A DIGITAL SECURITY CERTIFICATE FRAMEWORK FOR SERVICES

Samuel Paul Kaluvuri1, Hristo Koshutanski2, Francesco Di Cerbo3, Renato Menicocci3, Antonio Maña2
1 SAP Labs France, 2 University of Malaga, Spain, 3 Fondazione Ugo Bordoni, Italy

samuel.paul.kaluvuri@sap.com, hristo@lcc.uma.es, francesco.di.cerbo@sap.com, rmenicocci@fub.it, amg@lcc.uma.es

Abstract

Service Oriented Architectures have facilitated a paradigm shift in software provisioning models: software gets consumed as a service providing enormous benefits to both service providers and consumers. However, a major barrier for a wider adoption of the new service provisioning model in business and security-critical domains is the lack of security assurance of a service. Although current security certification schemes typically provide the required assurance, applying them to service environments is practically infeasible, a major reason being the natural language representation of the security certificates which becomes a major obstacle for adoption in typical service environment scenarios such as service discovery and service composition. To overcome the limitations of existing security certificates, we present a full-fledged framework for the realization of the concept of digital security certificates for services. The framework comprises a language for machine processable digital security certificates (DSC), and the concept of a security profile that allows the generation of uniform DSCs and a tool that aids in the generation of the DSCs.

Keywords: Security Assurance, Digital Security Certificates, Security Certification, Web Service Security

1. INTRODUCTION

Service Oriented Architectures have facilitated a profound change in the manner software is provided and consumed. Software, now, is offered as a service relieving consumers from the complexity of procuring and maintaining large scale IT infrastructures and at the same time facilitating inter-organizational interoperability (Gartner, 2012). However, in such provisioning models, the consumer ceases to have any control over the software provisioned or its operational environment. This lack of control, especially in critical domains such as finance, defence and healthcare, raises concerns about the security of these services (Gartner, 2012). Traditionally, consumers were able to gain the required assurance by establishing Service-Level Agreements (SLAs) with the service providers. But service landscape is dynamic and in such environments SLAs do not scale well. More often than not, the consumers end up agreeing to the terms and conditions of the service provider rather than ascertain whether their requirements - both functional and security - are met by the service.

In traditional software provisioning models, security certification of software by trusted third party entities is used to provide security assurance to consumers. Certification schemes such as Common Criteria (The Common Criteria Recognition Agreement, 2009) are well established and quite successful in providing the required security assurance to consumers in a scalable manner. However, current certification schemes result in certificates that are represented in natural language, which do not cope well with the dynamic service environment.

Service consumers should be able to compare the (certified) security features of a service with their security requirements and in addition to compare the (certified) security features of service offerings from different service providers. However, current security certification processes result in certificates that are represented in natural language and filled with legalese; this prevents any sort of automated reasoning to be performed on them.

In order to bring security certification to the service environment in a manner that they can play a constructive role in the service selection and service consumption process, several modifications to the current state of the art are necessary (Lotz, Kaluvuri, Di Cerbo, & Sabetta, 2012). Among them, we focus on the aspects related to the security certificate generation and representation. In this regard, we have identified the following key requirements from a service consumer perspective that need to be addressed in order to facilitate security certificate adoption in service landscape.

Requirement 1: The security certificates must be machine processable in order to allow automated reasoning to be performed on them.

Requirement 2: Security certificates should contain enough information about the certified entity so that they can cater to consumers with varying levels of security knowledge, such as regular users with limited security understanding to security experts of organizations. In other words, it is necessary that the certificates are descriptive (Wallnau, 2004), meaning that they describe with sufficient

http://hipore.com/ijsc
details the security features of their services, together with supporting evidences.

Requirement 3: Mechanisms must exist in order to bind a service and its security certificate, given that a service implementation can change, while maintaining the same external interface or API. Consumers would need to have trustworthy and dynamic means to verify whether a service implementation they are using is the certified one.

The main objective of a security certificate of a service is to provide security assurance to potential consumers. The existing security certification schemes certify products at varying levels of assurance. The number of levels and the type of assurance depends on the certification scheme. Each certification scheme fixes these levels based on carefully designed and selected criteria. The levels of assurance provided by the current schemes distinguish certified products based on security features (such as FIPS-140) or the rigour of evaluation (such as CC, CPA) (The Common Criteria Recognition Agreement, 2009). However, security assurance is a multi-dimensional property that amalgamates assurance gained from security features of the product, evaluation of the product, etc. Hence, the levels of assurance provided by the existing schemes are a part of the overall security assurance that a consumer gains from the certificate. We have identified some of the aspects that can impact the security assurance provided to a consumer: i) the rigour of the evaluation; ii) the trust that the consumer has on the certificate issuer and evaluators; iii) the extent of information that is provided in the security certificate (trust through transparency). Hence, the security certificate representation should cope with different certification schemes with varying notions of “assurance”. This is yet another requirement that needs to be addressed by the security certificate.

Requirement 4: A security certificate should be certification scheme independent, so that different certification schemes could generate security certificates that are represented in a standard manner.

We present the concept of a digital security certificate (henceforth referred as CRT), that in our view addresses the mentioned security certification issues in service environment. In addition we propose the concept of a digital security certificate Profile (DSC Profile) that allows production of certificates that satisfy certain semantic and syntactical requirements.

The paper is structured as follows: Section 2 describes a service environment scenario for the application of the CRT concept, while Section 3 presents the state of the art in security certification and digital certificates. Sections 4 and 5 depict different aspects of the CRT; respectively its conceptual model and its technical representation. Section 6 explains the issues that arise from the CRT representation model and Section 7 presents the concept of a CRT Profile that overcomes the issues of the CRT representation model. In Section 8 we present the tool that can be used to generate CRTs by the Certification Authorities and in Section 9 we discuss the advantages of using the CRT concept and the impact it can have on the current security certification practice. Section 10 concludes the paper.

2. USECASE PRESENTATION

In order to illustrate the expressivity of the CRT, we present a scenario of a cloud storage service, “TitaniumBox”. We model the TitaniumBox service on the popular cloud storage service “Dropbox” (Dropbox inc, 2013), however, we do make certain assumptions where necessary and these are stated clearly. We assume that the TitaniumBox service has undergone a security certification process similar to a Common Criteria certification. In the following sections, we describe the service architecture and the security properties that the service claims to possess.

2.1 Service Description of Titaniumbox

The service allows consumers to store their files remotely. In addition, it also performs versioning of the files that are uploaded by the consumers. The service exposes an interface through a WSDL file that enables consumers to invoke the service. The WSDL file provides information only regarding the interface, as in the data types and the operations available to a consumer. The service is actually composed of multiple components that work together to provide the storage service to the end-user. A more detailed description of the system is shown in Figure 1. As can be noted from the figure, the service uses another storage service, Amazon S3, provided by an external party (Amazon). Let us assume the service runs on the Apache CXF framework on a server that runs on Debian operating system platform, in addition, the service uses a relational database, MySQL to store user specific information. When such a service undergoes security certification process, similar to the Common Criteria, it results in a security certificate that is captured in human readable form. Clearly, such a certificate does not allow automated reasoning to be performed, thus not supporting assessments and comparisons among alternatives which is essential in scenarios such as service discovery.

2.2 Security Properties of Titaniumbox

The TitaniumBox service provider claims to have implemented several security mechanisms to provide certain

![Figure 1. TitaniumBox Architecture Overview](http://hipore.com/ijsc)
security properties to consumers. The service provider
claims to have implemented the AES-256 encryption to
provide the confidentiality of the user data when stored.
This mechanism is implemented within the component
“FileValidation” component. However, these claims alone
will not provide any assurance unless the certification
authority evaluates the service and certifies that the service
indeed has the security properties claimed by the providers.
The certification authorities must represent these certified
security properties at varying levels of abstraction to cater to
consumers with varying levels of security knowledge, needs and requirements.

3. STATE OF THE ART
A survey of the current security certification schemes reveals that there are quite a few established and successful schemes such as Common Criteria for Information Security (CC) (The Common Criteria Recognition Agreement, 2009), Commercial Product Assurance (CPA) (National Audit Office, 2013) and so on. Security certification schemes can be broadly classified based on the domains that they are applicable in, the recognition of the certification schemes, the descriptive or normative character of the issued certificates and so on (Wallnau, 2004).

Table 1 presents an overview of the existing security certification schemes. It can be observed that several certification schemes are specific to a particular domain such as Mobile Security, Network Security and so on, while a few schemes such as Common Criteria (CC), Commercial Product Assurance (CPA), and First Level Security Certification (CSPN) are applicable across the different domains. Among these broadly applicable schemes, the CC is a widely recognized (Common Criteria, 2012a), used (Common Criteria, 2012b), multi-domain (Turner, 2009), partially descriptive certification scheme (Beckert, Bruns, & Grebing, 2010). Schemes such as CPA, CSPN have different criteria towards the certification but they result in certificates that are represented similar to the CC Scheme. And hence we examine the Common Criteria scheme further for our analysis.

3.1 Common Criteria Certification Scheme
The CC scheme can trace its origin from schemes such as Trusted Computer System Evaluation Criteria (TCSEC) (USA Department of Defense Standard, 2013) also referred as “Orange Book”, the Canadian Trusted Computer Evaluation Criteria and Information Technology Security Evaluation Criteria. It has unified its predecessors with a standard set of criteria for security evaluation. It is designed in a very generic way which enables the CC scheme to certify products that range from software, firmware to hardware.

It decouples the specification of security functional and assurance requirements, this is in direct contrast with the approach of its predecessor TCSEC where the security functional and assurance requirements are coupled (USA Department of Defense Standard, 2013) together to provide a “balanced” assurance regarding a system. This decoupling is needed as CC targets a commercial security market as opposed to the “Orange Book” which was limited to products designed to be used by the US Government organizations. The CC scheme is very generic, as it aims to evaluate security of products that range from software, firmware to hardware. It avoids an all or nothing benchmark, by providing security assurance at varying levels, called Evaluation Assurance Levels (EAL), this provides flexibility for product vendors to get their product certified at lower assurance levels and improve the EAL over time.

The CC scheme is primarily “claims” based as it allows product vendors to describe the security functional requirements (SFRs) that are met by the product and to prove that the set of SFRs are able to counter the threats identified for a Target of Evaluation. This information is captured in a document called “Security Target” (Common Criteria, 2012b) (CC-ST) which is often seen as the descriptive part of the CC certification (Beckert et al., 2010). The product vendor then specifies the set of Security Assurance Requirements (SARs) in order to provide a certain Evaluation Assurance Level.

The CC scheme also enables consumers or consumer groups to specify their security functional and security assurance requirements for a certain type of product in a implementation independent way. This information is captured in a document called “Protection Profile” (CC-PP). Product vendors can then design products that conform to a certain protection profile.

The standardized SFRs and SARs are the “common” part of the CC scheme allowing, in theory, comparison of security features of certified products. However, in practice, the comparison of products which have different “claims” can be very hard. This is due to the representation of the CC related documents (CC-PP, CC-ST) in natural language, which is often filled with legalese and heavy security jargon making it rather complex to understand for non-security experts. Hence it becomes extremely difficult to determine if a particular product satisfies a consumer’s security requirements and to compare different products against their requirements. In (Lotz et al., 2012), it is observed that natural language representation of certificates is not a scalable solution when we consider service marketplaces such as Google Apps for business, Salesforce etc. There, the analysis of thousands of application/service offerings and their human readable security certificates would represent an unsustainable burden for customers; though the availability of security certificates could represent a means to gain assurance on offerings, it cannot facilitate any sort of automated reasoning such as compare and/or contrast the security properties of different services. It also prevents
consumers to search for services based on not only their functional but also security requirements.

Although there are a few “digital security seals” such as the TRUSTe privacy seal (Benassi, 1999), McAfee SECURE seal (McAfee, 2007) and so on, these seals are purely normative statements regarding the security feature of an entity, which can be seen as a step towards digital security certificates, but cannot provide any meaningful assurance to consumers as they do not contain any information regarding the certified entity.

In order to overcome these limitations we have developed a language to represent a security certificate in a structured, machine processable format. The conceptual model on which the language is based on, is presented in the following section.

4. CONCEPTUAL MODEL OF DIGITAL SECURITY CERTIFICATE

The CRT, which is the conceptual model for a Digital Security Certificate (DSC), is designed to capture information emanating from security certification processes. In particular, we have considered the CC scheme, as it is the most widely used scheme currently. The CC-ST, which is the descriptive part of the CC scheme, serves as a foundation for our CRT. However, we have extended this significantly, in order to make it machine processable and suitable for service-specific needs. In contrast to CC, and other existing certification schemes, the digital security certificate is designed to be completely descriptive (Wallnau, 2004), and hence it contains the description of the certified entity, the security properties of the certified entity, the evaluation details that prove the certified properties, while the UDE can be either used by the service providers to disclose any additional information and/or by the certification authorities to state any further criteria. The UDE provides flexibility to the CRT concept by allowing certification scheme-specific or service-specific information to be captured. These four elements serve different purposes and together contribute in providing assurance on the security of the service.

4.1 Service Description

In CC-ST, a unique link cannot be made between the security properties and the assets that they secure, since, e.g., the assets (a native concept in CC) are described in natural language and without identifiers, and the security properties (not a native concept in CC) have to be derived by analysing the whole security problem definition. In order to overcome this, we adopt an asset-centric approach with explicit references between the assets and the different elements in the security certificate.

Definition 1: An Asset, a, is an entity that is of some value to the consumer or the provider. Assets can be data, applications, the IT equipment on which the service operates or even users of the Information System.

The CC-ST contains the Target of Evaluation (TOE) that describes the system that is being certified and the boundaries of the evaluation are indicated, albeit in an ad-hoc manner. However for a machine readable certificate there should be a clear distinction between the system that is
being certified and the aspects of the system that are subject to evaluation. It is of utmost importance in service based systems, due to the fact that services can be easily composed of external services and this information should be a part of the service description but clearly marked as outside the scope of evaluation.

The TOE in a CC-ST also contains the system architecture, the different components that compose the system among other information such as configuration in which the system is evaluated, the underlying IT architecture etc., and this is represented in natural language accompanied by architecture diagrams. This poses another issue in representing the TOE in a machine processable manner. In order to address these two issues, we introduced an element called Target of Certification (TOC) that describes the service being certified, while the TOE describes the part of the Target of Certification that is evaluated.

**Definition 3:** A Target of Certification is a tuple \( \text{TOC} = (\text{ACI}, \text{DM}, \text{TT}) \) where \( \text{ACI} \) is the Asset-Component Identification, \( \text{DM} \) is the Deployment and Implementation Model and \( \text{TT} \) is the TOC Type.

**Definition 4:** An Asset-Component Identification is a tuple \( = (A, C, \alpha) \), where \( A \) is the set of all the assets identified for the TOC, \( C \) is the set of all the components in the TOC and \( \alpha \subseteq A \times 2^C \) maps each Asset with a set of Components.

**Definition 5:** The TOE is a subset of the Asset-Component Identification. \( \text{TOE} \subseteq \text{ACI} \)

The TOC Components are an integral part of the TOC as they allow TOC to be expressed in a modular and structured manner. A TOC Component comprises an abstract model of the Component, the Component Model: it can be as simple as just containing the interfaces of the component, or a more detailed specification of the internal dynamics of the component as deemed sufficient by the Certification Authorities. It must also contain technical specifications of the Component, again at the level of abstraction as deemed sufficient.

**Definition 6:** A Component is a tuple \( C = (\text{CM}, \text{TM}) \), where \( \text{CM} \) is the component model and \( \text{TM} \) is the technical model.

We do not provide a formal definition for \( \text{CM} \) and \( \text{TM} \) as such concepts can be considered atomic for the sake of our work.

A Security Problem Definition (SPD) is essential in a security certificate as it provides the rationale for securing the assets. The rationale for securing the assets can stem from the threats that are identified for the assets by the service provider or from the service provider’s security policy (which in turn could be due to compliance to regulations etc.).

**Definition 7:** The security problem definition is a tuple \( \text{spd} = (\hat{A}, \text{spr}) \), where \( \hat{A} \subseteq A \) is a set of assets that need to be secured and \( \text{spr} \) is a security problem rationale for securing the assets.

**Definition 8:** The security problem rationale (spr) is a union of threats \( T \) and service provider’s security policy \( \text{SSP} \).

The service description must contain the description of the certified system, the part of the system that is evaluated and the rationale for protecting the assets that are identified.

**Definition 9:** The Service Description is a tuple \( SD = (\text{TOC}, \text{TOE}, \text{SPD}) \) where, \( \text{SPD} \) is the set of security problem definitions (spd).

### 4.2 Security Property Specification

The CC-ST contains a vast amount of information but is often presented in heavy-jargon; this rarely allows a consumer (a non-security expert) to get a high level perspective of the security features provided by the software/service. Hence we introduced a new element in the CRT model called as “security property specification” which enables a fine grained description of the security property that originates from a multi-layered model. It comprises of different elements, from abstract security properties to concrete security mechanisms.

**Definition 10:** An Abstract Security Property \( \hat{p} \) is an atomic security attribute for an asset.

For example, abstract security properties can be confidentiality, integrity, availability, authenticity, non-repudiation, utility, privacy and so on. Since abstract security properties by themselves do not convey any information on how the property is applied, there is a need for contextual information. Hence we define Contextual Security Property.

**Definition 11:** A Contextual Security Property is an abstract security property realized in a certain context. \( \hat{p}_c = (\hat{p}, c) \), where \( c \) is a context.

Contexts depend on the abstract security property. Abstract security properties that are data centric such as the CIA triad can have contexts such as transit, rest and usage (such as Confidentiality in rest and Integrity in transit). However these properties still lack a subject, i.e., no indication of “what” is being secured. This is addressed by the certified security property.

**Definition 12:** A certified security property \( p \), is a contextual security property \( \hat{p}_c \) applied on a set of Assets \( \hat{A} \). \( p = \hat{p}_c \times \hat{A} \)

The (certified) security property provides a high level overview of how an asset is secured. But this does not provide any information on how the SPD are addressed. This is overcome by using the concept of “Security Objectives” similar to the CC scheme. A security objective, \( \text{so} \), counters, mitigates or detects a spd that is identified for the TOE and contributes to the realization of a security property \( p \) for the TOE.
Definition 13: A security objective is a tuple \(so = (O, OT, SPD)\), where \(O\) is the objective, \(OT\) is the objective type, \(SPD \subseteq SPD\) is a set of security problem definitions.

All the security objectives are necessary and sufficient conditions to realize the security property. In other words, a TOE can have a security property \(p\) if and only if all the security objectives for the TOE are satisfied. Security Objectives are realized by security mechanisms that should be implemented in the TOE.

A Security Mechanism, \(sm\), is an action, device, procedure, or a technique that meets or opposes (counters) a threat or an attack by eliminating or preventing it, by minimizing the harm it can cause or by discovering and reporting it so that corrective action can be taken. Security mechanisms refer to the security objectives that they satisfy, and they can be mapped to specific functional criteria of a particular certification schemes.

Definition 14: A security mechanism is a tuple \(sm = (M, SFC, SD)\), where \(M\) is the mechanism that is implemented, \(SFC\) is the security functional criteria of a certification scheme and \(SD \subseteq SO\) is a set of security objectives that the mechanism realizes.

Definition 15: A Security Property Specification is a tuple \(SPS = (P, SO, SM, \gamma, \eta)\) where \(P\) is a set of certified security properties, \(SO\) is a set of Security Objectives, \(SM\) is a set of Security Mechanisms, \(\gamma \subseteq P \times 2^{SO}\) maps each security property to a set of security objectives and \(\eta \subseteq SO \times 2^{SM}\) maps each security objective to a set of security mechanisms.

This fine grained representation has two major advantages: allows consumers with varying security understanding to gain understanding of the security features provided by the service (security properties to security mechanisms); allows the certified security property to be machine processable that enables consumers to easily search for services that match their security requirements.

4.3 Evaluation Specific Details

The ESD defines the representation of the details and results of the service evaluation process needed to support the certified security property. We identified these three different categories for evaluation of services: Evaluation through testing (M. Anisetti, Ardagna, & Damiani, 2011; Marco Anisetti, Ardagna, Damiani, Pandolfo, & Maña, 2011), Evaluation through formal analysis (Fuchs & Gürgens, 2011), and Evaluation through ontology-based analysis (D’Agostini, Di Giacomo, Pandolfo, & Presenza, 2012). Given that these different types of evaluation results in very heterogeneous models and results, we refrain from modelling them at a conceptual level. We rather allow many different evaluation models and results to be plugged into the element in the broad categories mentioned before.

5. REPRESENTATION OF A DIGITAL SECURITY CERTIFICATE

In order to realize the conceptual model of the digital certificate \(CRT\), we have developed an XML-based language that enables the representation of the certificate in a machine processable form, which from henceforth we refer to as an ASSERT. A detailed description of the latest version of the schema can be found in (Koshutanski et al., 2013) and in this section we will explain its most relevant elements using the example introduced in Section 2.

5.1 SAML as Container of ASSERT

The management and exchange of the ASSERTS is an important consideration for a successful implementation of a certification ecosystem, i.e., production, maintenance, consumption of certificates. In this context, the container of the ASSERTS assumes significant importance as it is needed to encapsulate the certificate data into an interoperable format that can be used with existing web service standards and technologies. We have chosen the SAML standard (SAML Specification, 2012) as a container because it is widely used in decentralized systems for its support for request and exchange of SAML Assertions, either for authentication or authorization of entities, or any attributes of an entity. The SAML standard has support for several standard profiles for usage of SAML tokens in specifications such as WS-Security (OASIS, 2006), WS-SecurityPolicy (OASIS, 2007a), WS-Trust (OASIS, 2007b), etc. These aspects make SAML a good choice to be a container for exchanging ASSERTS in service environment. We use the SAML Assertion tokens to encapsulate ASSERT-specific data.

Figure 2 shows the main elements of the SAML assertion token structure where the Statement element defines an abstract statement of an assertion. Similar to how SAML authentication and authorization decision statements extend the abstract Statement element, we extended this element to provide a statement about a service’s description, its security property along with the corresponding evidence. The standard field Issuer in the SAML token is used as a means to capture the ASSERT Issuer’s identity (the certification authority issuing the ASSERT). The Subject field represents the identity of the certificate requester, which in most cases will be the service provider. And the validity conditions and the signature data are inherent to all security tokens.

5.2 ASSERT Structure

Figure 2 shows main elements of the ASSERT structure. It has three major elements: ASSERTCore, ASSERTType-Specific and UserDefinedExtensions. The ASSERTCore part contains elements that are independent of the evaluation of a service, i.e. the \(SD\) and the \(SPS\) elements. The evaluation information in the conceptual model, i.e. \(ESD\), is contained
in the ASSERTTypeSpecific element, while the UDE is captured in its namesake element UserDefinedExtensions. The ASSERTCore element contains, in addition to the SD and SPS, elements such as ServiceBinding that provides a robust link between the service and its ASSERT, CertificationProcess that provides information related to the certification process of a given service, and a textual description of the certificate in the ASSERTHumans element where the certified service and the certified property are explained in natural language for end-user comprehension. The AssertSigner element identifies the entity that signs the ASSERT, while the PerformedBy element in the CertificationProcess identifies the entity who performed the service evaluation. Since multiple entities can be involved in a certification process, for example the ASSERT issuing process and service evaluation process may be undertaken by different entities, we provide this feasibility so as to increase the accountability during the production of certificates. In order to better illustrate the ASSERT language we provide code excerpts from the ASSERT of the example we provided in Section 2.

5.2.1 Service Description in ASSERT Core
The SD in the conceptual model is mapped to the CertificationProcess element in the ASSERT language. It contains the elements such as TargetOfCertification and SecurityProblemDefinition which map to the TOC and SPD respectively in the conceptual model. In addition, we have incorporated an element called CertificationCriteria used to represent any specific criteria followed during the service certification process (e.g., compliance to regulations).

The TargetOfCertification element is depicted in Figure 3. The elements in the ACI are represented directly in the TargetOfCertification element i.e., the Assets, TOCCOMPONENTS. It also contains the Type, DeploymentAndImplementationModel and Description providing textual description of the TargetOfCertification for end-user comprehension. We enforce the explicit identification of both the Assets and TOCCOMPONENTS by making the use of the ID element mandatory. The set that maps assets with components, α, in the ACI is realized within the asset definition by mapping each asset to specific components using the TOCComponentRef (which is of type IDRef) to provide a binding between the assets and components.

The TOE is not represented as an explicit part of the service description in the ASSERTCore, but we use the flag InTargetOfEvaluation in the TOCCOMPONENT element that indicates whether the component is a part of the TOE, and avoids a duplicate representation of the components in both the TOE and TOC to have an optimized ASSERT.

5.2.2 Security Property Specification in ASSERT Core
The SecurityProperty element maps to the p element in the conceptual model. However, on the representation (language) level we have defined a single property certified
in ASSERT. Such “separation” of certified properties allow us to have practical implications on management of ASSERTs throughout their life-cycle, such as generation, consumption (reasoning), and revocation of ASSERTs. For example, if an ASSERT certifies two properties, say “confidentiality in transit” and “confidentiality in storage”, and during the ASSERT lifetime the given service does not anymore comply/provide “confidentiality in transit” due to some technical reasons, the certification authority has to revoke the ASSERT although the second property may still hold.

The SecurityProblemDefinition element in the ASSERT-Core contains a list of ProblemDefinition, as shown in Figure 4. Each ProblemDefinition is mapped to the spd in the conceptual model. However, instead of a set of assets, the ProblemDefinition contains a list of references to the Assets defined in the TargetOfCertification section. Figure 5 shows the SecurityProperty element structure consisting of an abstract security property realized in a context and on a set of assets. The SecurityProperty contains a NameID that defines a name identifier of the described property. The NameID allows reference to external ontologies to describe the certified security property. The PropertyAbstractCategory defines the abstract category of the security property. The PropertyContext element defines a context in which the abstract security property is realized. The Assets defines a set of Asset elements on which the security property applies. Each Asset element is a reference to an Asset definition in the TargetOfCertification section. The SecurityObjectives defines a set of SecurityObjective elements of the security property. The main elements of the SecurityObjective are: a) an identifier of the described security objective; b) a set of SecurityProblemDefinitionRef each referring to a ProblemDefinition defined in the SecurityProblemDefinition section; c) Name that contains the name of the security objective; d) Description which describes the security objective. It is an implicit assumption that all SecurityObjectives together contribute to the realization of the SecurityProperty. A SecurityObjective can refer to one or more ProblemDefinitions.

The SecurityMechanisms defines a set of Security-Mechanism elements. Each element consists of an ID that identifies the security mechanism, the Type of the security mechanism (or the family of the security mechanism), a set of SecurityObjectiveRef elements each referring to a security objective that the security mechanism corresponds to. A SecurityMechanism can refer to one or more SecurityObjectives.

6. SEMANTIC COMPLEXITY AND INCONSISTENCIES DUE TO THE EXPRESSIVITY

Since the ASSERT language is designed in a way that it is security certification scheme agnostic, there are major issues that can arise out of that design choice:

- Facilitate comparison among security certificates. Given the flexibility and richness of certificate languages and ability to express similar security assertions in different ways, a certification authority may wish to define a certificate profile (e.g., by defining various certificate structure and content mandatory) to enforce uniformity of content of certificates when issued by accredited entities.

- Facilitate production of security certificates compliant with specific certification criteria. Given that a certificate language can support various certification schemes, a certification authority has to define its
certification criteria in a certificate profile, so that all issued security certificates will conform to the criteria defined by the certificate profile.

- Enable consumers to specify their security requirements for the services. Similarly to CC-PP, the consumers or consumer groups may wish to define a certificate profile with domain-specific security requirements (criteria). When services conform to such certificate profiles, it eases the decision making process for the consumers as the conformance to a profile implies that their requirements are met by the service.

7. Digital Security Certificate Profile

A DSC profile is a mechanism to specify the contents and semantics of a class of DSCs. The main goal of a DSC profile is to provide suitable means for creation of certificates by ensuring semantic uniformity of certificates for a specific (domain of) certification capturing any certification and evaluation specific aspects, vocabulary of products certified, security properties, or other aspects relevant to the semantics of DSCs.

7.1 Conceptual Model of DSC Profile

In the following, we will introduce the profile structure (Montenegro, Maña, & Koshutanski, 2013). A DSC profile is composed of three parts: (i) Certificate Template: specification of the common structure and the values of specific fields mandatory for a given certificate class, (ii) Semantic Rules: specification of the semantics of the certificate class in the form of semantic rules, and (iii) Vocabulary: specification of vocabulary terms (ideally ontology-referenced terms) providing restrictions on use of vocabulary for language artefacts of security certificates of the given certificate class.

Figure 6 shows the abstract structure of the DSC profile. The three profile components provide certificates content uniformity in three different dimensions: certificate template ensures structural uniformity; semantic rules ensure integrity of intended semantics of certification; while certificate vocabulary ensures common ontology-based ground of terms and ranges of possible values of certification (in a given domain).

The certificate template is a partially filled certificate that establishes the common structure and content of all certificates created based on a certificate profile. Therefore, any certificate conforming to a DSC profile must include the fields, structure and values defined in the template of the profile. A certificate template specifies an incomplete realization of a structure with respect to a given certificate syntax (e.g., XML schema). It is used as baseline for creating new certificates.

Alternatively, a certificate template can be considered as a set of implicit (semantic) rules. These rules are simple and easy to understand. For this reason, it is not required to represent a template as a set of rules, but used as a certificate template – a more intuitive notion for expressing predefined structure and values of certificate elements.

The Semantic Rules define semantic constraints and dependencies between content of certificate artefacts within a given class of DSCs. While the implicit rules defined by the certificate template are enough for structure-wise restrictions (requiring an optional element be mandatory, constraining specific structure or content of certificate artefacts, etc.), there are cases where more complex restrictions are needed such as to express artefact dependencies or artefact content constraints.

Semantic rules represent a solution allowing to formulate rules to ensure integrity of an intended semantics of a given certificate class, i.e., preserving specific semantics of certification artefacts. Semantic rules can be formulated in rule based languages (such as Schematron (Schematron, 2006) or variants of OCL (Object Constraint Language, 2012)) or imperative languages (such as Java or Javascript) in function of the underlying certificate language and supported implementation. The choice of a language for expressing semantic rules has an important implication to achieve machine processability and reasoning of the rules. The language should allow rich fine-grained expression of patterns over certificates’ content and structure.

The certificate vocabulary part of the profile provides a means to define and restrict use of vocabularies on different certificate artefacts. One of the goals of the vocabulary part is to facilitate integration of the certificate language with ontology terms coming from different domains of knowledge. The ontology integration will enhance the semantic robustness among all certificates conforming to a given profile, which have been diminished by flexibility and openness of security certificate languages (models). Ontologies provide not only a suitable source of semantically defined terms but also provide means to define relations between terms, and equivalences between different terms. That gives us a powerful way to query ontologies for different aspects of certification and related semantics. The certificate vocabulary section enables the use of static or dynamic vocabularies. A static vocabulary defines actual terms inside a profile. It is suitable for offline processing, but could be out-dated by an ontology evolution/update. In contrast, a dynamic vocabulary defines actual terms by means of a query over ontology, which requires Internet connection for online processing.
7.2 Representation of DSC Profile

We have realized a DSC profile structure tailored to the ASSERT representation, which we will refer to as ASSERT profile henceforth. We refer to (Koshutanski et al., 2013) for more details on the actual profile schema and in this section we provide the main aspects of a DSC Profile.

An example of an ASSERT profile structure is shown in Figure 7. The profile defines the following class of ASSERTS. The ASSERTTemplate defines that all ASSERTS conforming to this profile must: (i) Be for software-as-a-service (SaaS) model services, i.e., all ASSERTS must have TargetOfCertification element with an attribute Type qualified as “http://assert4soa.eu/ontology/-a4s-language#Software-as-a-service”; (ii) Be issued by the University of Malaga as issuer, i.e., all ASSERTS must have an ASSERTIssuer element with the defined value structure (conforming to X.509 subject structure) “O=University of Malaga,OU=Computer Science Department,C=ES”; (iii) Be produced by a test-based certification process, i.e. must contain ASSERT-E type-specific structure, but without defining any particular content for the structure. This means that ASSERTS stating conformance to the profile can contain any specific ASSERT-E content. The SemanticRules section defines one Schematron rule, which forces the security property abstract category value as defined in the SecurityProperty element in the ASSERTCore match the value of the PropertyName of Property definition of ASSERT-E.

Such an integrity constraint is difficult to enforce without a semantic rule. The ASSERTVocabulary defines two vocabularies – one for the PropertyAbstractCategory attribute of the SecurityProperty element, and another one for the PropertyContext attribute of the same SecurityProperty element. The first vocabulary defines static values of the CIA triad − Confidentiality, Integrity and Availability − as terms from an ontology specific definition, and marks those as mandatory. The second vocabulary defines optional values for the artefact PropertyContext, such as InStorage, InTransit and InUsage, as terms from an ontology-specific definition.

8. DSC MANAGEMENT TOOL

We will describe the necessary tool support for the core profile-based certificate management operations: profile-based creation of DSCs, and profile conformance verification of DSCs. These are the most relevant DSC management operations a DSC issuer would need to perform when issuing DSCs.

8.1 Profile-based Generation of DSC

Figure 8 shows the profile-based creation process. When a profile is selected and loaded, there is a pre-processing step for all dynamic vocabulary specifications. If some dynamic vocabulary specifications depend on other artefacts and values in order to be processed, these vocabularies are processed at the time when the issuer creates the corresponding artefacts.

Step 1: Initialize DSC content. Once the profile is processed, first duplicate the certificate template and create an initial certificate instance with an initial structure and content of the duplicated template data.

Step 2: Edit DSC content. Next step is the actual process of editing the certificate artefacts and creation of new artefacts as needed by the issuer. This step heavily relies on the use of certificate vocabulary defined in the profile. When an artefact’s vocabulary is specified as mandatory, the tool will enforce the choice of the vocabulary terms. Otherwise, if optional, the tool will recommend, suggest a choice of terms but leaving the issuer to specify own terms when he finds necessary.

Step 3: Profile Conformance Verification. The final certificate instance is verified for conformance to the profile (presented in the next subsection). All non-properly used artefacts and corresponding vocabularies will be reported. Step 3 will give a feedback to redo step 2 of the creation process. Repeat step 3 until the certificate instance conforms to the profile.
8.2 Profile Conformance Verification of DSC

A prerequisite to conformance verification is to ensure if the certificate instance conforms to the syntax of a given certificate model, that is, if the certificate instance is a syntactically valid certificate. Otherwise, the verifier should not proceed with the verification process. Figure 9 shows the three main steps of conformance verification process.

**Step 1:** Structure validation. DSC structure is validated if it contains all required elements and values as declared in the profile template.

**Step 2:** Vocabulary Validation. DSC vocabulary is validated for compliance with the vocabulary defined in the profile. Prerequisite to this step is to first process all dynamic vocabularies. That is, retrieving all certificate artefacts’ vocabulary terms from the corresponding ontologies by executing the queries. Once dynamic vocabularies are instantiated, all certificate artefacts’ vocabulary terms within the vocabulary part are checked against the corresponding artefacts’ content in the certificate instance. All certificate artefacts defined to have an optional (non-mandatory) vocabulary will not be verified for conformance.

**Step 3:** Semantic Validation. DSC structure and vocabulary is validated for compliance with the profile rules, that is, if all constraints are satisfied. All semantic rules are processed, checked if satisfied by the certificate structure and content. Since the semantic rules of the profile may depend on the actual content (vocabulary) of a certificate artefacts in order to determine the semantic integrity of the certificate content, it is important to verify vocabulary conformance first, and then the semantic rules conformance.

8.3 DSC Management Tool Realization

A DSC management tool has been developed to support the management of ASSERTS from the perspective of ASSERT issuers. The tool is called ASSERT Management Tool* (AMT). The AMT is designed to provide an intuitive and formative GUI for standard ASSERT management operations including complete support of multi-profile based ASSERT management. We will present the GUI of the AMT for the case of profile-based creation of ASSERTS.

Figure 10 shows the AMT designer view. The designer view provides content-centric ASSERT management: abstracting issuers from underlying XML representation. All mandatory ASSERT language elements are coloured in red for convenience of issuers. A help icon is shown to each element name describing the rationale of the selected element. The AMT has the ASSERT language schema built-in. A designer pane shows all language elements as buttons. Upon pressing a button, the corresponding data element is created. As we can see in Figure 10, the ASSERT profile has been already loaded and the corresponding profile vocabulary processed showing the vocabulary terms defined for the PropertyContext element of the SecurityProperty (ref. Figure 7). We note that in case of multiple profiles, the AMT loads each profile in a separate profile view (tab) so that the issuer can at any moment check if any of the profiles is well-formed and what messages are shown of profile validation.

9. IMPACT ON CURRENT SECURITY CERTIFICATION PRACTICES

Current security certification schemes are a result of harmonizing the security assurance requirements by security experts, government regulatory bodies and they provide a broad consensus on the assessment of software security in a manner that is both repeatable and observable. Each certification scheme has at its own criteria, chosen carefully so that it can cater to evaluating products that range from software, firmware to hardware. However, a detailed analysis of the existing criteria must be performed to verify if the current criteria is capable of providing assurance for services or it needs to be adapted. This analysis is outside the scope of this paper, while here we focus on the security certification practice. The current security certification practices would be profoundly impacted by the concept we proposed. The model allows certification results to be captured in a machine processable form thereby allowing automated reasoning to be performed on them. Automated reasoning on security certificates opens up hitherto unexplored scenarios such as instantaneous comparison of certified products, requirements compliance by certified products, fine grained and precise description of secured

---

* Available at http://proteus.lcc.uma.es/projects/assert4soa/software/

---

http://hipore.com/ijsc
elements within a certified product leading to less ambiguity or errors in interpretation of the security certificates.

In addition, the AMT tool can be used by the certification authorities to issue security certificates in a much faster manner reducing the previously time consuming process of producing certificate artefacts. In fact, the AMT tool and its output artefacts were evaluated by a focus group composed of domain experts, two Common Criteria experts, and two experts in security certification. The activities comprised an explanation of the basic underpinnings for DSC, ASSERT, and ASSERT Profile concepts, as well as an evaluation of the tool operations and outputs. The evaluation consisted of a questionnaire composed mostly by closed-answer questions using 5-point Likert scale, as well as a number of free answers questions. The latter allowed for the expression of feedback on specific DSC aspects, which was used later on for improving the AMT tool. The questionnaire was structured in two sections: the first part was devoted to usability assessment (Rubin & Chisnell, 2008), while the second aimed at assessing the suitability of AMT tool outputs and procedures (e.g. ASSERTs and ASSERT validation against an ASSERT Profile) for their application in a security certification process operations.

An analysis of focus group results can be summarized as follows. Regarding the usability of the AMT tool, it was assessed positively, especially considering its nature of research prototype. Nevertheless, several suggestions were proposed, and some of them have been incorporated in the prototype; others were deemed interesting for future commercial exploitation of the concept. With respect to the suitability of ASSERTs and AMT tool operations to security certification operations (especially considering Common Criteria), we will focus our attention on a selection of the focus group questions, which are:

- Q-15: From the point of view of a Certifier, an ASSERT is suitable to represent a typical security certificate.
- Q-16: The AMT speeds up the ASSERT management process.
- Q-17: The AMT improves the control over the ASSERT management process.
- Q-18: From the point of view of a Certifier, the automation of a typical security certificate management process is a priority.

Their proposed answers (and their mapping to the 5-point Likert scale) were: I fully agree (2), I mostly agree (1), I neither agree nor disagree (0), I mostly disagree” (-1) and I completely disagree” (-2). The answers are represented in Figure 11.

The answers to Q-15, Q-16 and Q-17 are particularly encouraging; Q-15 essentially confirms the suitability of the ASSERT and DSC concepts for representing security certificate contents, while Q-16 and Q-17 assess positively the AMT tool operations.

From Q-18 and open answer questions, it is possible to derive the following findings.

From the point of view of the technical quality of the AMT, most of the respondents generally agreed that the representation capability of an ASSERT is suitable from the certifier’s point of view. However, from the point of view of its adoption and relevance, the main output of the validation session indicates that current real world certification processes (and especially Common Criteria) are probably not ready to embrace digital certificates in their current forms. For Common Criteria, this is mainly due to its complexity and difficulty to automate its operations. However, the advantages of using ASSERT and DSC Tool can represent a stimulus for the evolution of a debate inside the Common Criteria community.

10. CONCLUSION

To conclude, we claim that the introduction of the AMT can represent
significant benefits for an uptake of third-party service offerings in business-critical domains such as financial, defence and healthcare (Di Cerbo et al., 2012). The digital security certificates provide security assurance of services that would allay the security concerns of potential customers, which is one of the most relevant obstacles nowadays (Gartner, 2012). In addition, we presented the concept of a DSC Profile that helps in the generation of uniform CRTs. Finally, we presented a tool that helps in the generation of the CRTs by a Certificate Issuer.

11. ACKNOWLEDGMENTS

This work was partially supported by the EU-funded projects ASSERT4SOA (grant no. 257361), CUMULUS (grant no. 318580) and ANIKETOS (grant no. 257930) of Framework Programme VII of the European Commission. Portion of this paper appeared under the title "Security assurance of services through digital security certificates" in Proceedings of the 20th IEEE International Conference on Web Services (ICWS-2013), Santa Clara Marriott, CA, USA, June 2013, pp. 539—546.

12. REFERENCES


Authors

Samuel Paul Kaluvuri is a Research Associate in SAP Research at Sophia Antipolis, France. He is also pursuing a PhD from Telecom Paritch, Paris. Samuel received his Masters in Business Studies in Electronic Commerce from National University of Ireland, Galway. He has participated in SAP’s contribution to FP7 European projects PrimeLife, Avantssar, Assert4SOA and Aniketos.

Dr. Hristo Koshutanski received M.Sc. in Mathematics from Plovdiv University "Paisii Hilendarski" in 2001 and Ph.D. in Information and Communication Technology from the University of Trento in 2005. He had a postdoctoral position in the field of security and trust management at CREATE-NET international research centre for 2005-2006. He holds EU Marie Curie EIF postdoctoral fellowship with a host institution the University of Malaga for 2007-2009. He has participated in several European projects in both Sixth Framework Program (FP6) and Seventh Framework Program (FP7) of the European Commission. He was a member of advisory board/interest group of the European project ONE-Open Negotiation Environment (FP6), and is an external ethics advisor of the European project SUPERHUB (FP7). His research interests span across: (i) Models of authorization, access control and trust management for service-based systems, business processes and dynamic coalitions; (ii) Security architectures for distributed, federated and cloud-based systems; (iii) Security assurance, certification and digital security certificates of cloud and Web services; (iv) Identity management in federated systems, (v) Public Key Infrastructures and Privilege Management Infrastructures (PKI/PMI). He has co-authored a number of scientific papers.

Dr. Francesco Di Cerbo received his PhD in 2008 from University of Genoa, Italy. In 2011, he joined SAP Research, in the Security & Trust practice. Previously, he was assistant professor at the Faculty of Computer Science at Free University of Bolzano-Bozen. He is author of a number of contributions to international ACM and IEEE conferences; he also served as Programme Committee member of conferences, and as reviewer for a number of journals. He was research associate for the Italian National Research Council. His current interests include mobile security, trust management and usage control mechanisms in SOA and cloud environments.

Renato Menicocci graduated with a summa cum laude degree in Electronic Engineering at the University of Rome “La Sapienza” in July 1991 with a thesis on Cryptanalysis. From 1992 to 2000, he was a Researcher of the Cryptology Group of Fondazione Ugo Bordoni (FUB). From 1996 to 1999, he was the Italian delegate at ISO/JTC1 SC27 WG2. From 2000 to 2002, he was a Cryptography and Computer Security Expert of the Rome Crypto Design R&D Centre of Gemplus. From 2003 to 2008, he was a FUB consultant for Digital Television Security and ITSEC/Common Criteria Security Evaluation. Since August 2004, he’s been cooperating with the Department of Electronic Engineering, University of Rome “La Sapienza”, where, from 2004 to 2006, he was under a research fellowship connected to SCARD (EU-FP6). Since 2008 he is a researcher of FUB, where his main interests are ICT Security Evaluation and Certification, especially under Common Criteria. He has been involved in ASSERT4SOA (EU-FP7) and is currently involved in CUMULUS (EU-FP7). He has been involved as a teacher in several academic contexts. He is author or co-author of several scientific works published in international journals.

Prof. Antonio Maña received his MSc and PhD degrees in Computer Engineering from the University of Malaga in 1994 and 2003, respectively. In 1995 he joined the Department of Computer Science of the University of Malaga where he is currently Professor in the Computer Science Department. His current research activities include security and software engineering, information and network security, application of smart cards to digital contents commerce, software protection and DRM. He has more than 100 peer-reviewed publications. He has participated in several EU funded projects and is currently the Principal Investigator of the UMA team in FP7 PARIS and CUMULUS projects, and has previously been involved in the FP7 ASSERT4SOA, SECFUTUR and OKKAM, FP6 SERENITY, GST and iAccess, and FP5 CASENET projects. Prof. Maña is member of the Editorial Board of the Intl. Journal of Electronic Security and Digital Forensics (IJCSEDF) and reviewer for several other international journals. He is member of different professional and scientific societies and workgroups and is actively involved in the organisation of research and educational activities.