Software and Hardware Certification Techniques in a Combined Certification Model

Antonio Muñoz and Antonio Maña
GISUM, University of Malaga, ETSII Informatica, Malaga, Spain

Keywords: Certification, Trusted Computing, Cloud Computing.

Abstract: Certification has been proved as an essential mechanism for achieving different security properties in new systems. However, it has important advantages; among which we highlighted the increasing in users trust by means of attesting security properties, but it is important to consider that in most of cases the system that is subject of certification is considered to be monolithic, and this feature implies that existing certification schemes do not provide support for dynamic changes of components as required in Cloud Computing running systems. One issue that has special importance of current certification schemes is that these refer to a particular version of the product or system, which derives that changes in the system structure require a process of recertification. This paper presents a solution based on a combination of software certification and hardware-based certification techniques. As a key element in our model we make use of the Trusted Computing functionalities as secure element to provide mechanisms for the hardware certification part. Likewise, our main goal is bringing the gap existing between the software certification and the means for hardware certification, in order to provide a solution for the whole system certification using Trusted Computing technology.

1 INTRODUCTION

New computing paradigms, such as autonomic computing, grid computing, service oriented computing, and cloud computing are transforming the Internet, from an information space to a dynamic computing space, where distribution of data and remotely accessible software services, dynamism, and autonomy are prime attributes. In particular, cloud technology offers a powerful and fast growing approach to the provision of infrastructure, platform and software services, known as IaaS, PaaS and SaaS services respectively, without the high costs of owning, operating and maintaining the computational infrastructures required for this purpose. However, despite its appeal from the economic, operational and even energy consumption perspectives, cloud technology still raises concerns regarding the security, privacy, governance and compliance of the data and software services offered through it. Such concerns arise from the difficulty to verify security properties of the different types of applications and services available through cloud technology and the uncertainty of the owners and users of such services about the security of their services, and of the applications based on them, once they are deployed and offered through a cloud. Our work uses the Trusted Computing technology(TCG, 2014) to provide trustworthiness at the lower levels of the stack, as underlying infrastructure. Trusted Computing uses cryptography to help enforce a selected behaviour. The main functionality of TC is to allow someone else to verify that only authorized code runs on a system. This authorization covers initial booting and kernel and may also cover applications and various scripts. Just by itself TC does not protect against attacks that exploit security vulnerabilities introduced by programming bugs. However, since this kind of technology is considerably restricted, we have adopted an approach based on different levels of dependence on the TPM chipset. According to the flexibility requirements of our system, by means of this mechanism we provide a tailored solution for different contexts. The provision of a complete study of the spectrum of possible cases is out of the scope of this paper. Our motivation is presenting the problem of the existing gap and gives a solution for the highest level of certified system stack, that is the more TPM dependent one. The remainder of this paper is structured as follows section 2 gives an overview of the state of the art. Section 3 gives an overview of our hardware based certification mechanism. Section 4 presents the binding scheme and gives some implementation de-

Muñoz A. and Maña A..
Software and Hardware Certification Techniques in a Combined Certification Model.
DOI: 10.5220/0005098204050410
In *Proceedings of the 11th International Conference on Security and Cryptography* (SECRYPT-2014), pages 405-410
ISBN: 978-989-758-045-1

tails of our basic scheme approach. Section 5 gives some conclusion and ongoing work.

2 STATE OF THE ART

Certification is a well-established approach for the provision of assertions on properties of entities such as systems and services. In these terms, those parts using certified entities can rely on the asserted properties, provided that the process of certification is known to produce sufficient evidence for the validity of the property of the certified entity. Some recent works put in evidence that the certification has started playing an important role in the Service-Oriented Architecture (SOA) environment with the aim certifying service functional and non-functional propertiescitedos,tres. However the definition of tailored solutions for cloud environment need a natural evolution of existing certification schemes, as the natural way to provide trustworthiness in cloud-based services and applications. Certification of cloud-based services and applications share a similar ground with SOA certification, due to the highly dynamic nature of both infrastructures, but introduce new requirements. Some approaches(Grobauer et al., 2011; Khan and Malluhi, 2010) address the future role of certification in the cloud but not describing how to solve the issues related with the extremely dynamic cloud environment. Software security certification demonstrates the reliability and security of software systems in such a way that an independent authority can check it without having to use the techniques and tools used in the certification process itself. It builds on existing software assurance, validation, and verification techniques but introduces the notion of explicit software certificates, which contain all the information necessary for an independent assessment of the certified properties. However, there is no existing mechanism to express and confront claimed security properties. Existing works, especially in the context of security certification schemes, have mainly focused on monolithic software components, and usually provide human-readable certificates used at deployment and installation time. Therefore, the approaches proposed so far for software security certification do not support a service-based scenario, since this scenario requires the availability of machine-readable certificates, and their integration within service selection and composition frameworks(Damiani and na, 2009). Damiani et al.(Damiani et al., 2008) first study the problem of assessing and certifying the correct functioning of SOA using security certificates based on signed test cases. A first step in the certification of SOA and Web Service has been done in 2008 by the USbased Software Engineering Institute (SEI) that has defined a Web service certification and accreditation process for the US Army CIO/G6. Anisetti et al. (Anisetti et al., 2011) then provide a test-based security certification solution for services and a first approach to its integration within the SOA environment. Important work has been also done in the context of certifying the Quality of Service (QoS) of web services(Al-Moayed and Hollunder, 2010). The proposed approaches (e.g., (Rajendran et al., 2010; Ran, 2003; Serhani et al., 2005)) mainly deal with the definition of extended UDDI services supporting QoS metadata in the discovery process. The research on certification of cloud-based services and applications is still in its infancy; only few pioneering works foresee the potential benefits of integrating certification schemes within the cloud infrastructure, but none of them provides a concrete solution to this open research issue. The Trusted Platform Module (TPM) is a hardware chip designed to enable commodity computers to achieve greater levels of security than was previously possible. The TPM offers three kinds of functionality. The secure storage allows user processes can store content that is encrypted by keys only available to the TPM. Platform measurement and reporting functionality allows a platform the creation of reports of its integrity and configuration state that can be relied on by a remote verifier. And platform authentication functionality allows a platform to obtain keys by which it can authenticate itself reliably. TC is technically provided not just to secure the hardware for its owner, but also to secure against its owner. Additionally, TC is provided by remote attestation(Coker et al., 2008) mechanism, which allows changes to the user's computer to be detected by authorized parties. We make use of a certification mechanism based on the underlying remote attestation to achieve a semantic attestation model appropriate for cloud computing to achieve a semantic remote attestation (Haldar et al., 2004). Consequently we will provide support for relating certifications with trusted computing proofs in order to provide a comprehensive trust chain covering the full cloud stack. Certification on TC platforms, where certificates of the higher levels (e.g., services) include conditions on the low levels based on TC, will provide scope for exploiting TC in complex execution tasks and provide a basis for completing the trust chain.

3 HARDWARE BASED CERTIFICATION MECHANISM

As we have mentioned in this paper, we have chosen Trusted Computing Technology as the hardware element to build our certification mechanism. TC Proofs are used to certify the lower layers (hardware, native OS and software infrastructure) and the software certificate is used for the higher levels (applications, services, software infrastructure). Current certification schemes do not provide a reliable way to assess the trustworthiness of a composite application at the point of use due to certificates are intended for human use, lack machine readable format, lack explicit and precise formulation of security properties, cannot be used for runtime security assessment. Additionally these are not suitable for dynamic environments, highly distributed environments, systems without a central control or controlled ownership, systems modified (e.g. by policy decisions), and these do not support dynamic replacement of components and the most relevant any kind of runtime binding mechanism. For this reason we introduce a new element named ASSERT, which is a new type of digital certificate. ASSERTS are implemented as a digitally signed SAML-contained XML document. Our approach is a mechanism based on a combination of certification of higher levels of abstraction (applications, services and parts of cloud software infrastructure) with the lower levels (hardware, Native OS, rest of cloud security infrastructure) with the TC Proofs, in such a way that software certificate and TC Proofs keeps. In practical application, we propose a model based on three levels of certification, to know the Service/Application, Platform and TC Proof. Our definition of binding corresponds to any means used to guarantee that an ASSERT corresponds to a service in the cloud.

4 A COMBINED APPROACH: BINDING SCHEME

The main aim of our approach is to ensure that the service (code and data) remains unchanged since evaluation, which is the hardest target to achieve. Therefore, we should not bind asserts to services but to complete configurations. Likewise, if the service asserted uses other services, these are also unchanged. Also, any dynamic check can be conveniently performed (this means quickly, transparently), and all relevant information contained in the ASSERT is bound to the service. We aimed that our approach is based on a previous study of the spectrum of possible cases of certified services in the cloud according to the level of flexibility and therefore its dependence on the TPM chipset.

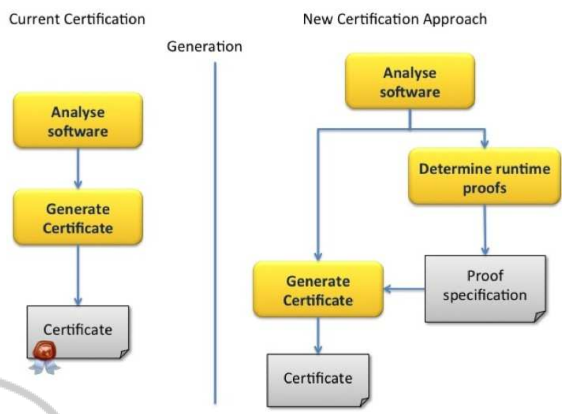

Figure 1: Certification Approach.

According to the flexibility requirements of our system, by means of this mechanism we provide a tailored solution for different contexts. However, since a description every solution for each context is out of the scope of this paper. Figure 2 shows our certification approach together with the existing certification mechanisms structure. Opposite to current certification mechanisms based on previous software analysis then the certificate is generated. In our semantic based approach, we perform two intermediate steps to determine the proofs and to specify them to generate the certificate.

Figure 2 shows an overview of our approach, with both use cases. The certificate authorization (assert creation) and client use case. The assert creation entails three steps to know; the evaluation of the service where the CA inspect the service and search in the properties. The second step is the assert creation, according to the properties the CA fills assert. The last step is securing the assert, a key pair is generated using the TPM functionalities sealed with the state of the system, then both the public key and the migratable key are included in the assert. The second use case corresponds to the client consists on three steps. By means of the set up the client verify the assert using the key related with the service.

The second step is the usage, client requests a service this replies encrypted/signature with private keys and the public key is used for the verification. In the last step the data within assert are used to install the migratable key in the destination TPM. Service providers can be made legally responsible for using the key pair only with the asserted service. The key pair resides in a TPM and it is bound to the asserted configuration of the service, since the TPM chipset provides means needed to both key generation and securely storage. The workflow is when the service is called, the TPM functionalities are used to attests the (complete) service configuration and the key is made available to the service. If the service has changed the TPM functionalities are used to check (attestation) will fail and the key will not be available. We highlight the fact that each response to a call to the service is signed with the service private key. In order to allow for different configurations each group of services sharing an infrastructure is executed in a virtual machine. Obviously, this solution implies some restrictions, such as services always run on TPM-equipped hardware. In those cases in which services run on virtual machines, then hardware must be provided on this. Clients has there requirement of being TPM-equipped hardware, at least in the more restricted cases. However, these restrictions are not hard to meet. Moreover, some restrictions can be relaxed since our model can be implemented without virtualization, and we can use sessions to avoid signature of all messages. One important appeal of our model is the inherent flexibility of this model since it allows that several binding mechanisms can co-exist. As we previously mentioned, our approach is based on the usage of trusted computing technology, which is essential to perform the attestation of both the hardware and the native OS. Our basic scheme makes use of the sealed bind key functionality provided by the trusted computing technology, in such a way that the sealed bind key is used to encrypt part of the service code in such a way that it can be only used when the platform is in that trusted state. This model is very restrictive, but it provides a high level of security since it establishes a very limited execution environment. Nevertheless, the limitations of this model make difficult its integration in real world scenarios, but a tailored solution based on this scheme can be suitable for particular cases.We differentiate between two use cases, the assert creation use case, where the Certificate Authorization is involved and the Assert User use case. The first step in the Assert Creation use case is the service evaluation, which involves that the CA checks a list of properties that must be fulfilled and makes an inspection of the service. Thus, CA fills in the assert form with extracted data. For this purpose, a key pair is created from a sealed key related to the state of the platform. A possible binding scheme must to consider that each service is asserted to operate with a key pair. Additionally, service providers can be made legally responsible for using the key pair only with the asserted service. The key pair resides in a TPM and it is bound to the asserted configuration of the service. When the service is called, the TPM attests the (complete) service configuration and the key sets as available to the service. If the service has been changed the TPM functionality is used to attest the state the checking will fail and the key will not be available. Each response to a call to the service is signed with the service private key. In order to allow different configurations, each group of services sharing the same infrastructure is executed in a virtual machine. This solution implies several restrictions, such as the services must be run on TPM-equipped hardware (or even virtualized). Additionally, the service clients must use cryptography. Nevertheless, these restrictions are not hard to meet and even some restrictions can be relaxed. In some cases we can do without virtualization. The most relevant appeal of our approach is the scalability, since this approach allows the use of asserts for the orchestration of different services (i.e dynamic coalitions for emergency targets), by means of the composition of services can be asserted by a composed assert. TC Proofs are limited (for this scenario), in the case that a high-level certificate (for in stance for a service) refers to a standard TC proof to define the platform state, it should be needed issue a different certificate for each valid platform configuration. This addresses to the need of improvements in flexibility, and interoperability. Following this target, we propose semantic approaches that can be the basis for the necessary improvements. Certificate Service A provides the property ”confidentiality of user data” if the platform provides encrypted isolated storage. The semantic proof specification is encrypted isolated storage. And destination platform provides encrypted isolated storage if measured configuration is ”mc”. Figure 4 shows different steps in the workflow of the certification process. In the traditional scheme validity and properties are analysed and then the certificate is accepted or rejected. In our scheme we introduced the extraction of runtime proofs that making use of a semantic proof specification TC software measurement are generated. Then workflow is split to generate measured TC Proof and we get the platform semantic certificate. Then the semantic proof certificate is generated and validated. Finally, measure TC proofs and required TC proofs from the semantic proof certificate are compared to accept or reject the certificate.

4.1 The Basic Binding Scheme Implementation

For the implementation of our basic binding scheme we make use of the Certified Migratable Key (CMK) functionality provided by the TPM. It is a kind of key that is halfway between a non-migratable key and a migratable key. These are inside of a TPM chip, but it

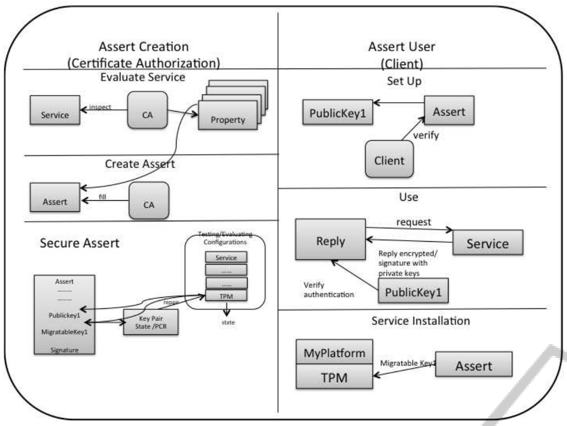

Figure 2: Our Certificate Authorization.

can be migrated to another TPM. Due to its features it fits in our requirements. At the time they are created, the creator has to pick up a Migration Authority (MA) or a Migration Selection Authority (MSA), which will have the authority to migrate the key. CMKs can both be migrated and also be considered secure, in the case you trust the MSA and MAs to migrate the key.

The complete trust and security functionality of a Trusted Computing Platform is based on the capabilities of the TPM to protect these keys and certificates. A TPM contains a Root of Trust Storage (RTS), which protects data and keys entrusted to the TPM. The RTS manages a small amount of volatile storage inside the TPM device that is used to hold currently used keys. Unused keys may be encrypted with a storage key and moved off the TPM chip, e.g., to a hard disk drive. The migratable key chain is designed so that only one key, the Legacy key (or Platform Migratable Storage Key) needs to be migrated in order to take all the keys below in the hierarchy in to a new TPM. For the migration it could be necessary that keys are required on different platforms, which are handled by a user alternatively. Sharing the same key across multiple platforms may be achieved by using key migration mechanisms. Key migration mechanisms allow the private keys from TPM-protected key hierarchy to be attached to other TPM-protected storage trees.

Migratable Keys (MK) can be moved to another TPM by using either rewrap (TPM_MS_REWRAP) or migration scheme (TPM_MS_MIGRATE). To migrate a key with TPM_MS_REWRAP, a destination TPM selects a storage key. This will be used as a parent for the migratable key and sends its public part to a source TPM. The source TPM rewraps the migratable key under the destination public key. The using of destination public key should be authorized by owner with TPM_AuthorizeMigrationKey command and rewrap procedure can be done with command TPM_CreateMigrationBlob. Resulting blob is then forwarded to the destination TPM in conjunction with a plaintext object describing the public key from the key pair to be migrated. The destination TPM can load blob with TPM_LoadKey command.

5 CONCLUSIONS & ONGOING WORK

Software certification is considered as an appropriate and robust mechanism for supporting assurance and compliance, but there are two important problems. The first of them is that it has been traditionally targeting humans and has not been able to support automated processing of certifications (i.e. verification, selection based on certifications, etc.). The second drawback is that certification cannot provide dynamic proofs of the status of a system at runtime, which are extremely important in a dynamic, heterogeneous and unpredictable scenario such as cloud computing.

Cloud computing architectures are based on a hardware, software and firmware underlying basis that is stable, in the sense that few changes are done in this basis. However, in the most abstract layers of the software (i.e., applications) changes are produced frequently, different applications are launched in systems sharing resources, resulting in many changes in the system execution stack. Trusted Computing (TC) technologies are well suited to provide proofs of the trustworthiness on the lower level of the cloud stack starting with the hardware layer, but are not efficient and practical when it comes to dealing with the very dynamic and heterogeneous higher layers (service / application).

The proposed scheme can successfully bridge the gap between Trusted Computing and Software Certification by combining the best of both worlds and overcoming their respective limitations. The concept of ASSERT as a computer-oriented form of certification is an essential key for improving the flexibility and practical applicability of TC mechanisms. This approach can open new application fields for TC. The approach is based on the results of the ASSERT4SOA project, and the current model has been developed in the CUMULUS project.

Among the ongoing work we are working in the study of different schemes for every particular case building a tailored solution for every one. Another alternative consists on the use of sessions, we can use sessions to avoid signature of all messages. In this way, the handshake is done at the starting of the session; while the session is open it is assumed that the services is unchanged. This solution allows that every time that we want to check that the service is unchanged it is possible to restart the session. Additionally, we can implement different binding mechanisms for specific purposes (i.e. for legal aspects) that can co-exist.

ACKNOWLEDGEMENTS

This research was undertaken as part of the EC Framework Programme as part of the CUMULUS, Certification infrastructure for multi-layer cloud services and PARIS Privacy preserving infrastructure for surveillance, and the Andalusian Government funded project ICES, Identification and Certification of Software Elements.

REFERENCES

Al-Moayed, A. and Hollunder, B. (2010). Quality of service attributes in web services. In *5th International Conference on Software Engineering Advances*. IEEE.

Anisetti, M., Ardagna, C., and Damiani, E. (2011). Fine-grained modeling of web services for testbased security certification. In *8th International Conference on Service Computing*. IEEE.

Coker, G., Guttman, J., Loscocco, P., Herzog, A., Millen, J., OHanlon, B., Ramsdell, J., Segall, A., Sheehy, J., and Sniffen, B. (2008). Principles of remote attestation. In *special issue of the 10th International Conference on Information and Communications Security*. Springer-Verlag.

Damiani, E., Ardagna, C., and Ioini, N. E. (2008). Open source security certification. Springer-Verlag.

Damiani, E. and na, A. M. (2009). Toward ws-certificate. In *ACM Workshop on Secure Web Services*. ACM.

Grobauer, B., Walloschek, T., and Stocker, E. (2011). Understanding cloud computing vulnerabilities. In *Security & Privacy*. IEEE.

Haldar, V., Chandra, D., and Franz, M. (2004). Semantic remote attestation: a virtual machine directed approach to trusted computing. In *3rd conference on Virtual Machine Research And Technology Symposium*. ACM.

Khan, K. and Malluhi, Q. (2010). Establishing trust in cloud computing. In *IT Professional*. IT Professional.

Rajendran, T., Balasubramanie, P., and Cherian, E. (2010). An efficient ws-qos broker based architecturefor web services selection. In *International Journal of Computer Applications*. International Journal of Computer Applications.

Ran, S. (2003). A model for web services discovery with qos. In *ACM SIGecom Exchanges*. ACM.

Serhani, M., Dssouli, R., Hafid, A., and Sahraoui, H. (2005). A qos broker based architecture for efficient web services selection. In *IEEE International Conference on Web Services*. IEEE.

TCG (2014). *Trusted Computing Group: TCG Specifications*. Trusted Computing Group.