing (TC), Virtual Organization (VO), Conformed Quality of Services, Platform Attestation.

1 Introduction

A computational Grid [4, 6, 8] is a distributed computing system comprising a number — possibly large — of physically separated resources, each subject to their own various security, management and usage policies, to combine to a federated computing environment called **virtual organization (VO)**. The name "Grid" follows analogously from tapping electricity supplied by the power grid, meaning that computational resources also can and should be tapped from super computers and data centers *elsewhere*, instead of using local ones only. Early computational Grids were more or less confined to high performance computing applications in which a Grid VO comprises of one user plus a number of computational and/or data storage resource providers. Grid computing has now evolved to a more general model of federated

*A Research Group in Global Grid Forum led by HP Labs China and participated by Huazhong University of Science and Technology, Wuhan University and Oxford University.

computing which supports not only resource and data sharing for high performance computing applications, but also science collaboration. In the general federated computing model, a VO of principals who are (may be plural number of) users, computing platforms or devices may be working on a number of common tasks and therefore having similar requirements on resource and device utilities.

1.1 Grid Security: Threat Model and Limitations of the Existing Solutions

In the general setting of a Grid VO, principals are physically separated in different trust and management domains which can span governmental, industrial and academic organizations. Moreover, these principals are loosely and ad hoc related to one another. This is because (i) a VO does not have a reliable control over a principal as a real organization does over its employees and assets, (ii) these principals need not to maintain a responsible relationship to one another as they should in a real organization or in a longterm social relationship, and (iii) a VO may come up into being, grow, diminish or terminate dynamically.

Despite these loose, ad hoc and dynamic properties, Grid computing still needs security services in a rather strong way. In addition to usual security services for conventional distributed computing which protect mainly owned or organizationally controlled resources and assets against external adversaries, a principal in Grid computing also has interest needing protection on platforms which are out of the principal's ownership or organizational control, and often against the very owners of these platforms. Here are a few typical Grid security problems:

Security for Grid user Most grid applications entail code written in one place being executed in another. A resource provider should not be able to compromise guest user's security. For example, an algorithm running on a resource provider for a guest user may need to provide protection on the user's input data to it and the computed result returned from it, in terms of both confidentiality and integrity. The protection may need to be strong enough against even a privileged entity at the resource provider, such as the owner or system administrators.

Security for Grid resource provider A user should not be able to compromise security policies at a resource provider. For example, a VO principal must not be able to disseminate certain data received or as computed result returned from a resource provider to any other principal outside the VO. The protection may need to be strong enough against a collusion among a group of VO users or even with a privileged entity at the resource provider.

Conformable VO policy Even an ad-hoc constructed Grid VO may need to have a specifiable and executable policy. For example, a user, upon starting the construction of a VO, may want to demand that resource providers joining the VO satisfy certain criteria in order to maintain quality of services. The quality of services can be in terms of the configurations of hardware and/or software systems. The point (and the difficulty) here is *comformability* of the policy against the loose and ad-hoc nature of the VO.

Auditability Any misuse of resource by users, and any compromise to users' data and computations possibly by a system administrator at a resource provider, must be detected in an undeniable manner.

Thus, to protect interest on a platform which is faraway and beyond organizational control is a distinct nature of Grid security. We can summarize in the following the threat model in Grid security:

> **Threat Model for Grid Security** A principal has interest in a foreign platform whose owner is both a partner and a potential adversary.

We may use "partner-and-adversary" to name this threat model of Grid security. This threat model makes Grid security a difficult problem.

However, available and mainstream security mechanisms for Grid security, in fact, those supported by Grid Security Infrastructure (GSI) [7] for a standard Grid middleware Globus Toolkit [12], are essentially based on a conventional trust model of reputation and social relationship maintenance. The trust model of GSI is a direct application of the standard public-key authentication infrastructure (PKI): an unknown principal will be deemed trustworthy if it is introduced by a reputable and known trusted third party. It is hoped that the introduced principal will behave in a responsible manner since in this trust model it is assumed that a principal will try its best effort to honor the introduction of the trusted third party.

We shall argue that the conventional trust model of "best-effort for reputation maintenance" is inadequate for Grid security requirements which face the partner-and-adversary threat model. As will be analyzed and discussed in §2, the existing Grid security mechanisms in GSI as straightforward applications of public-key authentication infrastructure fall short of satisfactory solutions to a number of Grid computing problems.

1.2 Behavior Conformity from Trusted Computing

We consider that what Grid security needs in trust model should be one stronger than that of "best-effort for reputation maintenance." The stronger trust model can be named **behavior conformity**. In this trust model a principal can be imposed into complying with a conformed behavior desired by an application. Note that such principals can be remote to one another or ad-hoc related. Our approach to realizing behavior conformity for improving Grid security is to let a remote principal in a Grid VO demonstrate an incapability to violate the rules defined by the security policies of the VO.

We consider that **Trusted Computing (TC)** technology [17, 24, 25] developed by Trusted Computing Group (TCG) forms a practical and readily available technical means to serving our purpose. TC is an important industrial initiative for improving computer security by means of a hardware supported security architecture. For a federated computing system, the TC technology with a tamper-protection hardware module can not only improve security in a conventional sense, such as strong protection on cryptographic key material, but also allow a conformed behavior of a remote principal to be measured by the rest of the confederated computing system. In specific, not only can the TC technology provide practical solutions to the partner-and-adversary security problems which we have listed in §1.1, but also the TC solutions should be particularly suitable for the Grid environment of loosely and ad hoc related principals.

1.3 Daonity

Daonity is a Global Grid Forum (GGF) project [26] attempting strengthening GSI using the TC technology. We have completed system design, specification and an early round of implementation. As a TC-technology enabled Grid security system, Daonity not only implements a hardware based key protection for GSI, but more importantly also attempts at adding behavior conformity to GSI. We shall report a number of TC innovations as improvements to GSI from the work of Daonity. These include: (i) security suitable for constructing a dynamic virtual organization of an unbounded resource supply, (ii) construction of property based virtual organization with conformed quality of services, (iii) supporting sharing of security resource, and (iv) stronger protection of the Grid authorization mechanism.

1.4 Organization of the Paper

The remainder of this paper is organized as follows. In §2 we provide a background review of the mainstream Grid security mechanisms — GSI. We shall see how the current GSI has to offer solutions to a number of Grid security problems, and what its inadequacies are. In §3 we overview the Trusted Computing technology. In §4 we present the Daonity system, and discuss how it provides needed solutions to the identified problems in GSI. In §5 we discuss issues of the TC implementation and deployment. In §6

we review related work. In §7 we discuss and make clarification on a widely spread negative view on the TC technology. We finally conclude in §8.

2 Grid Security Infrastructure

Grid Security Infrastructure (GSI) [7, 11] is a security architecture and kernel for the Grid middleware Globus Toolkit (GT). In this section we review GSI with focus on its use in the most recent version of GT (Version 4 or GT4) [12].

2.1 An Abstract Description of Grid Security Solutions at Present

In GT4, security is composed of both Web-Services (WS) based and non-WS based elements. The latter, which is based on transport-level security using TLS protocols [9], remains the mainstream default methodology since it has a performance advantage at present.

For entity authentication, message confidentiality and data integrity, GSI includes security mechanisms for both client-side and resource-side. The security mechanisms are based on applications of the standard public-key authentication infrastructure (PKI) [13].

In the client side, GSI security mechanisms include facilities to create and maintain an X.509 identity credential and certificate [13] for identifying a user on the client, and a temporary credential/certificate for the client to act as the user's proxy. The X.509 identity credential/certificate are also the bases in the resource side for identifying the resource.

In the resource side, GSI also allows a resource provider to create and maintain a temporary credential/certificate, called a **proxy credential/certificate**. The notion of proxy credential is a result of careful design by GSI. Not only can the design achieve a practical means to protecting user's credential in a Grid VO environment of rights delegation, but also it works well with a single sign-on service.

Using so organized credentials, mutual authentication can be achieved between a client (on behalf of the user) and a resource provider. They can thereby establish a secure communication channel by applying the Transport Layer Security (TLS) protocol, including the standard SSL Authentication Protocol (SAP) suite [9]. Through the established secure channel the resource provider can also obtain a proxy credential from the client for delegation use. We shall review GSI's proxy delegation technique in detail in §2.3.

In GT4, GSI also supports message-level security which implements security services from WS-Security and WS-Secure Conversation specifications. WS Security [3] is potentially much more flexible, and in principle should be more efficient (since only selected elements of the communication are encrypted) — though present implementations has not shown any evidence for efficiency. WS Security takes a message level security approach by performing encryption at the Web Services layer, such as the XML messages. These solutions also make use of X.509 PKI. It is envisioned that the non-WS Security methodology will eventually be deprecated as message-level security eventually will have an acceptable performance, but that is not expected to happen or likely to happen in any near future.

2.2 Our Focus on Grid Security Problems

We consider that Grid computing should have the following two important security requirements:

Behavior conformity Principals in a Grid VO must behave in conformity with the VO's security policy. This is a distilled description of the security requirements under the partner-and-adversary threat model which we have discussed in §1.1.

Secure delegation of user's resource request In a Grid computing task a user may need an un-pre-

determinable amount of computational, storage and instrumental resources which are not local to the user. A resource request from the user should be automatically executed by a resource provider through a proper delegation of the requesting user. Secure delegation means to make a mis-delegation a very difficult, if not impossible, problem.

By focusing on these two requirements it is our intention to improve GSI in these two aspects because we believe it is in these two aspects GSI is weak.

We notice that GSI does already have well-explored solutions to a number of other Grid security requirements [7]; those requirements are named: "unattended user authentication," "single sign-on," "uniform credentials/certification infrastructure," "interoperability with local security solutions," "exportability," "support for secure group communication," and "support for multiple implementations." The reader is referred to [7] for detailed descriptions of these requirements.

While we believe that GSI has not considered "behavior conformity" as a needed security service for Grid computing, GSI does have a thought design for dealing with "secure delegation of user's resource request." Let us now review this design feature of GSI in detail.

2.3 Delegation of User's Resource Allocation Request

In a Grid computing task a user may need an un-pre-determinable amount of computational, storage and instrumental resources which are not locally available to the user. The resource request of the user should be automatically executed by a resource provider through a proper delegation of the requesting user. In GSI, the idea of proxy is to achieve an automatic delegation of resource request. Figure 1 illustrates the resource request delegation mechanism of GSI. It is in fact the structure of the main form of a Grid VO in GT4 [11].

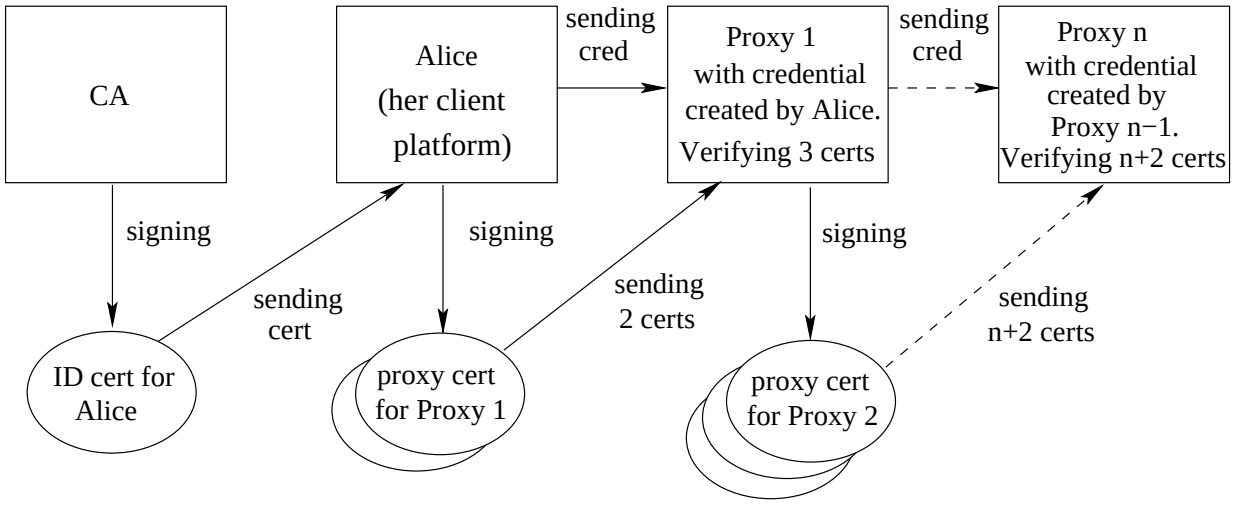

Figure 1: Grid VO Construction in GSI: Using Proxy Credentials/Certificates

In this form of VO creation, the user Alice has an identity certificate issued by CA. Upon her request for ("tapping") external computational resource, the GT middleware will find for her a resource provider which is denoted Proxy 1 in Figure 1. Like Alice, Proxy 1 also has an identity certificate issued by a CA (not necessarily the same CA serving Alice). We should note that Figure 1 is already a simplified case in two senses. First, it doesn't depict the communications of CA's certificate, nor the identity certificates of each Proxy servers. Secondly, we have omitted an distinction between Alice's identity certificate and her client platform's (proxy) certificate. In reality, Alice should create a pair of proxy credential/certificate

for her client platform so that the latter can act on Alice's behalf. This is necessary since Alice may need to be away from the client. This is called **un-attended user authentication** property of GSI [7]. In our simplification, we let Alice's identity certificate play the same role of her platform certificate. Otherwise, Alice (Proxy 1) should have sent three (verified four) certificates, instead of two (three) as shown in Figure 1.

In the general setting, Alice and Proxy 1 do not maintain a long term relationship and hence they do not know each other. Nevertheless, using their respective identity certificates these two parties can conduct mutual entity authentication to establish a secure channel between them. Through the secure channel, Alice could be securely served in various security services, such as data confidentiality and integrity. Alice shall also construct a new cryptographic credential (a private key), called "proxy credential." The public part of the proxy credential will be certified by Alice to be verifiable using her ID certificate, and the result the certification is a "proxy certificate" of Alice. Via the secure channel, Alice can send her proxy credential, i.e., the private key matching the public key in her proxy certificate, securely to Proxy 1. We now explain the use of this proxy credential/certificate.

If Proxy 1 can already satisfy Alice's resource requirement, then this VO is just in a trivial case with Alice and Proxy 1. In the general case, the VO is larger. Then Proxy 1 could request additional resources on Alice's behalf. This is why the resource provider is named a proxy, meaning a proxy working for Alice. But here is a question; why not Proxy 1 simply informs Alice and lets her sort out her own problem of resource allocation? The GT security for VO construction in Figure 1 actually has a very good design feature; it can achieve an important goal of Grid computing: ease of use by the user. Alice should only do the simplest things: submit the Grid job to Proxy 1 and go! It is assumed that Alice's client platform has a quite limited computational power and therefore it should not be used as the center of responsibility for the complex process of resource allocation. The complex job should better be done in streamline by a powerful principal such as a resource provider.

Now suppose Proxy 1 cannot meet Alice's need. It will then have to allocate further resources on Alice's behalf. The proxy credential/certificate which Alice has created/certified for use by Proxy 1 will now allow the latter to properly delegate Alice's resource allocation. Suppose that a Proxy 2 is found to be able to provide additional resources. Then the same protocol which has run between Alice's client and Proxy 1 can now run between Proxy 1 and Proxy 2. And likewise for Proxy 2 and a Proxy 3, and so on, ..., until this VO construction eventually completes at a Proxy n, when Alice's resource need is satisfied.

It is clear that a VO constructed this way is not only done in a streamline automatic fashion, the size of the VO can also be unbounded. Therefore, this way of constructing a Grid VO is regarded scalable. The potential of building a VO with unbounded resources is the very key motivation for Grid computing.

2.3.1 Limitations

As clearly illustrated in Figure 1, each subsequent proxy must verify an increasing number of certificates in order to be sured the genuineness of the resource allocation request from Alice. We note that this is not only a performance disadvantage, it also creates a high system complexity. The latter problem is explained below.

For the VO building to be automatically done, each proxy credential, including that of Alice (in her platform) is stored in the file space of each platform. The protection of such a credential is up to the level of file protection in the operating system. We know this is a weak protection for a cryptographic key. To mitigate the potential danger of key compromise, GSI stipulates that a proxy credential/certificate should have a short lifetime, the default value is 12 hours. Thus, if Alice's Grid job is long, then she will have to come back to her platform to maintain the liveness of her VO.

Another drawback of so constructed VO is that it does not support a conformable means for VO policy enforcement. Namely, Alice can have little control on the quality of services that the joining

proxies could provide. A control on the quality of services can be that a resource provider meets certain criteria, e.g., has a desired system configuration (fast enough CPU, large enough storage, strong enough mechanisms for data confidentiality/integrity protection, etc.), to join the VO for the resultant VO to have a desired QoS property. With the chained construction of the VO illustrated in Figure 1, all Alice could have in VO policy enforcement is a hope that a resource provider will be acting in the best effort to have the VO's policy maintained.

3 Trusted Computing

In recent years, increased reliance on computer security and the unfortunate fact of lack of it, particularly in the open-architecture computing platforms, have motivated many efforts made by the computing industry. Among these is the development of **Trusted Computing (TC)**. In 1999 five companies — Compaq, HP, IBM, Intel and Microsoft — founded **Trusted Computing Platform Alliance (TCPA)**. The motivation of TCPA was to add trust to open-architecture computing platforms. In 2003 TCPA achieved a membership of 190 plus companies, when it was incorporated to **Trusted Computing Group (TCG)** [25, 17]. TCG is a vendor-neutral and not-for-profit organization for defining, specifying and promoting industrial standards for the TC technology. The TCG work has so far been developed to contain sufficient innovations and become a standard methodology for adding trust and security to open computing platforms.

TCG's approach to adding trust is to integrate to a computer platform a hardware module called **Trusted Platform Module (TPM)**. TPM has a tamper-protection property. It is intended by TCG that TPM can play the role of an in-platform trusted third party agent to enforce a conformed behavior for software systems running on the platform. TPM must be trusted to function properly as it is designed. Trust in TPM's correct functionality is underpinned by a number of elements. As an industrial standard body, TCG considers that this notion of trust is materialized by the following elements.

- The tamper protection assumption of the TPM that the behavior of any of its inner component cannot be subverted by any external principal.
- Open specifications of the design for the hardware, software, firmware components, algorithms and protocols, which are used by the TCG. The open specifications facilitate expert review for minimizing the possibility of design errors.
- Standard processes and criteria for evaluation and certification of the system. TCG stipulates that evidence of engineering practice and industry review follow the Common Criteria (CC) [5] certification results.
- Good engineering practices by the manufacturer, with standard approach to defining, guiding and industry review of manufacturing processes.

We believe that the above mechanism of trust is reasonable for establishing and maintaining a reliable behavior to be expected from a TPM. However, we shall report in §7 a widely publicized view which controverts whether the TCG notion of trust should actually be called trust at all [1]. Although it was expressed in the guise of arguing the meaning of trust, that view is in fact a concern on whether the TCG technology may be misused, in particular by a few large companies, to stifle competitions. Given the fact that a TPM can indeed play the role of a in-platform agent with a conformed behavior which is designed out of control of the platform owner, the concern of technology misuse is an understandable one.

At any rate, we are fortunate to be able to avoid the issue of technology misuse. In this paper when we speak of trust, we confine ourselves to the idea of a platform system having an expected behavior

supporting Grid or federated computing applications in which principals accept that they should comply with a commonly agreed behavior. In our Grid computing model, conformed behavior is a requirement rather than a problem, and hence misuse of trust is not an issue.

3.1 TC Working Principle

The following four notions are at the core of the TC technology.

3.1.1 Trusted Platform Module (TPM) — an In-platform Trusted Third Party

This is a tamper-protection hardware module uniquely integrated to a platform. The tamper protection is in the following two senses:

Shielded locations These are places (memory, register, etc) which have the hardware base inside the TPM and may be extended to a place outside the TPM via the supporting software system. A shielded location holds information which cannot be accessed by any external principal in any way to violate data confidentiality. The information held in a shielded location can only be used in the TCG designed ways by "protected capabilities" (see below). Information protected by shielded locations includes cryptographic keys and some system integrity metrics which are held by a number of hardware registers inside the TPM.

Protected capabilities These are designated computations and operations performed by the TPM which have exclusive permission to access shielded locations. Because of the need for accessing shielded locations which are not available to any principal external to the TPM, a protected capability cannot be controlled or subverted by any of such principals in any non-pre-designed manner. Protected capabilities include: cryptographic operations inside the TPM such as the generation of random numbers and cryptographic keys, encryption, decryption and digital signature generation using keys in shielded locations; TPM functional operations which are related to measuring, storing and reporting of system metrics (see below), and TPM's system operations which maintain power detection, counters, etc.

The tamper protection is an assumption. Under this assumption, any principal external to the TPM is considered a potential adversary, and this even includes the owner of the TPM. This assumption is an important concept in TCG. Using the tamper protection assumption it is intended that a TPM can indeed play the role of an in-platform trusted third party to protect some important data and perform some designated computations and functions.

It is actually reasonable to believe the validity of the tamper protection assumption. The validity rests on an inequality between the cost of making a hardware chip with a tamper-protection quality and that of subverting it. This is somewhat analogous to the following fact in cryptography: designing a hard problem using an NP witness element is easier than solving the problem without the witness.

The integration between a TPM and a platform is also tamper protected. We note that this property will be important in our application of TC to realize a conformed policy for a VO.

3.1.2 Root of Trust for Measurement (RTM)

The simplest form of platform measurement is the hardware configuration properties of the platform in which a TPM is integrated in a tamper-protection manner. A certificate of hardware configuration can be issued by a trusted third party to the TPM-platform *after* the integration of the TPM into the platform. The signing capability of the TPM can later report the platform's hardware configuration status to a remote querier (see ROR in §3.1.4). In this simple form of RTM, the TTP is trusted (by the remote querier) that it will not issue the certificate of hardware configuration if the platform does not have the configuration specified in the certificate.

A more advanced form of platform measurement is on its software configuration properties. At the platform boot time, the TPM measures the system's data integrity status. The measurement starts from the integrity of BIOS, then that of OS and finally to applications. Note that the lowest part of the program code of the RTM is also called the **core RTM (CRTM)** which is a firmware implemented instruction code programmed to measure the very first piece of software system a platform will be running. With CRTM, it is in principle possible to establish a desired platform environment by loading only well behaved systems. Although a practical realization is so far still beyond commercial use. New thoughts on virtualization of operating systems have been proposed, e.g., [10, 16], to be a way round the problem.

3.1.3 Root of Trust for Storage (RTS)

The RTS is a computing mechanism which realizes TPM-enabled shielded locations in such a manner that they are able to hold information of a size not bounded to the (usually small) size of the TPM. There are two ways to achieve the RTS for two different services, respectively. One of the storage service is for storing information which requires confidentiality protection and the RTS locations for this use are usually outside the TPM in an external persistent storage (e.g., a hard disk drive). The other is for storing the RTM integrity metrics and the RTS locations for this use are usually inside the TPM.

The RTS location for storing confidentiality data is a straightforward application of the well-known cryptographic key management technique. At the TPM initialization time, the owner of the TPM creates a **storage root key (SRK)** in its shielded locations. The SRK a public/private key pair with the private key residing in a shielded location inside the TPM. Now consider a tree-structured hierarchy of key management system in which the SRK is in the root of the tree. The public key of the SRK is used to encrypt a plural number of children "blobs." Each of the children blobs can be a "wrapped key blob" (encrypted key) or a "wrapped data blob." A key in a wrapped key blob can be a symmetric key, a private key for decryption or signature generation uses. A symmetric key in a key blob can further encrypt a plural number of children blobs, and the same structure of the hierarchy is maintained downwards, ...

The RTS location for storing the RTM integrity metrics of the system software is a plural number of registers inside the TPM called **Platform Configuration Register (PCR)**, A PCR is a 160-bit hardware register which is implemented in volatile storage. Upon either system startup or the system reset event, a PCR is always reset to the initial state with a default NULL value. Each RTM result of system software measurement, which is a hash function evaluation of the system image, is then recorded in the PCR in a hash total formulation. The hash total formula permits a PCR to cumulatively record the RTM integrity metric results in an unbounded fashion. The stored platform environment status is maintained until system reboot.

Related to these two RTS services, in TCG there is a notion of "migratable TPM information" (**TPM migration**). The TPM migration mechanism allows an authorized user of a TPM to securely transfer a secret in a shielded location of the TPM (the one the user is authorized to use) to a shielded location of another TPM of the user's choice. This mechanism permits a user to move her/his cryptographic credentials and/or mission-critical data from one trusted platform to another. This is a necessary security service for, e.g., a situation when the user changes platforms. While some user credentials and data secured in the RTS under the key management system of the SRK tree may be rendered migratable, information in the RTS for storing system integrity metrics is non-migratable.

3.1.4 Root of Trust for Reporting (RTR) and Remote Platform Attestation

The TPM can report to a remote requester a RTM result regarding an executable running in the platform system. Here, the RTM result has been recorded in an RTS protected location. This service is achieved using a well-known cryptographic protocol technique of challenge-response: a remote querier challenges

the TPM-platform with a random number, and the TPM-platform responses by the TPM signing the random challenge. The protocol enabling this TCG feature is called **remote platform attestation protocol**.

Remote platform attestation is a very innovative part in TCG. It is highly relevant to secure collaborated computing using the TCG technology. With remote platform attestation, a remote principal as a stakeholder in a collaborated computing job can be assured of a conformed behavior of a platform which is required by the secure collaborated computing application. Under the tamper protection assumption of the TPM, the remote stakeholder knows that the conformed platform behavior cannot be easily subverted, not even by the platform owner.

3.2 Daonity's Applications of the Trusted Computing Technology

The behavior conformity property which we shall add to GSI as contributions from the Daonity Project can be achieved using only a few very basic TC enabled security services. These are:

1) An RTS service for secure storage of a Grid user's cryptographic credential. This is to improve from GSI's weak and file-system based protection for proxy credentials (temporary cryptographic keys) to a hardware based one. This will be presented in §4.1 and §4.2.

2) A combination of RTM-RTR and key-migration services. First, the simple form of the RTM service which we have described in §3.1.2 will be realized. This allows each resource provider to have a property certificate issued by a trusted third party. The RTR service which we have described in §3.1.4 will then allow the RTM information to be reported to a remote querier. Upon satisfying of the RTR result, a key migration service will allow the Grid user's cryptographic credentials to be migrated to the TPM of the attested platform. In this way, Daonity can construct a VO of a conformed VO policy with desired hardware configurations. This will be presented in §4.3 and §4.4.

3) A simple form of RTM-RTR services for attesting a specific software system. Here, the RTM is to record Grid authorization data, and the RTR is for the recorded data to be audited. This forms a security strengthen to the Grid authorization mechanism in GSI. This will be presented in §4.5.

We are now ready to present the Daonity system.

4 Trusted Computing for Grid Security

We believe that the TC technology can offer good solutions to Grid security problems for which GSI at present seems to have played little role.

4.1 Strong Protection of User's Cryptographic Credential

Unattended user authentication is an important feature in the Grid. This means that a user working in a VO is mainly doing so via their proxy. Work within a VO may involve dynamic sessions of resource allocation and hence require user entity authentication without having the user present.

In GSI, this is achieved by having a user client platform be issued a proxy certificate. The cryptographic credential of this certificate (i.e., the private key matching the public key in this certificate) is simply stored in the file system of the platform protected under the access control of the operating system. In this way, the client platform does not need to prompt the user for cryptographic operations. The obvious danger of leaving a private key in the file space is mitigated by stipulating a short lifetime for the proxy certificate. The default lifetime of a proxy certificate in the GSI is 12 hours. Upon expiration, a new proxy certificate must be re-issued. We feel this is an unacceptable security exposure.

In TC, with a platform containing a TPM with the tamper-protection property, it is natural to store a user's cryptographic credentials in the TPM, or under an RTS location protected by the TPM. In specific, we consider that Alice in Figure 1 possesses a TC equipped platform and can store the private key of her identity certificate inside the TPM.

This use of the TPM is possible even if Alice's platform is a shared one. In TC, each user of a platform can be assigned an RTS location which is exclusively usable by the user using a password protection. Note that, because Alice's password is also protected in the TPM, the Grid middleware serving Alice can be initiated to run with the TPM system, and hence, Alice only needs to input her password once to authorize the Grid middleware to run on Alice's behalf, even long after Alice is away from the platform.

We should notice the most basic behavior conformity property of the TPM: prohibition of even the owner of the TPM from accessing certain protected data. Thus, in the case of the platform which Alice uses is owned by another principal, the owner, even in the role of a system administrator, cannot access Alice's cryptographic credential, not her password, which are protected in the TPM, due to the tamper protection property of the TPM.

4.2 Sharing of Security Resource

GSI makes use of MyProxy on-line credential management services to achieve single sign-on solution to users who do not have public-key cryptographic capacity [15]. These users include, e.g., a roaming professional using a foreign platform to obtain Grid services ubiquitously. The idea is to use an online credential repository which can deliver temporary Grid credentials with certified proxy certificates to the end users upon request. This is achieved via simple user authentication mechanisms such as password based. Let Alice share a password with MyProxy. Whenever and wherever Alice requests for a cryptographic credential by authenticating herself to MyProxy, the server will generate a proxy credential and certificate for Alice. The certificate is sent to Alice, with the private key encrypted using the shared password. From now on, Alice could use the proxy credential/certificate as her security basis. Note, provided Alice could link to MyProxy, this on-line credential service is ubiquitous whenever Alice is.

Daonity shall retain, and in fact strengthen, this nice feature of ubiquitous usability of Grid services. Suppose Alice's platform does not have a TPM, and Alice is a MyProxy user (this means she shares a password with MyProxy for on-line credential service purpose). Let Bob's platform has a TPM. Upon Bob's permit, Alice could obtain a public key, $PubKey_{\mathrm{Bob's\ TPM}}$, of Bob's TPM, and sends it to MyProxy when she request the an on-line credential service. MyProxy server can encrypt the proxy credential for Alice under the public key $PubKey_{\mathrm{Bob's\ TPM}}$ and make the encrypted credential usable only by Alice using the password. Bob, the owner of the TPM, if trying to gain an access to Alice's proxy credential, must at least lunch a password attack on his TPM. TCG has well designed means against such attacks.

In this way, TC's behavior conformity property enables a secure remote use of Bob's TPM as a piece of shared security resource. We notice that, although TPM may not become everywhere available very soon, this use of the TPM as a shared resource in Grid security can indeed happen within a short period of time as a result of the Daonity work.

It is insightful to view Bob's platform in the position of Proxy 1 in Figure 1. Now Proxy 1 is playing the role of a resource provider for security services.

We know that it will take some time to gradually get near to a point of a ubiquitous deployment of the TC technology. However, with Daonity enabling to use the TPM as a shared security resource, the TC strengthened Grid security can take place much sooner.

4.3 Secure Delegation of User's Resource Request

Now let's return to the main problem of Grid VO construction in GT4's use of GSI, which we have reviewed in §2.3 and illustrated in Figure 1. We have discussed in §2.3.1 a number of limitations with this method of Grid VO construction, which we summarize as follows:

Limitation 1 It has a potentially high computational complexity for building a VO since the number of proxy certificates to be sent and verified grows as the size of the VO does.

Limitation 2 It has a high system complexity for maintaining a VO due to the need of renewal proxy credentials/certificates which are stipulated short-lived because of a weak protection for them.

Limitation 3 A so constructed Grid VO does not support a user to define and enforce a policy; a needed policy can only be maintained by the best effort of the participants.

Now TC can offer nice solutions to these problems. Let us first consider Limitations 1 and 2. Suppose the proxies are TPM equipped platforms. Alice, when starting to build the VO, can act as a "migration authority," to have her cryptographic credential rendered migratable. TCG permits the owner of an RTS system protected by a TPM to migrate the RTS system so that it is to be protected by another TPM. Of course, the migration needs to be done with care, and we shall look after that part as our solution to Limitation 3 which will be presented in §4.4.

Suppose for the moment that the need care is taken, and Alice's cryptographic credential can be migrated her TPM to the TPM of Proxy 1 in Figure 2. If the VO further needs a Proxy 2, then Alice's credential should further be migrated from the TPM of Proxy 1 to the TPM of Proxy 2, and likewise for the further subsequent joining proxies. In specific, Alice should create a proxy credential/certificate, and only make that proxy credential migratable. She should send her identity certificate and the proxy certificate to Proxy 1 for it to act on her behalf.

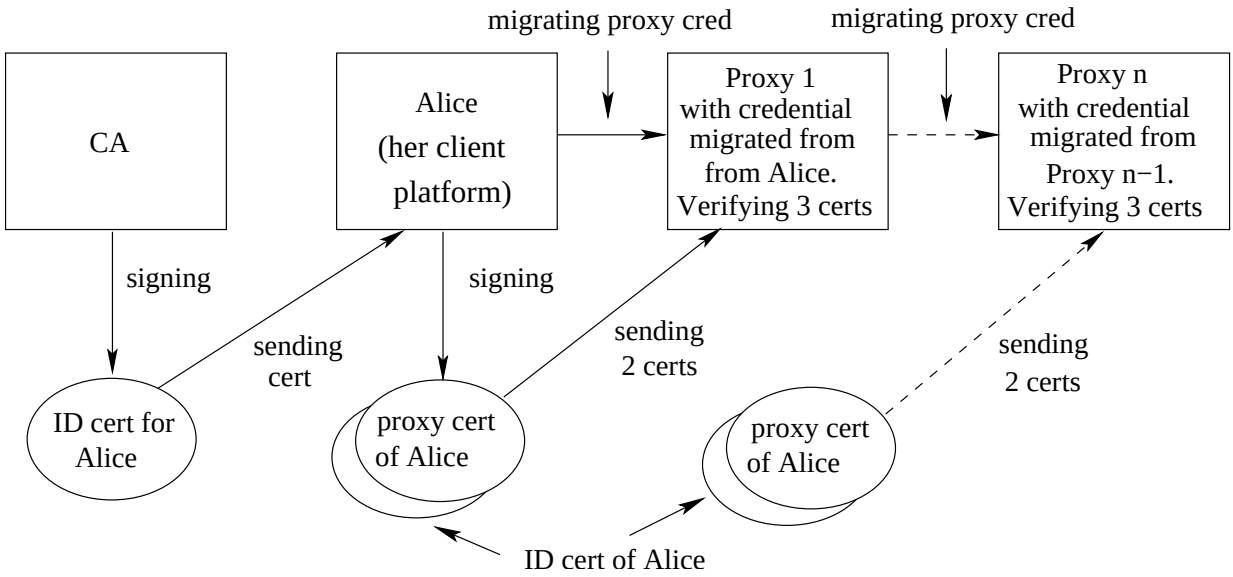

Figure 2: Grid VO Construction in Daonity: Using Migratable Credential

As illustrated in Figure 2, now the number of proxy certificate needed for the VO construction is precisely one. Each subsequent joining resource provider will be able to verify the final signature for resource request in a short chain of certificates: the signature, which is created in the TPM in the previous Proxy using Alice's migrated credential, is supported by Alice's proxy certificate, which in turn

is supported by Alice's identity certificate. So it is Alice who has authorized the procedure for resource allocation and VO construction. The unique proxy credential of Alice which has been migrated to the TPM of each resource provider will suffice to entitle them to act on behalf of Alice in a secure manner.

Moreover, with TPM protection, short lifetime stipulation for the unique proxy certificate becomes unnecessary. After the Grid task completes, Alice can revoke her proxy certificate, using the standard method for certificate revocation.

4.4 Attestation of Platform Physical Configurations for Conformed VO Policy

Now let us deal with Limitation 3 in §4.3. We confine ourselves to a relatively easy case of considering what care should be taken in order for Alice's proxy credential to be securely migrated to resource providers.

Firstly and most obviously, Alice should be sured that the destination of each migration of her proxy credential will be an RTS shielded location protected by a bona-fide TPM. This confidence of Alice can be obtained by a probably most basic form platform attestation: the platform to join the Grid service for Alice's VO does have integrated with a TPM. In TCG, a bona-fide TPM-platform is uniquely identified by an **endorsement key (EK)** which consists of a pair of public/private keys. In principle, the public EK is certified, and an attestation challenger can challenge the TPM-platform for the latter to respond with a signature, and so the challenger can verify the bona-fine-ness. In reality, each EK is pseudonymized by an unlimited number of **attestation identity keys (AIKs)** which are pseudonymously certified, and linked to the EK, by a **Privacy CA**'s service, so that in each attestation instance a Privacy CA certified AIK is used. This is a privacy positive design feature of TCG. In Daonity, we follow TCG's standard and use the Privacy-CA and pseudonymously certified AIK whenever we need to identifying a TPM-platform.

Using a slightly more advanced form of platform attestation, Alice can further impose a policy to the VO constructed in Figure 2. For the purpose of Daonity applications and for clarity in our exposition while without loss of generality, let us confine the VO policy to platform physical configuration properties. Here physical can mean either some hardware configurations of a platform such as the number of CPUs, the speed of the CPUs, and the capacity of the storage, that the platform is equipped, or some other information of a platform such as the organization it belongs to, or its geographical location, or any other specifiable and identifiable information about the platform. We further assume that these physical configuration properties of a platform have been certified by a trusted third party after the integration of the TPM into the platform. The process of the certification is out-of-band but is standard.

Even with this simple form of attestation of platform physical configurations, Alice can already impose that a VO be constructed to satisfy a specified policy properties. For example, any participating resource provider must meet some specifiable criteria for hardware, organizational or geographical configurations. The credential migration which we described in §4.3 will be conducted only when the criteria are met.

We notice that the simple forms of platform attestation described here are practically achievable with the TCG specification for TPM version 1.1b. We have implemented them in Daonity.

4.5 Cryptographic Protection for Grid Authorization Mechanism

Globus Toolkit designed a **GridMap** as a Grid utility authorization mechanism. The GSI System Administrator's Guide [7] provides a detailed description of this Grid authorization mechanism. We provide here a brief description of this mechanism.

When new Grid users in a VO require access the Grid computing resource from a resource provider, they will need to be vouched for by the VO leader. Once authorized, their Grid PKI certificate subject

will be added to a file called **grid-map file**, which is used to map distinguished names to local new unique usernames (such as login accounts) in the resource provider.

When an updated grid-map file is created, a notification will be sent to all Globus system administrators, and each of them will implement this new file on their respective clusters the VO will be using.

At a resource provider, the Globus Toolkit uses a grid-map file to discover a GSI-authenticated user's local user id. In order for a given user to be able to upload and download files using the data service, that user must have an entry in the grid-map file.

The following is a brief summary of the format of the grid-map file:

- The grid-map file is a text file.
- Its default location on a unix system is `/etc/grid-security/grid-mapfile`
- Each line is of the form "distinguished name" user-id$_1$, user-id$_2$, ..., user-id$_n$.

Here, "distinguished name" (DN) is the identity found in the Grid users' global PKI certificate.

A Globus Authorization Manager (GAM) at the resource provider, who has the root user's privilege, creates and maintains the grid-map file. Because of the dynamics of a VO and its utility of resource providers, a grid-map file also has a dynamic property and requires often updating. Once updated, it is saved in the default location (see above) for the GT to use in an automatic fashion (i.e., the use the grid-map file by the GT does not need GAM's action). The GT uses the grid-map file as the default information-source for the authorization of client requests.

The grid-map file is an important file, which needs protection at least in terms of data integrity, and optionally in terms of data confidentiality. Obviously, protection in these quality needs cryptographic operations. In order for the GT to use it in an automatic fashion, GSI actually does not provide this quality of protection on the grid-map file. The only protection on the grid-map file is in the form of the unix file access protection mechanism. It is well known that the file access protection is too weak against sophisticated adversaries. Buffer overflow to grant a root user (GAM) privilege to a malicious code is a well-known attacking technique. Moreover, GSI does not provide any protection for the grid-map file against the GAM. To put this in an equivalent way, it is very hard for the GAM to prove her/his innocence upon occurrence of an authorization misuse event.

With TC's readily available protected capabilities from the TPM, cryptographic protection for the grid-map file becomes straightforward. We illustrate here an audit trail maintenance mechanism which allows the GAM to prove her/his innocence.

Whenever the GAM have completed the maintenance of the grid-map file by saving the file to its location, the RTS mechanism (see §3.1.3) will record the file in the form of a hash total to a PCR inside the TPM. Let

$$gm_1, gm_2, \ldots, gm_i$$

denote a history of grid-map files maintained so far from the beginning of an audit period. Upon maintenance and saving gm_{i+1}, the RTS mechanism in the TPM will perform the following audit keeping step:

$$\text{PCR\_for\_GridMap} \leftarrow \text{SHA-1}(\text{SHA-1}(gm_{i+1}) \parallel \text{PCR\_for\_GridMap}).$$

The TPM can sign the hash total value in the PCR periodically (using an AIK) and save the result to shielded location in the persistent storage, and so the history of grid-map files can be audited periodically.

This way of the TC supported audit trail keeping can find many other applications in utility and enterprise computing scenarios such as outsourcing mission-critical enterprise computing tasks.

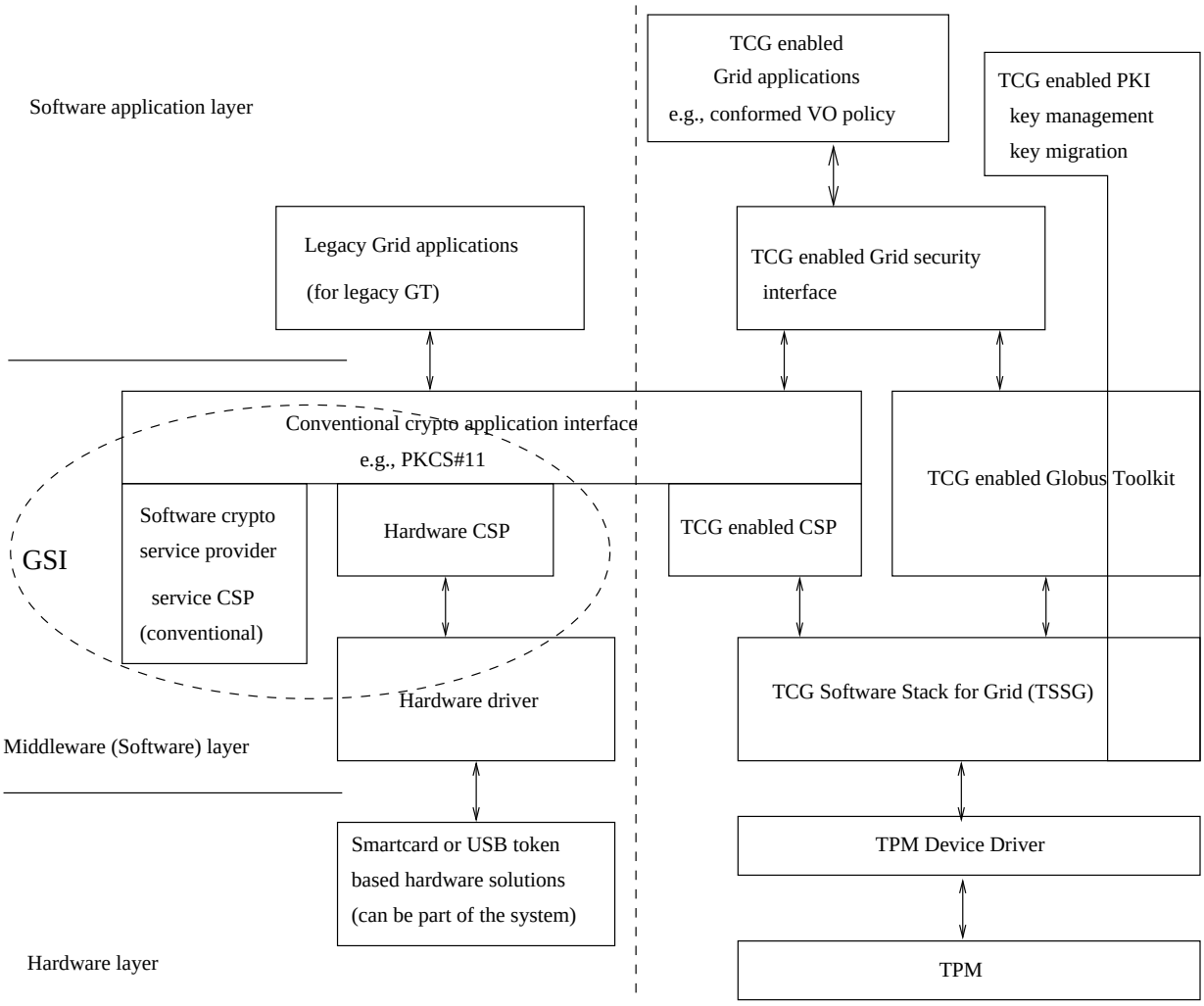

Figure 3: The Daonity System Architecture

5 Implementation

Figure 3 illustrates an architectural description of the Daonity system. The architecture provides a standard description for security programming model. It is divided into two parts:

- The part in the left-hand side of the dashed vertical line describes the standard programming model for security applications. The dashed ellipse indicates the legacy GSI realized under this programming model. In this model, (Grid) application developers who work on “Legacy Grid applications” call standard security services in e.g., PKCS#11 languages. Cryptographic service providers (CSP) consist of real implementations of crypto algorithms which can be in either software (running in the general CPU) or hardware (running in e.g., a smartcard or a USB token). This part should remain in Daonity since we anticipate a gradual adoption of a TC enabled Grid security solution (i.e., not to force people stop using the legacy GSI).

- The part in the right-hand is our new add-on to GSI to make it TC strengthened.

5.1 Implementation Status

We have planned to make Daonity a fully open-source system. The implementation of Daonity has greatly benefitted from the open-source TSS package TrouSerS [22] and the open-source Grid middleware package GT4. In fact, apart from the TPM migration component (see §4.3), all other TSS parts of Daonity are readily adapted and modified from the open-source code of TrouSerS and plugged into the open-source code of GT4.

At the time of writing this paper the TPM migration component has been completed for the TPM chip version 1.1b manufactured by Infineon Technologies AG on a number of HP platforms. We have planned our immediately next work to have Daonity implemented for TPMs of all major TPM manufacturers. The open source code will be released with full documentation in a very near future.

In the remainder of this section we describe the right-hand side of the Daonity architecture in more detail.

5.2 TCG Software Stack for Grid (TSSG)

In TCG specifications, the **TCG Software Stack (TSS)** component is a software system which is designed to make a uniform use of the TPM functions. In our Grid applications, this component is named TSSG and should interface and be integrated to Grid middleware toolkits. This is depicted in the right-hand side link between TSSG and the "TCG enabled Grid middleware" component.

5.3 TCG Enabled Grid Middleware

This component is a TCG enabled Grid security infrastructure in place of the legacy GSI in dashed ellipse. Facing upward, it should provide Grid application developers with standard security services which should be enlarged with TPM specific functionalities (such as key generation and the like). Facing downward, it is integrated to TSSG.

5.4 TCG Enabled Grid Security Interface

This component is an added interface between Grid applications and Grid security middleware. The need of this interface is to make Grid applications on our new Grid security system backward compatible to the legacy GSI. The backward compatibility is necessary because the legacy platform systems will co-exist with TCG platforms for a long time, and therefore a Grid application (mostly a protocol) must be able to run in both systems (in the case of a Grid protocol, it is even possible that part of it runs on a TCG enabled platform, and part of it runs on the legacy platform).

Ideally, the interface should be designed to allow a Grid application to detect if a local security under layer system supports TCG technology.

5.5 TCG Enabled Public-key Infrastructure

Grid security, even in the legacy GSI, applies the public-key authentication infrastructure (PKI) for ease of key management in cross-organization and cross-trust domains. In Daonity, the Grid PKI system should have additional TCG specifics such as TPM endorsement, Privacy certification, TPM-platform physical configurations certification, TPM initialization, TPM ownership management, TPM key backup, TPM migration, etc.

6 Related Work

Apart from GSI, GT and TrouSerS which are obviously closely related works, the following two papers seem to be most relevant to the work of Daonity.

The first is a paper on property-based attestation for computing platforms [20]. This paper argues that the attestation functionality proposed by the existing specification of the TCG can be misused to discriminate certain platforms and the operating systems running on the platform. Therefore the really essential element for an attestation should be properties of the software systems in the platform, not the software systems themselves. We consider that our work on attesting platform physical information for VO policy conformity (see §4.4) a form of property-based attestation.

The second is the work of SHEMP (Secure Hardware Enhancement for MyProxy) [14]. This is a system which hardens MyProxy, the on-line PKI credential management servers, using a hardware trusted computing base. A MyProxy server is an important and attractive target for the adversaries. Strong hardware based protection of the credentials and user passwords which are managed by a MyProxy server is very desirable. The hardware TCB proposed by SHEMP is an IBM cryptographic co-processor.

7 Controversy: Monopoly of Controlled Use of Software

With a platform which can be set by the owner into a state to have the properties of a conformed behavior for its physical configuration and software environment and a restricted capability for its users, the TCG technology has its detractors [1, 2] which interpret the TCG as to permit monopoly control over the use of software or information technology products. These views concern that, under a possible monopoly control of large companies, the TC technology can force the users to using only information technology products which are chosen by some given technology suppliers. Under these interpretations, "trust" will lose its original desired meaning, instead of offering protection for the users, it becomes the opposite: harming their interest. It is indeed a familiar recurrence of many technology innovations and advances which have been developed in the history, new ways of doing things have often been enable to serve good as well as evil. There are indeed a number of questions to be answered.

In a general setting of using connected computers, which is outside our confined Grid computing model, there are a number of questions to be answered regarding the merit of the TCG technology, for example [18]:

- If you have a platform that protects information, should a third party be able to use that mechanism in your computer to protect the third party's information from you?
- How will open-source self-certify software distributions, to say that these distributions will "do the right thing"?
- Will owners of commercial data trust open-source software distributions?
- What if Trusted Platforms are used merely to restrict choice, in circumstances where no data protection is really required?

Some authors argue [21] that market forces, combined perhaps with light-touch regulation and scrutiny, will help to keep the world sane. We may also observe that faulty software and malicious code abounds and will help to keep the market from becoming completely controlled by any single party.

At any rate, for our chosen Grid computing model, we are able to avoid this controversial issue. In the TC applications to Grid computing which we attempt in this paper, there should be much less disagreement since this computing model requires behavioral compliance from either an individual user

as a condition for using remote resources, or an service provider for proof of correct conduct in processing customers' computations, or implies federation and cooperation among a group of collaborators.

8 Concluding Remarks

As Grid security is becoming a more and more important topic, a number of problems remains untackled by the existing Grid security solutions. We have identified that behavior conformity is an essential requirement for Grid security, or in fact, for any distributed computing applications where a partner-and-adversary threat model applies. We have argued that trusted computing technology, thanks to its inherent property of behavior conformity, can provide suitable solutions to the identified problems in the existing Grid security solutions.

As hardware and software support for the TCG technology is gradually becoming widely deployed, it is timely to consider how such tools can be used to maximum effect in enhancing trust and security in Grid environments. The work of Daonity can be regraded as an early trial. The behavior conformity property in the work of Daonity is still in a very primitive stage, mainly and specifically, to limit the behavior of the TPM owne. Also the platform attestation is confined to easy cases of some certified configurations of hardware properties. Nevertheless, sufficient innovations have been identified through the progress of the work. We hope that, given the Daonity system to be released in open source, more interest will be generated through further development of the work by the community.

Acknowledgments

Greg Astfalk reviewed an early draft of this paper and provided insightful comments and suggestions. Graeme Proudler provided help with the TC technology background. Paul Vickers provided help with the Grid computing background. Nigel Edwards and Dirk Kuhlmann provided insights in the notion of secure virtualization.

References

[1] Ross Anderson. TCPA/Palladium frequently asked questions, 2003.

[2] Bill Arbaugh. Improving the TCPA specification. *IEEE Computer*, pages 77–79, August 2002.

[3] B. Atkinson, et. al. Specification: Web Services Security (WS-Security), Version 1.0, 05 April 2002.

[4] R. Bair (editor), D. Agarwal, et. al. (contributors). National Collaboratories Horizons, Report of the August 10-12, 2004, National Collaboratories Program Meeting, the U.S. Department of Energy Office of Science.

[5] Common Criteria. Available at `www.commoncriteriaportal.org`.

[6] I. Foster and C. Kesselman. *The Grid: Blueprint for a New Computing Infrastructure*, chapter 2: computational Grids, pages 15–51. Morgan Kaufmann, San Francisco, 1999.

[7] I. Foster, C. Kesselman, G. Tsudik, and S. Tuecke. A security architecture for computational grids. In *5th ACM Conference on Computer and Communications Security*, pages 83–92, 1998. Also see: `www.globus.org/toolkit/docs/3.2/gsi/admin/configuration.html`.

[8] I. Foster, C. Kesselman, and S. Tuecke. The anatomy of the Grid: Enabling scalable virtual organizations. *International Journal of High Performance Computing Applications*, 15(3):200–222, 2001.

[9] A.O. Freier, P. Karlton, and P.C. Kocher. The SSL Protocol, Version 3.0. INTERNET-DRAFT, draft-freier-ssl-version3-02.txt, November 1996.

[10] T. Garfunkel, M. Rosenblum and D. Boneh. Flexible OS support and applications for Trusted Computing. In the 9th Hot Topics in Operating Systems (HOTOS-IX), 2003.

[11] Global Grid Forum. Overview of the Grid Security Infrastructure. Available at `www.globus.org/security/overview.html/`

[12] Globus Toolkit Version 4. Available at `www-unix.globus.org/toolkit/`.

[13] ITU-T. Rec. X.509 (revised) the Directory — Authentication Framework, 1993. International Telecommunication Union, Geneva, Switzerland (equivalent to ISO/IEC 9594-8:1995.).

[14] John Marchesini and Sean Smith. SHEMP – Secure Hardware Enhancement for MyProxy. Technical Report TR2005-532, Department of Computer Science, Dartmouth College, Hanover, New Hampshire, February 2005.

[15] J. Novotny, S. Teucke and V. Welch. An Online Credential Repository for the Grid: MyProxy, Proceedings of the Tenth International Symposium on High Performance Distributed Computing (HPDC-10), IEEE Press, August 2001.

[16] OpenTC. Available at `www.opentc.net/`

[17] Siani Pearson, editor. *Trusted computing platforms*. Prentice Hall, 2003.

[18] G. Proudler. Trusting "Trusted Computing". Presentation at the TCG Business Community Day at 2005 International Conference on Information and Communications Security (ICICS'05), December 9 2005. Beijing, China.

[19] RSA Security. PKCS#11 v2.20: Cryptographic Token Interface Standard. 28 June 2004. Available at `www.rsasecurity.com/pub/pkcs/pkcs-11/v2-20/pkcs-11v2-20.pdf`.

[20] A.-R. Sadeghi and C. Stüble. Property-based attestation for computing platforms: caring about properties, not mechanisms. New Security Paradigm Workshop (NSPW), 2004.

[21] David Safford. Clarifying misinformation on TCPA, October 2002.

[22] TrouSerS. The open-source TCG Software Stack. Available at `http://trousers.sourceforge.net/`

[23] Trusted Computing Group. `www.trustedcomputinggroup.org`.

[24] Trusted Computing Group. Trusted computing platform alliance (TCPA) main specification, version 1.1a. Republished as Trusted Computing Group (TCG) main specification, Version 1.1b, Available at `www.trustedcomputinggroup.org`, 2001.

[25] Trusted Computing Group. TCG TPM specification 1.2. Available at `www.trustedcomputinggroup.org`, 2003.

[26] Trusted Computing Research Group. Available at `https://forge.gridforum.org/projects/tc-rg/`