Initial SbD methods for multi-cloud applications

Abstract:
This deliverable includes the initial description of the mechanisms and tools that MUSA offers to define the security constraints of a multi-cloud application, in order to support a security-by-design (SbD) development process. Since the adopted SbD approach is based on Security SLAs as a means to express the security requirements of a multi-cloud application and of each of its components, the deliverable provides a discussion of the Security SLA model adopted in MUSA and illustrates the SLA Generation process, aimed at building components’ Security SLAs. In order to support this process, the document introduces an SLA Generation proof-of-concept application, which applies a risk categorization and assessment procedure to each component and uses state-of-art Security Control Frameworks for the identification of their security requirements.
MUSA consortium

Fundación Tecnalia Research & Innovation (TECNALIA, Spain)
www.tecnalia.com/en

Centro Regionale Information e Communication Technology (CER ICT, Italy)

CA Technologies Development Spain SAU (CA, Spain)

Montimage (MI, France)

AIMES Grid Services (AIMES, UK)

Lufthansa Systems (LSY, Germany)

TTY-säätiö (TUT, Finland)

Project manager: Erkuden Rios
erkuden.rios@tecnalia.com
+34 664 100 348

Contact: Massimiliano Rak
massimiliano.rak@unina2.it

Contact: Victor Muntés
Victor.Muntes@ca.com

Contact: Edgardo Montes de Oca
edgardo.montesdeoca@montimage.com

Contact: Prof Dennis Kehoe
dennis.kehoe@aimes.net

Contact: Dirk Muthig
dirk.muthig@lhsystems.com

Contact: José Luis Martínez Lastra
jose.lastra@tut.fi
Table of contents

MUSA consortium ................................................................. 3
Table of contents ................................................................. 4
List of figures ......................................................................... 5
List of tables ........................................................................... 6
Executive summary ................................................................. 7
1 Introduction ........................................................................ 8
  1.1 Objective of this document .............................................. 8
  1.2 Structure of this document .............................................. 8
  1.3 Relationships with other deliverables ............................... 8
  1.4 Contributors .................................................................... 9
  1.5 Acronyms and abbreviations .......................................... 9
  1.6 Revision history ............................................................. 9
2 Multi-cloud applications ....................................................... 10
  2.1 Multi-cloud application definition .................................... 10
  2.2 Existing multi-cloud approaches ..................................... 10
  2.3 Multi-cloud applications in MUSA .................................. 12
    2.3.1 Multi-cloud application deployment scenarios .............. 13
      2.3.1.1 Deployment scenario 1: a single IaaS CSP ............ 15
      2.3.1.2 Deployment scenario 2: multiple IaaS CSPs .......... 16
      2.3.1.3 Deployment scenario 3: multiple IaaS and SaaS CSPs 17
  2.4 Security SLAs for cloud and multi-cloud applications ......... 18
3 MUSA initial methods and tools ............................................ 20
  3.1 Multi-cloud application design methods ......................... 20
    3.1.1 Conceptual models ............................................... 21
      3.1.1.1 The SPECS Security SLA model ....................... 21
      3.1.1.2 Security SLAs and Threat Modelling ................ 22
      3.1.1.3 The MUSA Security SLA model and the adopted SbD approach 23
    3.1.2 Design process ...................................................... 23
      3.1.2.1 The design process in the MUSA Framework ......... 25
  3.2 The SLA Generation process ........................................... 27
  3.3 Supporting tools and modules for SLA Generation ............. 28
    3.3.1 MUSA Security Metric Catalogue ............................ 28
    3.3.2 Discussion on Risk Analysis .................................. 29
  3.4 Security SLA Generation: proof-of-concept ..................... 30
    3.4.1 Design analysis .................................................... 30
    3.4.2 Risk Analysis ....................................................... 31
    3.4.3 SLA Generation .................................................... 31
  3.5 Examples on MUSA case studies ...................................... 31
4 Requirement coverage discussion ........................................ 34
  4.1 MUSA SLA Generation .................................................. 34
5 Conclusion and further work ............................................... 36
References ............................................................................. 37
Appendix A. MUSA Proof-of-concept SLA Generator Threat Catalogue ................................................. 41
Appendix B. The TSM Engine component Security SLA ................................................................. 49
List of figures

Figure 1: MUSA multi-cloud application model ................................................................. 12
Figure 2: Lufthansa Systems’ Flight Scheduling multi-cloud application ............................. 14
Figure 3: TUT’s Tampere Smart Mobility multi-cloud application ........................................ 15
Figure 4: Deployment scenario 1 .......................................................................................... 15
Figure 5: TUT’s multi-cloud application: deployment scenario 1 ........................................... 16
Figure 6: Deployment scenario 2 .......................................................................................... 16
Figure 7: Lufthansa Systems’ multi-cloud application: deployment scenario 2 ..................... 17
Figure 8: Deployment scenario 3 .......................................................................................... 17
Figure 9: TUT’s multi-cloud application: deployment scenario 3 .......................................... 18
Figure 10: The SPECS Security SLA model ........................................................................... 21
Figure 11: Security Controls and Threat Analysis .................................................................. 23
Figure 12: The MUSA Security SLA model ........................................................................... 23
Figure 13: Security SLA-based SbD development process activities ..................................... 24
Figure 14: MUSA Design Process collaboration model ......................................................... 26
Figure 15: Details of the SLA Generation process ................................................................. 27
Figure 16: Selecting the threats associated with the TSM Engine component ....................... 32
Figure 17: Assessing the risk for the TSM Engine component .............................................. 33
Figure 18: Selecting metrics and defining SLOs for the TSM Engine component .................. 33
List of tables

Table 1. Summary of the main multi-cloud application concepts ........................................... 12
Table 2. Security metric specification format ............................................................................. 28
Table 3. Examples of metrics ...................................................................................................... 29
Table 4. Requirements for the MUSA SLA Generator ................................................................. 34
Table 5. Summary of MUSA SLA Generator requirements and coverage ................................. 35
Table 6. Threat catalogue used by the proof-of-concept SLA generator ................................. 41
Executive summary

This deliverable includes the initial description of the mechanisms and tools that MUSA offers to define the security constraints of a multi-cloud application, in order to support a security-by-design development process.

In particular, this document focuses on the *SLA Generation* activity, which consists in identifying the security requirements of each component of the multi-cloud application and in generating a Security SLA for each of them. The Security SLAs obtained as result of this activity include the security guarantees that should be granted by each component’s implementation in order to fulfil elicited requirements. The security requirements of the components are obtained by performing a risk analysis process, aimed at identifying the main vulnerabilities and threats the components are subject to based on their nature and behaviour.

This deliverable introduces the Security SLA model used by MUSA to express security-related needs and guarantees related to a component or application. This model has been borrowed by the SPECS EU project and has been extended in MUSA to express the security requirements of the multi-cloud application components and of the multi-cloud application itself. In addition to the Security SLA model, MUSA also adopts the SPECS Security Metric Catalogue, which contains relevant security metrics that can be used to monitor the fulfilment of a security-oriented agreement from the customer’s point of view. The metric catalogue has been enriched with MUSA-specific metrics, suggested and validated by Industrial Partners and related to MUSA case studies.

Finally, the deliverable discusses a proof-of-concept application that enables to carry out the whole SLA Generation phase based on a simplified risk analysis process that uses common threat analysis and risk assessment tools and procedures.

In future deliverables, the other activities of the design process will be addressed, such as the multi-cloud application modelling, the verification of the feasibility of resulting Security SLAs, the Security SLA composition as well as the implementation and integration of security libraries needed to provide all requested security features at runtime.
1 Introduction

1.1 Objective of this document

This document is the deliverable entitled *D2.1: Initial SbD methods for multi-cloud applications* of the MUSA Project.

The document provides a description of the preliminary mechanisms and tools that MUSA offers to define the security constraints in multi-cloud application components.

The main objective of this document is to introduce the initial security-by-design (SbD) methods and tools adopted to support security-aware design of multi-cloud application, in order to address the project Objective SO2. Therefore, we will focus on the following main objectives:

- Develop formalisms and methods for the specification of combined security properties in multi-cloud application SLAs.
- Design and implement the methods and tools for the security aware analysis and design of multi-cloud application components.

1.2 Structure of this document

Section 2 provides a deep analysis of the state of art of multi-cloud applications in general and of the associated security issues. Moreover, this section includes the model adopted in MUSA for multi-cloud applications and a set of scenarios that illustrates how multi-cloud applications behave in practice. The section terminates illustrating the role of SLAs in multi-cloud application security.

Section 3 focuses on MUSA methods, tools and techniques for developing secure multi-cloud applications. In particular, the section introduces the methodology adopted to develop secure multi-cloud applications and illustrates the underlying conceptual models related to Security SLAs and to their relationship with threats and threat modelling. Moreover, the section gives details on the SLA Generation activity of the design process and on the tools supporting this activity. Finally, the section discusses a proof-of-concept application that aims at implementing the proposed methodology and at identifying its concrete applicability, by also providing an example of its application to part of MUSA case studies.

Section 4 identifies the requirements related to the SLA Generation activity among those presented in Deliverable D1.1, and discusses their coverage with the current proof-of-concept application.

The document ends with Section 5 summarizing conclusions and future works.

Moreover, the document includes two appendices:

- Appendix A illustrates one of the tools used by the proof-of-concept SLA Generation application, namely the Threat Catalogue.
- Appendix B reports an example of Security SLA generated for a component of the TUT case study application, discussed in Section 3.5.

1.3 Relationships with other deliverables

This deliverable contains the initial methods and tools developed in the context of the MUSA framework.

It assumes as an input the requirement analysis and the architecture defined in *D1.1 Initial MUSA framework specification* and refines such results mainly with respect to the design process, whose overview is also provided in *D1.2 Guide to security management in multi-cloud applications lifecycle*, and in particular to its SLA Generation activity. The development of the methods and tools will continue throughout the duration of the project and the advances and final results will be proposed in deliverables *D2.3 Final SbD methods for multi-cloud applications*, *D2.2 Initial MUSA IDE for*
security-aware design of multi-cloud applications and D3.1 Initial security based discovery and composition mechanisms and tools.

1.4 Contributors

The following partners have contributed to this deliverable:

- CA: risk analysis activity definition;
- Tecnalia, MI, TUT, LFY: multi-cloud application deployment scenarios definition, security metrics validation, example of SLA for case study generation.
- Tecnalia: multi-cloud application model definition and alignment with D1.1.

1.5 Acronyms and abbreviations

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPIM</td>
<td>Cloud Provider Independent Model</td>
</tr>
<tr>
<td>CSP</td>
<td>Cloud Service Provider</td>
</tr>
<tr>
<td>DST</td>
<td>Decision Support Tool</td>
</tr>
<tr>
<td>IaaS</td>
<td>Infrastructure-as-a-Service</td>
</tr>
<tr>
<td>PaaS</td>
<td>Platform-as-a-Service</td>
</tr>
<tr>
<td>SaaS</td>
<td>Software-as-a-Service</td>
</tr>
<tr>
<td>SbD</td>
<td>Security-by-design</td>
</tr>
<tr>
<td>SLA</td>
<td>Service Level Agreement</td>
</tr>
<tr>
<td>SLO</td>
<td>Service Level Objective</td>
</tr>
</tbody>
</table>

1.6 Revision history

<table>
<thead>
<tr>
<th>Version</th>
<th>Date issued</th>
<th>Author</th>
<th>Organisation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>15/02/2016</td>
<td>Massimiliano Rak</td>
<td>CeRICT</td>
<td>Initial draft (D2.1_draft_v1.docx)</td>
</tr>
<tr>
<td>0.2</td>
<td>06/04/2016</td>
<td>Massimiliano Rak</td>
<td>CeRICT</td>
<td>Initial completed draft (D2.1_draft_v2.docx)</td>
</tr>
<tr>
<td>0.3</td>
<td>07/04/2016</td>
<td>Erkuden Rios</td>
<td>Tecnalia</td>
<td>Review and comments (on D2.1_draft_v2.docx)</td>
</tr>
<tr>
<td>0.4</td>
<td>12/04/2016</td>
<td>Alessandra De Benedictis</td>
<td>CeRICT</td>
<td>Creation of D2.1_draft_v3.docx and subsequent updates based on first review</td>
</tr>
<tr>
<td>1.0a</td>
<td>20/04/2016</td>
<td>Alessandra De Benedictis</td>
<td>CeRICT</td>
<td>Creation of D2.1_draft_to_review.docx submitted to internal review</td>
</tr>
<tr>
<td>1.0</td>
<td>28/04/2016</td>
<td>Alessandra De Benedictis</td>
<td>CeRICT</td>
<td>Creation of final revision: D2.1_final.docx and D2.1_final_comments.docx</td>
</tr>
<tr>
<td>1.0</td>
<td>29/04/2016</td>
<td>Erkuden Rios</td>
<td>Tecnalia</td>
<td>Final release</td>
</tr>
</tbody>
</table>
2 Multi-cloud applications

This section focuses on multi-cloud applications: it provides a discussion of the state of the art on multi-cloud solutions and describes the multi-cloud application model adopted in MUSA, by also discussing a set of deployment scenarios that illustrate how multi-cloud applications may look like in practice. Finally, the section discusses the adoption of Security SLAs as a means to manage cloud and multi-cloud security.

2.1 Multi-cloud application definition

At state of art, the term multi-cloud is used in many different contexts and refers to the idea of accessing resources from different Cloud Service Providers (CSPs). As such, multi-cloud can be considered as a special case of inter-cloud computing, which has been defined in [24] as:

A cloud model that, for the purpose of guaranteeing service quality, such as the performance and availability of each service, allows on-demand reassignment of resources and transfer of workload through an interworking of cloud systems of different cloud providers based on coordination of each consumer’s requirement for service quality with each provider’s SLA and use of standard interfaces.

Even if the terminology is not yet stable, [24] and [25] propose to adopt the term inter-cloud as the generic term indicating the adoption of multiple CSPs, the term cloud federation to describe a set of cloud providers that voluntarily interconnect their infrastructures to allow sharing of resources, and the term multi-cloud whenever the usage of cloud services from different CSPs is involved without an explicit agreement with the service providers.

Indeed, it is important to distinguish the multi-cloud approach from the concept of cloud federations: in the first case, the usage of multiple independent cloud providers happens from the customer side and CSPs are not aware of it, while in the latter CSPs actively cooperate in the federation.

It is worth noticing that, with respect to the above discussed terminology, in the context of MUSA we focus on multi-cloud applications, i.e., MUSA aims at developing applications that use resources from multiple providers without requiring a direct involvement of CSPs in such a process.

2.2 Existing multi-cloud approaches

Even if, at state of the art, few concrete multi-cloud solutions exists, the topic is considered extremely relevant: the need for multi-cloud solution is well demonstrated by the number of research projects that are proposing solutions and techniques to address the multi-cloud and/or the federation approaches, like OPTIMIS [34], mOSAIC [33], MODAClouds [47], PaaSage [35], Cloud4SOA [43], SeaClouds [44] and MUSA. An interesting overview of the research activities and innovative results can be found in [2], [14], [22], which offer good surveys on the multi-cloud topic.

The authors of [2] outline that the (multi-)cloud application development can be made either (i) in terms of composition of cloud services at software-as-a-service level, which is equivalent to the service orchestration in well-known service oriented architectures, or (ii) as a composition of infrastructure services, which is the actual multi-cloud application deployment. At state of art, the most common solutions focus on the composition of infrastructure and platform services and even projects like MODAClouds, Cloud4SOA or PaaSage aim at deploying multi-cloud application components over multiple CSPs.

Two main approaches exist for the management of multi-cloud solutions, i.e., broker-based and proxy-based.

Broker-based solutions rely on technologies like deltacloud, libcloud, jclouds (a good overview is available in [23]), which enable to write software for acquiring and providing cloud resources from multiple vendors. Such an approach assumes that CSPs offer IaaS cloud services and the application
D2.1: Initial SbD methods for multi-cloud applications

assumes a multi-cloud behaviour due to the concurrent deployment over virtual machines hosted by different CSPs. Brokers can be integrated into the multi-cloud applications and/or offered as a service. Examples of brokering solutions are available in the mOSAIC, Broker@Cloud [46], Cloud@Home [45] and Cloud4SOA projects. Interesting research papers on the topic are represented by [6], [10], [13], [17], and [23].

Proxy-based solutions rely on the development of a dedicated daemon, the proxy, which acts as a gateway and offers a homogeneous interface toward different CSPs. This technique is commonly adopted in the federated solutions. Good surveys and examples of proxy based solutions are available in [7], [11], [12], [18] and [20].

Proxy based solutions rely on the idea of delegation: the delegator transfers its signing rights and capabilities to another entity called the proxy. In a cloud federation, the cloud customer accesses the federation starting from one of the CSPs, which authenticates and authorizes the customer, and delegates to a proxy when there is need to access to resources hosted on a different CSP.

The authors of [18] propose the following types of proxy-based solutions:

- Cloud-hosted proxy: each CSP can host proxies within its cloud infrastructure, manage all proxies in its administrative domain and handle service requests from clients to use those proxies for collaboration.
- On-premises proxy: a client can host proxies within its organization’s infrastructure (or on premises) and manage all proxies within its administrative domain. A client that wishes to use proxies for collaboration will deploy its on-premises proxies.
- Peer-to-peer proxy: proxies can interact in a peer-to-peer network managed by either a Proxy Service Provider (PSP) or a group of CSPs that want to collaborate. Each proxy in the peer-to-peer network is an independent entity that manages itself and handles requests to use its own services.
- Proxy-as-a-Service: proxies can be deployed as autonomous cloud providers that offer collaborative services to clients and CSPs. A group of CSPs that are willing to collaborate can manage this proxy in an as-a-service fashion, or they can use a PSP for the management. Clients can directly subscribe to the proxy cloud service and use it for inter-cloud collaboration.
- Hybrid Proxy: a hybrid infrastructure can include on-premises, CSP and PSP-maintained, and peer-to-peer proxies. Selecting proxies for collaboration will depend on the type of service being requested and the entity that initiates collaboration, among other factors.

In a part of the existing literature, scientists claim that relying on multi-cloud solutions can improve security. Others, on the contrary, believe that the multi-cloud paradigm will bring new security risks and vulnerabilities.

The authors of [1] and [3] offer simple surveys of solutions that try to improve the security using multi-cloud techniques: as an example, [16] and [21] propose techniques to distribute a storage service over multiple providers or untrusted networks, granting higher confidentiality and data integrity. Other papers that propose concrete solutions for multi-cloud security and that focus on cloud storage services are [3], [5], [8], [9], and [15].

On the other hand, [4] and [19] face the security in multi-cloud applications from a different perspective: they analyse different multi-cloud solutions and try to make a security assessment of the overall application behaviour. According to such vision, multi-cloud is open to new security threats that decrease the global security level.

In particular, [4] presented a four-type classification of the security-enhancing architectural approaches for multi-cloud applications, namely: replication of application tasks, partition of system
D2.1: Initial SbD methods for multi-cloud applications

Concrete examples of security issues due to multi-cloud approaches usually refer to the proxy-based solution: delegation approaches open new security issues that must be explicitly addressed. Authors of [11] and [18] illustrate some of the possible issues and solutions.

2.3 Multi-cloud applications in MUSA

In MUSA, a multi-cloud application is modelled as shown in Figure 1: the application (mc app in the figure) consists of one or more software components (mc app component in the figure), which can be executed independently and which interact with one another during the multi-cloud application execution. The application is “multi-cloud” in the sense that its components use SaaS (Software-as-a-Service) cloud services, are hosted by IaaS (Infrastructure-as-a-Service) cloud services and/or use PaaS (Platform-as-a-Service) cloud services offered by different CSPs.

As shown in Figure 1, a multi-cloud application is modelled by means of the MUSA Modeller component, which allows obtaining a CloudML-based specification of the application at different layers of abstraction. The CloudML specification can be found in [26]. The application is deployed on the target providers by the MUSA Deployer, which is responsible for the provisioning of the needed infrastructure resources and which may rely upon (use) specific PaaS cloud services to accomplish its tasks (e.g., Openshift [27]).

Table 1 summarizes the above concepts, providing related examples for clarity’s sake. In the following section, several deployment scenarios that implement this model are presented.

![Figure 1: MUSA multi-cloud application model](image)

<table>
<thead>
<tr>
<th>Concept</th>
<th>Definition</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>mc app</td>
<td>A cloud application that consumes cloud services from at least two different cloud service providers.</td>
<td>The Applications adopted as case studies in the project, i.e., the Tampere Smart Mobility</td>
</tr>
</tbody>
</table>
### D2.1: Initial SbD methods for multi-cloud applications

<table>
<thead>
<tr>
<th>mc app component</th>
<th>An independent software component, which cooperates with other components (commonly but not mandatory) by offering and consuming services.</th>
<th>A web application that runs on a virtual machine and invokes services from another component of the multi-cloud application and/or from an external provider.</th>
</tr>
</thead>
<tbody>
<tr>
<td>cloud service</td>
<td>One or more capabilities offered via the cloud computing paradigm, invoked using a defined interface. It can be of three different types: SaaS, IaaS or PaaS.</td>
<td>A storage service.</td>
</tr>
<tr>
<td>CSP</td>
<td>A Cloud Service Provider, which offers cloud services to its customers.</td>
<td>Amazon, Google, AIMES.</td>
</tr>
</tbody>
</table>

### 2.3.1 Multi-cloud application deployment scenarios

The multi-cloud application model presented in the previous section can be implemented through different deployment scenarios, which identify possible deployment and usage configurations for the multi-cloud application components.

In particular, as said before, the components of a multi-cloud application may be independent software components deployed on cloud IaaS resources (i.e., virtual machines), and/or they may rely upon SaaS services which implement needed functionalities. Moreover, the application may use PaaS services. The fundamental assumption is that the consumed cloud resources/services belong to different providers.

Based on this assumption and the analysis of the MUSA case studies in deliverable D5.1 *MUSA case studies work plan*, we identified three interesting preliminary deployment scenarios, involving respectively:

- The use of IaaS resources belonging to a single provider and of SaaS services offered by different providers.
- The use of multiple IaaS resources belonging to different providers;
- The use of multiple IaaS and SaaS resources belonging to different providers.

In the remainder of this section, we discuss such scenarios both in general and with reference to the two MUSA case study applications, namely the Lufthansa Systems’ *Netline/Sched Airline flight scheduling* multi-cloud application and the Tampere University of Technology’s *Tampere Smart mobility* multi-cloud application, whose architecture is reported in D5.1 and briefly summarized in the following.
The Netline/Sched flight scheduling application is shown in Figure 2. It consists of a web UI and a set of background services that realize the business functionality and expose HTTP REST interfaces. These modules are located behind a Central API gateway, which serves as service router, load balancer, service registry and security filter. The main backend components are the following:

- a Content Delivery Network (CDN) service, hosting the static components of the application,
- a fleet service module component, providing the service API and service implementation for fleet handling,
- an airport service module component, providing the service API and service implementation for airport handling,
- a schedule service module component, providing the service API and service implementation for schedule handling,
- a Message Broker component, acting as the command and event bus among the event sourcing-based service components,
- a Persistent storage component, representing the database to store the persisted status of the service modules, and
- a Security service, providing the application layer security functionalities.

In Figure 3, the architecture of the Tampere Smart Mobility (TSM) multi-cloud application by Tampere University of Technology (TUT) is depicted.
D2.1: Initial SbD methods for multi-cloud applications

Figure 3: TUT’s Tampere Smart Mobility multi-cloud application

The TSM application consists of the following components:

- A TSM Engine, in charge of handling the requests coming from a mobile application, intended to be run on mobile phones;
- A Journey Planner (JP), an application which provides multimodal and optimal journey options;
- A Consumption Calculator, which calculates the energy that is needed to complete every journey option;
- An Identity Manager/Access Manager (IDM), which handles user authentication and authorization;
- A Database, to hold all the information collected by the TSM application.

2.3.1.1 Deployment scenario 1: a single IaaS CSP

In the first deployment scenario, a single CSP offers all the infrastructure resources where the application components are deployed (see Figure 4), while one or more SaaS CSPs are used for the other needed services.

Figure 4: Deployment scenario 1

An example of implementation of this deployment scenario is given in Figure 5, which refers to a possible deployment of the Tampere Smart Mobility multi-cloud application. In this scenario, all the
multi-cloud application components are deployed on a single virtual machine (VM1 in figure), offered by an IaaS CSP (CSP1 in Figure 5), while multiple SaaS providers are used to acquire the services needed by the Journey Planner.

Figure 5: TUT’s multi-cloud application: deployment scenario 1

2.3.1.2 Deployment scenario 2: multiple IaaS CSPs

In the second deployment scenario, the multi-cloud application components are distributed among several virtual machines acquired from different IaaS CSPs, while no SaaS services are involved (see Figure 6).

Figure 6: Deployment scenario 2

An example of such deployment scenario is given in Figure 7 for the Lufthansa Systems’ application, whose components are deployed on the resources of two different providers.
2.3.1.3 Deployment scenario 3: multiple IaaS and SaaS CSPs

In the third considered deployment scenario, the multi-cloud application uses multiple IaaS providers to host its components and also relies upon SaaS services offered by multiple providers (see Figure 8).

An example of such deployment scenario is given in Figure 9 for the TSM application. In the depicted scenario, the database component is hosted on a virtual machine (VM2) offered by a CSP (CSP2), while the other three components are deployed on another virtual machine (VM1) offered by a different CSP (CSP1).
2.4 Security SLAs for cloud and multi-cloud applications

The adoption of the cloud computing paradigm has opened new business possibilities thanks to the virtual availability of huge computing resources at a low cost. However, at the same time, many potential users are still reluctant to move their critical data and applications to commercial clouds, due to a substantial lack of trust in providers for what regards security. Currently in fact, cloud security is still considered one of the factors inhibiting the widespread adoption of the cloud computing paradigm.

The need for guarantees, not only regarding security, has urged the introduction of Service Level Agreements (SLAs) in the cloud context [26], [29]. SLAs are formal contracts documenting the features of delivered services and related performance (and Quality of Service (QoS)) expectations, the so-called Service Level Objectives (SLOs). Moreover, they explicitly take into account responsibilities, obligations, service pricing and penalties in case of agreement violations.

Some public providers (e.g., Amazon) do currently offer SLAs for their services. These SLAs are typically written in natural language (often in a strict legal notation) and mostly include guarantees related to performance aspects, such as the availability or the response time of a service. Despite the strong interest in security and the existing efforts towards standardization in fact, security-oriented SLAs are still far from being adopted in reality. A shared format for Security SLAs including the representation of security attributes and security guarantees is not yet available [15, 16].

WS-Agreement (WSAG) [30], born in the context of GRID computing, is currently the only standard supporting both a formal representation of SLAs and a protocol for their automation, and has been recently widely adopted in the context of many cloud-oriented FP7 projects (e.g., SPECS [31], Contrail [32], mOSAIC, Optimis, PaaSage), to represent SLAs in the cloud environment. However, WSAG does not devise, by its original definition, the specification of security-related attributes.

Most of the existing common solutions for security enforcement, in both classical and cloud environments, rely upon the adoption of a certification approach (NIST [38], CIRRUS project [39],...
CCM [40]), which assumes a complete knowledge of all the service layers, and typically involves static security configurations, verified through security audits. The adoption of Security SLAs, instead, requires automating the process of setting up and configuring security features for a target service on the basis of customer requirements. This assumes, on the one hand, a well-defined representation of security requirements that is understandable for both customers and providers and, on the other hand, the capability of effectively monitoring that such requirements are met (by means of verifiable security-related Service Level Objectives). These issues have been tackled by the SPECS project, whose goal is the development of a framework to build applications that deliver cloud services controlled by Security SLAs. One of the main results of SPECS is the XML SLA Framework (available online at [41]), which provides a model and a machine-readable format for Security SLAs.

As discussed in Section 3.1.1, the SPECS Security SLA model has been adopted and extended by MUSA to represent multi-cloud applications’ Security SLAs.
3 MUSA initial methods and tools

This section presents the initial SbD methods, models and tools introduced in MUSA in order to enable the analysis of the security needs of a multi-cloud application, and the design of its components with respect to not only functional requirements but also security requirements. The conceptual models and the processes at the basis of the SbD development approach are presented in Section 3.1. Section 3.2 focuses on the SLA Generation activity and illustrates the main involved steps, Section 3.3 introduces the main tools introduced to support the development process, namely the Security Metric Catalogue and the risk analysis process. Finally, Section 3.4 presents a proof-of-concept application that implements the SLA Generation process, and Section 3.5 an example of SLA generated by such application for one of the components of the TUT case study application.

3.1 Multi-cloud application design methods

SbD requires security to be taken into account from the very early stages of the design process. Kreizman and Robertson, with their Gartner whitepaper [53], were probably the first to position SbD principles in the enterprise context. Indeed, SbD is a complex practice, not always addressed in common software even in enterprise solutions. Moving this challenge to the cloud (and multi-cloud) context just makes it even more complex, due to the lack of control over resources that developers and service providers experience.

The SbD approach is addressed in MUSA by means of the introduction of a design methodology based on the intense adoption of Security SLAs. According to such a methodology, the building blocks of a multi-cloud application (i.e., the application components and the invoked third-party services) are modelled in terms of their security requirements, expressed through (requested) Security SLAs. In particular, as we clarify later, the MUSA development framework generates a set of (requested) Security SLAs associated to each application component as the result of a risk analysis process. Such Security SLAs are then stored in a repository, so that the MUSA Security Assurance Platform can verify that the SLOs are respected during the multi-cloud application lifetime. In fact, during the design, deployment and execution of the multi-cloud application, we identify, negotiate and monitor the requested Security SLAs, trying to prevent possible security violations and/or to react to them when they cannot be avoided.

The following box summarizes the main concepts behind the proposed approach, which will be clarified in the following sections.

| Goal: | Develop an SbD process for secure multi-cloud applications. |
| Assumption: | MUSA multi-cloud applications are made of components that cooperate by invoking respective services and that may use services offered by external CSPs. |
| Approach: | Adopt Security SLA as a means to design security from the very early design stages. Each component has an associated Security SLA. The multi-cloud application Security SLA is a composition of the Security SLAs of the application’s components. |
3.1.1 Conceptual models
In the following, we illustrate the adopted Security SLA model and show how Security SLAs are linked to the concept of security threats. As anticipated in fact, these are the concepts at the basis of the MUSA SbD process of multi-cloud applications.

3.1.1.1 The SPECS Security SLA model
The MUSA SbD methodology relies on the adoption of Security SLAs as a means to represent and model the security associated with each component of the multi-cloud application. MUSA adopts and extends the Security SLA model developed within the SPECS project [31], which enables to express security-related terms and guarantees related to a specific cloud service. In MUSA, the concepts included in such model are used to express security-related features and guarantees related to both a single multi-cloud application component and to the whole application. However, in what follows, we refer to the Security SLA built for a single component, while the composite SLA will be discussed in the final version of this deliverable.

![Figure 10: The SPECS Security SLA model](image)

Figure 10 shows the SPECS Security SLA model. As depicted, the model represents the security by means of a declarative part, where security-related characteristics of the service being provided are described in a way that cannot be verified, but can be easily associated to common security concepts, and a measurable part, which specifies the guarantees by using security metrics that can be concretely measured.

The declarative part includes the description of the cloud resources consumed by a service with their respective providers. Note that in SPECS only IaaS providers were considered, therefore the SLA included the information on the provider (Resources Provider section) and on the types of virtual machine instances (VM section) available to host the service. As previously pointed out, in MUSA a multi-cloud application component may be both of IaaS, SaaS and PaaS type, and therefore this part of the SLA model needs to be changed. In particular, as will be clarified in the following two subsections, only the type of component (i.e., its functional behaviour) must be specified.

The declarative part of the SLA models security features requested/offered on each service by means of the identification of the involved security capabilities, defined in terms of the enforced security controls [38]. Security controls are defined by NIST as the safeguards or countermeasures prescribed for an information system or an organization, and designed to protect the confidentiality, integrity, and availability of its information and to meet a set of defined security requirements. In practice, standardization bodies like NIST or ISO, and consortiums like Cloud Security Alliance (CSA), propose catalogues of security controls, named Control Frameworks, which identify the countermeasures recommended to address specific security issues. Examples of such frameworks are
D2.1: Initial SbD methods for multi-cloud applications

the NIST Security Control Framework [38] and the CSA’s Cloud Control Matrix [40]. Security Controls are typically used to perform the security assessment of systems through manual auditing of the services, which aims at verifying their correct implementation.

Security SLAs need to specify security in a measurable way, therefore the model includes a measurable part that comprises the declaration of the security metrics that can be monitored by the service customer to verify the correct delivery of declared capabilities. In SPECS, a Security Metric Catalogue was proposed. As will be clarified later in Section 3.3.1, this catalogue has been adopted by MUSA with the aim of validating and extending the set of included metrics based on the feedback from case study applications.

The offered security guarantees are defined in terms of security SLOs. Security SLOs represent the security levels that the service customer requires and that the service provider accepts to offer. Therefore, SLOs are constraints on the admissible values of declared security metrics.

SLOs and the relative security metrics are associated with the declared security capabilities and are meant to offer a quantitative measure of the declared security controls.

3.1.1.2 Security SLAs and Threat Modelling

Figure 11 represents a simplified view of the MUSA domain model illustrated in Deliverable D1.1, which focuses on the relationships among the concepts of SLA, security controls, vulnerabilities, and threats and introduces the concept of component type.

As already said, multi-cloud application components use cloud services, which may belong to several cloud service types (i.e., IaaS, SaaS, PaaS) and which constitute the tangible assets of the application under design. Moreover, it must be noticed that a component is also characterized by a component type, which mainly refers to its functional behaviour (i.e., the component acts as a database, as a web application, etc.).

Depending on the cloud service type it belongs to and on its component type, a component may be subject to specific vulnerabilities. If these vulnerabilities are actually exposed, specific threats may exploit them, therefore posing a risk. Let us consider the case of a web application. Web applications are typically subject to code injection, but this vulnerability can be exploited only if the application evaluates the input from customers, while if user inputs are never evaluated the code cannot be injected. This means that the exposure of a component to vulnerabilities depends on its behaviour and on how it is implemented.

The threats pose risks that can be mitigated through the enforcement of proper treatments, which represent specific security controls to apply. Security controls are included in a Security SLA together with associated security metrics, which can be used to measure the level at which they are actually enforced. Security controls belong to a selected Security Control Framework, while security metrics are defined within a Security Metric Catalogue.
3.1.1.3 The MUSA Security SLA model and the adopted SbD approach

The above discussed model suggests that, in order to produce the components’ Security SLAs, it is possible to carry out a risk analysis process for each component (based on its type and behaviour) and to identify the security controls that can be applied to mitigate the existing risks. This process, which is the basis of the SbD approach followed in MUSA, will be discussed in the next section. Meantime, based on this achievement, it is possible to identify the MUSA Security SLA model, based on the SPECS SLA model and including component type and threat-related information. Such a model, which will support the whole SbD development process, is depicted in Figure 12.

![Figure 11: Security Controls and Threat Analysis](image1)

![Figure 12: The MUSA Security SLA model](image2)

3.1.2 Design process

Following the approach introduced above, possible security threats must be identified as soon as possible during the development process of an application, in order to avoid unnecessary risks. In the cloud and multi-cloud contexts, this need grows due to an increased attack surface; this is worsened by the fact that multi-cloud applications typically use different cloud services offered by different providers, to which the quality of acquired resources is delegated.

As anticipated, in order to develop a multi-cloud application, we need to define a methodology able to address security at the early stages of the development phase and to adapt to the application
D2.1: Initial SbD methods for multi-cloud applications

24 deployment. As said, this methodology relies upon Security SLAs, which are a useful tool to model the security granted by cloud services. Therefore, a multi-cloud application can be modelled as:

- a set of artefacts to deploy, one for each component of the application;
- a set of (requested) Security SLAs, one for each component of the application (external SaaS services used in the application must also be considered);
- the (offered) Security SLA of the multi-cloud application, representing the security level that the application is able to guarantee.

The Security SLA of the multi-cloud application can be obtained as the result of an iterative development process, shown in the activity diagram of Figure 13.

![Figure 13: Security SLA-based SbD development process activities](image)

The Security SLA-based SbD multi-cloud application development process consists of five main activities:

- In the **Design Modelling** activity, the multi-cloud application is modelled in terms of its Cloud Provider Independent Model (CPIM), which details the information about the cloud services used by the multi-cloud application components. The result of this phase is a model of the application in terms of its functional components, according to the model described in Section 2.2.

- In the **SLA Generation** activity, the security requirements at component level are identified and formalized in terms of Security SLAs. Each component is analysed based on a standard threat modelling procedure, and a simplified risk analysis methodology is carried out with focus on the tangible technical assets (the components), in order to identify the main risks associated with the security issues that may arise. Threats and risks are then associated with treatments and, in particular, to the set of security controls to enforce, as shown in Section 3.1.1.2, and a Security SLA is built for each component.

- In the **SLA Feasibility Verification** activity, an analysis on the feasibility of resulting Security SLAs is conducted, in order to check whether any existing provider can fulfil desired security requirements. The SLA feasibility phase takes as input the set of Security SLAs and produces, as an output, the list of possible available offers (i.e., the list of available services provided by CSPs) along with the list of possible missing security controls. The developer has to analyse the result of the SLA Feasibility process: if the Security SLAs cannot be satisfied with existing services, a set of security libraries must be added to fulfil the uncovered security requirements, thus requiring an update in the application model. If instead the Security SLAs are feasible, the SLA Composition activity takes place.

- In the **SLA Composition** activity, a Security SLA for the multi-cloud application is derived by composing the Security SLAs of single components. Note that, even if each component satisfies its own security requirements, this does not grant that the overall multi-cloud application is able to guarantee the desired level of security. As an example, consider the case of a simple application made up of two components, namely a web server and a database.
When executing the SLA Generation activity for both components separately, we may not take into account for example that, when using them together, also the communication channel should be protected (as a general requirement of the application). In general, if the composed Security SLA fulfills the initial multi-cloud application requirements, the implementation can take place, otherwise a new design modelling activity is required. It is worth noticing that, in a second iteration, the analysis made in the first round would be available in the form of an already elaborated security SLA, which now should only be updated according to higher-level requirements.

- The Add Security libraries activity is launched if any of the components’ Security SLAs results unfeasible, i.e., if there is no provider that already offers the required components with the needed security features. The missing security features are added, if possible, by means of the activation of proper security mechanisms available commonly in form of libraries, whose integration may imply the deployment and configuration of additional components. Let us consider for example the case of a multi-cloud application component represented by a storage service, and let us assume that, as a result of the risk analysis process, the end-to-end encryption of user data is recommended. If none of the cloud storage providers known by the MUSA IDE is able to provide such a feature, this may be added in form of a proper library, embedded in the client, which is responsible for the encryption of data before its submission to the remote storage service. In this case, the design activity must be repeated, in order to take into account the possible new security components to add to the multi-cloud application.

3.1.2.1 The design process in the MUSA Framework

The above Security SLA-based SbD multi-cloud application development process is implemented in the MUSA framework through the adoption of the MUSA IDE Modeller, the MUSA IDE SLA Generator and the Decision Support Tool (DST). The main steps carried out by the DevOps team using these tools during the multi-cloud application design are depicted in Figure 14 (refer to deliverables D1.4 and D1.2 for the detailed discussion).

As shown in the sequence diagram, in the first steps of the sequence (1-1.4), the DevOps team produces the preliminary design of the multi-cloud application (the one that takes into account only the functional aspects) with the help of the Modeller and the DST – Discovery tools (by using the associated data repository). Then (steps 2-2.1.1.2), the DevOps team resorts to the SLA Generator to produce the set of Security SLAs for each component, after making a preliminary risk analysis. These Security SLAs, associated with the components and/or the SaaS services used by the multi-cloud application, are submitted to the DST – Risk Analysis and Decision tools (step 3-3.1.1.4) in order to make a full risk analysis and treatment matching against possible cloud offerings. According to the result of such analysis, the DevOps team identifies the MUSA libraries needed to respect the agreed SLAs (when cloud offerings are lacking them or service them at a higher cost than MUSA) and produces the set of Security SLAs and the enriched design team (steps 4-7).

Note that all the steps described above happen in loops repeated when the results are not satisfying for the DevOps team. When the full process is complete, the multi-cloud application composite Security SLA is produced by the SLA Generator (step 8).

---

1 As pointed out in D1.1, the DevOps Team is the main MUSA stakeholder, responsible of the multi-cloud application development, deployment and execution management according to the DevOps approach.
In the present version of the SbD methods of MUSA, we have focused on the Security SLA Generation activity, as the SLA will be the glue for the security assurance at runtime. This activity is detailed in next sections.

The details of the proposed Design Modelling approach, including the CPIM modelling language and supporting tools, will be provided in future deliverable D2.2 Initial MUSA IDE for security-aware design of multi-cloud applications. This deliverable will also include the initial mechanisms for the Add security libraries activity, which will be supported by modelling tools.

The security SLA Composition mechanisms and tools will be described in future D2.3 Final SbD methods for multi-cloud applications, while the support to the SLA Feasibility Verification activity will be included in future D3.1 Initial security based discovery and composition mechanisms and tools.
3.2 The SLA Generation process

Figure 15: Details of the SLA Generation process

Figure 15 summarizes the main steps of the design process introduced in Section 3.1.2, and focuses in particular on the steps carried out during the SLA Generation activity.

As said, the SLA Generation process takes as input the model of the application resulting from the design activity. The design phase consists in identifying the technical tangible assets of the application and in modelling its architecture. The preliminary model resulting from this phase mainly includes the set of components adopted by the application, along with their functional description. It is worth mentioning that such model is completely independent from any cloud provider and does not rely on any specific cloud service (i.e., it is a CPIM).

At state of the art, the CPIM model we adopt is a simple list of components characterized by their type. As an example, a simple description of the TUT case study can be represented as:

```xml
<Components>
  <Component name="TSMEngine" type="web App"/>
  <Component name="DataBase" type="storage"/>
  <Component name="Journey Planner" type="web App"/>
  <Component name="Google Direction" type="external saas"/>
  <Component name="ITSfactory" type="external saas"/>
  <Component name="FMI" type="external saas"/>
</Components>
```

Note that we report both the components deployed over IaaS cloud services, which are completely under control of the DevOps team, and the external components that offer SaaS cloud services used by the multi-cloud application.

The CPIM of the multi-cloud application is used to feed the SLA Generation process, which consists of three main steps:

1. **Design analysis**: in this phase, the description of each component of the application is enriched with a set of properties, which depend on the implementation of the component and are related to the possible risks deriving from implementation and usage choices.
2. **Risk Analysis**: in this phase, the Risk Analysis Module of the DST, based on the component type and on the properties defined at the previous step, will identify the main threats, risks and treatments each component is subject to. Risks are classified and ranked (e.g., based on parameters such as likelihood, impact or consequence, etc.) according to existing risk assessment methodologies.

3. **SLA generation**: in this phase, for each threat encountered at the previous step, one or more security controls are identified. These controls, along with the set of security metrics of interest and related SLOs, are used to build up a Security SLA for each component of the application. Such SLAs represented the *requested* SLAs, namely the security requirements of each component.

The Security SLAs resulting from the SLA Generation process are the inputs of the **SLA Feasibility** phase, managed by the DST, which has to determine whether and how the included requirements can be fulfilled by existing known providers/services. In case some controls are not fulfilled, as anticipated, the **Add Security Libraries** activity will be executed.

### 3.3 Supporting tools and modules for SLA Generation

In this section, we give some insight on two of the main tools/modules adopted during the SLA Generation phase. A fundamental tool for the construction of Security SLAs is the Security Metric Catalogue, used to select the metrics associated with the security controls resulting from the risk analysis process. The second fundamental element to discuss is the risk analysis process itself, carried out by the DST- Risk Analysis Module.

#### 3.3.1 MUSA Security Metric Catalogue

As mentioned before, MUSA adopts and extends the Security Metric Catalogue developed in the SPECS project, which promoted, in a joint effort with relevant standardization groups, the definition of a shared model for security metrics representation and an XML-based machine-readable format [48] for their automated processing. Based on such model, SPECS released a collection of security metrics relevant to the project and an application for security metric management, available as [http://apps.specs-project.eu:8080/metric-catalogue-app/](http://apps.specs-project.eu:8080/metric-catalogue-app/). In MUSA, in addition to SPECS metrics, several other sources were analysed, including the Special Publication “Performance Measurement Guide for Information Security” by NIST [49], the “Consensus Security Metrics v1.1” by the Center for Internet Security (CIS) [50], and the results of other EU projects such as Cumulus [51] and A4Cloud [52].

The TUT and Lufthansa Systems partners in MUSA project, responsible for the case studies, were asked to validate the metrics and select those of interest, and to add new metrics that were considered relevant to the case study applications. The set of information collected for each metric is reported in Table 2, where the required fields are identified with a “**” symbol. The collection and validation of metrics is still an ongoing task, which will be finalized only with the final version of this deliverable. However, some examples of metrics that were considered in MUSA are reported in Table 3.

#### Table 2. Security metric specification format

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Metric Name</strong></td>
<td>A generic Identifier for the metric</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>A natural Language description of the Metric</td>
</tr>
<tr>
<td><strong>Value</strong></td>
<td>Range of possible values</td>
</tr>
<tr>
<td><strong>Unit</strong></td>
<td>Unit of measurement</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>(to be used in the SLA) the value to be proposed as default value</td>
</tr>
<tr>
<td><strong>Operator</strong></td>
<td>The allowed operators to define SLOs (e.g., =, &gt; etc.)</td>
</tr>
<tr>
<td><strong>Source</strong></td>
<td>Where the metric has been defined (e.g., SPECS, MUSA, CIS, NIST SP800-55)</td>
</tr>
<tr>
<td><strong>CCM v3.01</strong></td>
<td>Associated CCM Controls</td>
</tr>
</tbody>
</table>
NIST 800-53 v4 | Associated NIST controls

Table 3. Examples of metrics

<table>
<thead>
<tr>
<th>Metric Name</th>
<th>Description</th>
<th>Value</th>
<th>Unit</th>
<th>Default</th>
<th>Operator</th>
<th>CCM</th>
<th>NIST</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vulnerability report frequency</td>
<td>It represents the frequency of generation of the report of existing vulnerabilities</td>
<td>0 ≤ integer</td>
<td>hours</td>
<td>24</td>
<td>eq (=)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Service availability</td>
<td>Percentage of time in which service access is available to users</td>
<td>0 ≤ integer ≤ 100</td>
<td>%</td>
<td>n/a</td>
<td>ge (≥)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HTTP to HTTPS Redirects</td>
<td>It is the most common requirement on most servers which ensures that the connections are coming from customers using SSL.</td>
<td>yes / no</td>
<td>n/a</td>
<td>yes</td>
<td>eq (=)</td>
<td>EKM-03</td>
<td>SC-8</td>
</tr>
</tbody>
</table>

3.3.2 Discussion on Risk Analysis

Risk analysis is an essential part both for SLA Generation and to allow for decision support in the DST. In MODAClouds FP7 project, a Decision Support System was developed for cloud service selection that included a first version of a risk analysis methodology and tool to support decision making. In MUSA, we are planning to innovate on the mechanisms used for risk analysis in agile software development contexts where organizations aim at nurturing a DevOps culture.

Nowadays, risk is usually managed in a superficial way in general. Risk management is currently done at design time and focused on tracking impediments, while our customers are already talking about emerging architectures, continuous design, delivery and deployment, etc. We need new mechanisms that enable risk analysis in this context, as many companies are already interested in risk management for software development. While different attempts have been made to create agile risk management methodologies, protecting assets of a complex application architecture in the context of agile software development is a problem that has not been addressed yet.

In deliverable D1.2, we establish the base on top of which the risk analysis tool will be built. In particular, we explain how assets and, in particular, multi-cloud application components are identified and mapped, and we also describe how we plan to give priority to the different components by expressing the acceptability on the level of risk. We also include an explanation on the different types of assets and how they are defined. Besides, we also describe the process through which assets are taken for risk analysis, including the definition of risks and the definition of mitigation actions or treatments.

In deliverable D3.1, further details will be provided including how the DevOps team identifies the assets, their risks, the prioritization, the mapping to treatments, etc. We will also define a new tool to perform the risk analysis of software components in a way that it can be easily embedded in an agile process and allows for the collaboration of stakeholders from both “Dev” (architect, developer, product owner) and “Ops” (customer support engineer, system operator, system deployer, service administrator) sides.

As outlined in the process above, the SLA Generation process assumes as an input a preliminary model of the application (CPIM), which describes the technical assets of the multi-cloud application and, according to such model, produces an initial set of threats that should be addressed through adequate countermeasures.
For each technical asset (mainly components and their communication links), a risk analysis process will be conducted, which takes as input the asset description, enriched with a preliminary selection of the threats. The output of this activity will be an enriched description of the technical asset, made of:

- The list of threats: the same list that was submitted to the risk analysis tool, plus possible additional threats;
- The risk rating results: an evaluation of the level of risk for the component, possibly organized per threat category;
- A list of suggested countermeasures, expressed as the description of the countermeasure, a possible reference to standard security controls, and the list of threats that are addressed by such countermeasures.

### 3.4 Security SLA Generation: proof-of-concept

In this section, a proof-of-concept application implementing the described SLA Generation process is presented. The application implements the steps discussed in Section 3.2 as described in the following sections.

#### 3.4.1 Design analysis

The input of the *design analysis* phase is a provider-independent model of the application, which specifies the main components of the application mainly in terms of their nature and behaviour. Our proof-of-concept application supports the developer in this preliminary phase (referred to as the *design* phase), by enabling him/her to specify the application components and to assign them a type. Three types of components are currently supported:

- **custom web app**: the component is a custom web application, accessible via a network (either private or public). The web application offers specific functionalities through an application server, which can be consumed by generic clients represented by web browser;
- **external software-as-a-service**: the component is an external service that is not under the control of the developer but is provided by a third party;
- **storage-as-a-service**: the component is a storage service offered by an external provider.

During the design phase, the developer is asked to specify the component name, description and type among those discussed above. Once this information is provided, the actual *design analysis* phase can take place, where the application model is enriched with the assignment of specific properties to each application component, useful to assess the associated risk and determine the related security requirements. This first proof-of-concept does not include the integration with the DST. Therefore, in order to enable the subsequent threat analysis, our proof-of-concept application has been preloaded with a set of associations `<threat - component type>`, which basically define the threats a component is subject to based on its type. These associations have been built by following several existing guidelines for the development of secure applications. In particular, the OWASP Top 10 [54] was considered to identify the most critical web application security flaws, and several security bulletins and other relevant sources carefully analysed to find the main threats against other component types.

The result of this phase is a list of possible threats for each component of the application. Clearly, whether the component is actually subject to a certain threat may strongly depend on how the component has been developed or on how it is meant to be used. For this reason, we prepared a questionnaire aimed at guiding the developer in identifying, for each threat, the real issues related to the implementation and usage of the component, so to discard threats that are not actually solicited.

The questions were simply built by analysing the description of each threat, and by extracting the relevant information.

The set of relevant threats resulting from the filtering process is the input for the *risk analysis* phase.
The complete list of threats managed by our proof-of-concept application, along with the associations threat-component type, is reported in Appendix A. For each threat, we also provided a mapping with the STRIDE [56] categorization methodology [56], an approach developed by Microsoft for identifying and classifying computer security threats. STRIDE considers six threat categories: spoofing, tampering, repudiation, information disclosure, denial of service and elevation of privileges. These categories will be used later, in the risk analysis phase, to group risk values related to each threat.

### 3.4.2 Risk Analysis

In the risk analysis phase of this proof-of-concept application, the developer is guided through a risk classification process based on the OWASP Risk Rating Methodology [55]. For each threat, the developer has to assign a score (in the range [0-9]) to each of the available indicators belonging to the probability of occurrence (likelihood) and impact macro-categories, which are combined to obtain a final risk value (risk is calculated or defined as the product of likelihood and impact: Risk = Likelihood x Impact).

The risk values associated with each threat are then grouped according to the categories identified by the STRIDE methodology, producing 6 different risk values as result, one for each category. This grouping is done to give the user a comprehensive view, so to be able to identify which of the six aspects is more risky (this usually depends on the nature of the component and of the offered functionalities).

Please, note that this risk analysis methodology may be modified when the interaction is directly done with the final risk analyser implemented in the DST. In any case, the outputs of the final risk analysis will be compatible with this temporary mechanism. Therefore, this proof-of-concept is valid to show how the final SLA generator will work in collaboration with the DST.

### 3.4.3 SLA Generation

The first step of the SLA Generation phase consists in associating a set of security controls to each threats identified at the previous step. In our proof-of-concept application, the NIST security Control Framework has been considered as the source of security controls. In practice, the application suggests the enforcement of a set of security controls in order to cope with the existing threats, based on the associated risk. The association is done statically, based on a mapping built ad-hoc.

Moreover, the application also presents the developer with the controls associated with each of the six STRIDE categories, in order to give him/her a wider choice. The developer can select the security controls of interest, which will be included in the SLA of the component being analysed.

After the selection of the security controls, the relevant security metrics must be chosen and related SLOs must be defined. Security metrics are statically mapped to security controls, therefore the developer is prompted with applicable metrics and is asked to select those of interest and to define related objectives.

Finally, all selected controls and metrics with related SLOs are included in an SLA. For each component, an SLA is prepared according to the model presented in Section 3.2.1, and passed to the DST for the SLA Feasibility phase.

### 3.5 Examples on MUSA case studies

In this section, we report an example of execution of the Security SLA Generation proof-of-concept application for one of the components belonging to the MUSA case studies. In particular, we show the whole process referred to the TSM Engine component of the TUT Smart Mobility multi-cloud application. The complete SLA generated for the component is reported in Appendix B.
As discussed in Section 3.4, in the Design Analysis phase, the user has to specify the types of the components building the application. For the TSM Engine component, the custom web app type has been specified. As partly shown in Figure 16, the threats associated with the custom web app component type are prompted to the user, who can select those of interest.

---

**Figure 16: Selecting the threats associated with the TSM Engine component**

The complete list of threats that were selected is the following:

- Injection
- Broken Authentication and Session Management
- Insecure Direct Object References
- Cross-Site Request Forgery (CSRF)
- Cross-Site Scripting (XSS)
- Denial of Service
- Weak Identity, Credential & Access Management
- Unauthorized access to admin interface
- Over-privileged application and accounts

In the subsequent phase, partly shown in Figure 17, the developer is guided through a risk analysis process, where he/she is asked to assign a score to each threat, categorized based on STRIDE.
Based on this analysis, the application returns the list of suggested security controls to apply. The developer selects those of interest, which are included in the SLA. The list of baseline controls that were selected is the following (we do not report here all the control enhancements for brevity’s sake):

- RA-5: VULNERABILITY SCANNING
- IA-5: AUTHENTICATOR MANAGEMENT
- SC-23: SESSION AUTHENTICITY
- SI-4: INFORMATION SYSTEM MONITORING
- SC-5: DENIAL OF SERVICE PROTECTION
- CA-7: CONTINUOUS MONITORING
- SC-13: CRYPTOGRAPHIC PROTECTION

After, the list of metrics associated with the selected controls is presented, and the developer can define SLOs on top of them (the selection of metrics is partly shown in Figure 18).

As said, the complete SLA generated for the component is reported in Appendix B. Note that the example SLA includes a set of metrics that were directly borrowed by a pre-existing mapping among security controls and metrics identified in SPECS and present in some standards. This set of metrics was included for the sake of completeness with respect to the generation of SLAs, but it will be refined when the process of MUSA metrics’ validation and analysis is completed.
4 Requirement coverage discussion

This section reports on the coverage of requirements elicited in the context of WP1 and referring to the design process described in Section 3.1.2. In particular, the considered requirements refer to the following three aspects: (i) the generation of per-component Security SLAs, (ii) the composition of these SLAs to obtain the multi-cloud application Security SLA, and (iii) the definition of proper security libraries to provide required security features according to a given SLA.

As said before, this deliverable is focused on the SLA Generation process, while the aspects of SLA composition and of security libraries definition will be addressed in other deliverables. For this reason, in the following, we identify the requirements that are relevant for the SLA Generation process and discuss whether and how they have been covered by the current prototype.

The coverage of remaining requirements will be reported in the future in designated deliverables.

4.1 MUSA SLA Generation

Table 4 summarizes the MUSA requirements that are relevant to the SLA Generation phase.

<table>
<thead>
<tr>
<th>ReqID</th>
<th>Title</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1.4-R2</td>
<td>Multi-cloud app properties defining tool</td>
<td>The MUSA Framework should provide a tool that support application architects in the definition of functional and security properties of multi-cloud applications.</td>
</tr>
<tr>
<td>S1.4-R3</td>
<td>Tool for defining the requirements of the cloud services</td>
<td>The MUSA Framework should provide a tool that enables application architects to model functional and security requirements of the cloud services where the multi-cloud application will be deployed.</td>
</tr>
<tr>
<td>S1.4-R4</td>
<td>Support to SLA generation</td>
<td>The MUSA Framework should provide a tool that enables application architects to obtain (as automatically as possible) the SLA from the application model.</td>
</tr>
<tr>
<td>S2.1-R2</td>
<td>Support defining multi-facet requirements</td>
<td>The MUSA Framework should provide a tool that supports application architects in the definition of application’s security, functional and business requirements. The tool should enable the architect to specify complex requirements related to the usage of the cloud resources belonging to different providers.</td>
</tr>
<tr>
<td>S2.3.1-2</td>
<td>Coherent Interfaces for user interaction</td>
<td>The MUSA Framework should provide a mechanism that allows the user defining security and functional requirements of cloud resources to use.</td>
</tr>
<tr>
<td>S3.1-R2</td>
<td>Definition of Security and Functional reqs</td>
<td>The MUSA Framework should provide a mechanism that allows the user defining security and functional requirements of cloud resources.</td>
</tr>
<tr>
<td>S6.1-R1</td>
<td>Language for the specification of security properties in app contract</td>
<td>The MUSA framework provides a language which will be used to define the multi-cloud application security requirements. (SLA language).</td>
</tr>
<tr>
<td>S6.1-R3</td>
<td>Language for the specification of security properties in app model</td>
<td>The MUSA framework provides a modelling language which will be used to define the multi-cloud application security requirements. (App architecture modelling language).</td>
</tr>
</tbody>
</table>

Since MUSA requirements have been elicited by analysing a set of relevant scenarios, some of them are overlapping and express the same requirement in practice. Table 5 groups such requirements without repetitions and summarizes the main required functionalities. Moreover, it reports a brief discussion of how each requirement has been covered.
Table 5. Summary of MUSA SLA Generator requirements and coverage

<table>
<thead>
<tr>
<th>ReqID</th>
<th>Title</th>
<th>Description</th>
<th>Coverage discussion</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLAGEN_R1</td>
<td>Support for the definition of multi-cloud application’s properties</td>
<td>The MUSA Framework should provide a tool that supports application architects in the definition of functional, business and security requirements of multi-cloud applications.</td>
<td>Our proof-of-concept SLA Generator enables the user (application architect) to identify the main functional components of the multi-cloud application and to specify the security controls that the application must implement.</td>
</tr>
<tr>
<td>S1.4-R2</td>
<td>S2.1-R2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SLAGEN_R2</td>
<td>Support for the definition of multi-cloud application’s components’ properties</td>
<td>The MUSA Framework should provide a tool that enables application architects to model functional and security requirements of the multi-cloud application’s components (i.e., the cloud services and resources used by the multi-cloud application).</td>
<td>Our proof-of-concept SLA Generator enables the user (application architect) to identify the main functional components of the multi-cloud application and to specify the security controls that each component must implement. This is accomplished by carrying out a threat analysis on each component.</td>
</tr>
<tr>
<td>S1.4-R3</td>
<td>S2.3.1-2</td>
<td>S3.1-R2</td>
<td></td>
</tr>
<tr>
<td>SLAGEN_R3</td>
<td>Support to SLA generation</td>
<td>The MUSA Framework should provide a tool that enables application architects to obtain (as automatically as possible) the SLA from the application model.</td>
<td>Our proof-of-concept SLA Generator automatically generates an SLA for each component of the application. The SLA is compliant with the model described in Section 3.1.1.1.</td>
</tr>
<tr>
<td>S1.4-R4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SLAGEN_R4</td>
<td>Definition of a language for Security SLAs</td>
<td>The MUSA framework provides a language which will be used to define the multi-cloud application security requirements.</td>
<td>The language used for the specification of security requirements is represented by the Security SLA machine-readable format, discussed in Section 3.1.1.1.</td>
</tr>
<tr>
<td>S6.1-R1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SLAGEN_R5</td>
<td>Definition of a language for the specification of security properties in app model</td>
<td>The MUSA framework provides a modelling language which will be used to define the multi-cloud application security requirements. (App architecture modelling language).</td>
<td>UNCOVERED: The SLA Generator is meant to adopt a language based on CloudML, enriched with security-related concepts, to specify the requirements of a multi-cloud application. This language has not been integrated yet in our proof-of-concept.</td>
</tr>
<tr>
<td>S6.1-R3</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
5 Conclusion and further work

The document provided a description of the preliminary mechanisms and tools offered by MUSA to define the security constraints in multi-cloud application components. In particular, the document focused on the SLA Generation activity and provided:

- The detailed description of the steps accomplished in the activity;
- The definition of the conceptual models and tools on which the design process and this activity in particular found:
  - MUSA multi-cloud application model,
  - Threats-Security SLAs relationship model,
  - MUSA SLA Security model,
  - MUSA Security Metric Catalogue.
  - MUSA Threat Catalogue (used by the proof-of-concept application).
- The description of a proof-of-concept application that implements the whole SLA Generation activity;
- An example of execution of the proof-of-concept application for one if the components of the TUT case study application;
- A discussion of the current coverage of requirements related to this activity with the proof-of-concept application.

The other phases of the SbD development process will be addressed in other deliverables:

- The details of the Design Modelling approach, including the CPIM modelling language and supporting tools, will be provided in future deliverable D2.2 Initial MUSA IDE for security-aware design of multi-cloud applications.
- Initial mechanisms for the Add security libraries activity will be included in deliverable D2.2.
- The security SLA Composition mechanisms and tools will be described in future D2.3 Final SbD methods for multi-cloud applications
- The SLA Feasibility Verification phase will be included in future D3.1 Initial security based discovery and composition mechanisms and tools.
References


D2.1: Initial SbD methods for multi-cloud applications


### Appendix A. MUSA Proof-of-concept SLA Generator Threat Catalogue

#### Table 6. Threat catalogue used by the proof-of-concept SLA generator

<table>
<thead>
<tr>
<th>Threat</th>
<th>Description</th>
<th>STRIDE category</th>
<th>Involved component types</th>
<th>Condition</th>
</tr>
</thead>
</table>
| **Account Hijacking**               | Account hijacking is a process through which an individual’s email account, computer account or any other account associated with a computing device or service is stolen or hijacked by a hacker. It is a type of identity theft in which the hacker uses the stolen account information to carry out malicious or unauthorized activity. | SPOOFING        | • custom web app  
• external software-as-a-service                          | The component manages user accounts                                  |
| **Advanced Persistent Threats (APTs)** | An advanced persistent threat (APT) is a network attack in which an unauthorized person gains access to a network and stays there undetected for a long period of time. The intention of an APT attack is to steal data rather than to cause damage to the network or organization. APT attacks target organizations in sectors with high-value information, such as national defense, manufacturing and the financial industry. | REPUDIATION     | • custom web app                                                | The component has a state and computation resources                      |
| **Broken Authentication and Session Management** | Application functions related to authentication and session management are often not implemented correctly, allowing attackers to compromise passwords, keys, or session tokens, or to exploit other implementation flaws to assume other users’ privileges. | SPOOFING        | • custom web app,  
• storage-as-service                                        | The component offers services that use sessions                         |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th>TAMPERING</th>
<th>INFORMATION DISCLOSURE</th>
<th>DENIAL OF SERVICE</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Cross-Site Request Forgery (CSRF)</strong></td>
<td>A CSRF attack forces a logged-on victim’s browser to send a forged HTTP request, including the victim’s session cookie and any other automatically included authentication information, to a vulnerable web application. This allows the attacker to force the victim’s browser to generate requests the vulnerable application thinks are legitimate requests from the victim.</td>
<td>• custom web app,</td>
<td>• custom web app,</td>
<td>• custom web app,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• external software-as-service,</td>
<td>• external software-as-service,</td>
<td>• external software-as-service,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• storage-as-service</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>The component exposes an interface through which it is possible to send malicious requests</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Cross-Site Scripting (XSS)</strong></td>
<td>XSS flaws occur whenever an application takes untrusted data and sends it to a web browser without proper validation or escaping. XSS allows attackers to execute scripts in the victim’s browser which can hijack user sessions, deface web sites, or redirect the user to malicious sites</td>
<td>• custom web app,</td>
<td>• custom web app,</td>
<td>• custom web app,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• external software-as-service,</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>The component exposes an interface that does not validate inputs and allows to execute scripts</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Data Breaches</strong></td>
<td>Data breaches occur whenever protected data is read, stolen or used by a not-authorized individual. These violations may involve personal health information, personal data, industrial secrets or intellectual property.</td>
<td>• custom web app,</td>
<td>• custom web app,</td>
<td>• custom web app,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• external software-as-service,</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>The component hosts or manages sensible data</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Denial of Service</strong></td>
<td>DoS e DDoS (Distributed Denial of Service) are aimed at disrupting a service and undermining its availability.</td>
<td>• custom web app,</td>
<td>• custom web app,</td>
<td>• custom web app,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• external software-as-service,</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>The component exposes services</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Injection</strong></td>
<td>Injection flaws, such as SQL, OS, and LDAP injection occur when untrusted data is sent to</td>
<td>• custom web app,</td>
<td>• custom web app,</td>
<td>• custom web app,</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Threat Type</td>
<td>Description</td>
<td>Example Scenarios</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
<td>------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Insecure Direct Object References</td>
<td>A direct object reference occurs when a developer exposes a reference to an internal implementation object, such as a file, directory, or database key. Without an access control check or other protection, attackers can manipulate these references to access unauthorized data.</td>
<td>INFORMATION DISCLOSURE</td>
<td>custom web app, storage-as-service, evaluation</td>
<td></td>
</tr>
<tr>
<td>Man in the Middle attack</td>
<td>In the MITM attack, an attacker interferes with the communications between two parties by posing himself in the middle and intercepting/altering messages.</td>
<td>SPOOFING</td>
<td>custom web app, external software-as-service, storage-as-service</td>
<td></td>
</tr>
<tr>
<td>Missing Function Level Access Control</td>
<td>Most web applications verify function level access rights before making that functionality visible in the UI. However, applications need to perform the same access control checks on the server when each function is accessed. If requests are not verified, attackers will be able to forge requests in order to access functionality without proper authorization.</td>
<td>ELEVATION OF PRIVILEGES</td>
<td>custom web app, external software-as-service, storage-as-service</td>
<td></td>
</tr>
<tr>
<td>Over-privileged application and accounts</td>
<td>The user accounts or the execution of privileged programs is exploited to gain access to resources without authorization, with the goal of compromising the system.</td>
<td>ELEVATION OF PRIVILEGES</td>
<td>custom web app, storage-as-service</td>
<td></td>
</tr>
</tbody>
</table>
### Sensitive Data Exposure

Many web applications do not properly protect sensitive data, such as credit cards, tax IDs, and authentication credentials. Attackers may steal or modify such weakly protected data to conduct credit card fraud, identity theft, or other crimes. Sensitive data deserves extra protection such as encryption at rest or in transit, as well as special precautions when exchanged with the browser.

### Unauthorized access to admin interface

This attack is different from the previous one since it aims at obtaining directly the privileged access to the administration interface, in order to have total control over the system.

### Unvalidated Redirects and Forwards

Web applications frequently redirect and forward users to other pages and websites, and use untrusted data to determine the destination pages. Without proper validation, attackers can redirect victims to phishing or malware sites, or use forwards to access unauthorized pages.

### Weak Identity, Credential & Access Management

The lack of scalable systems for the management of identities and of multiple-factor authentication systems, the use of weak passwords and the scarce automatic update of cryptographic keys, are all high-risk

### SPOOFING

- custom web app,
- external software-as-service,
- storage-as-service

The component hosts or manages sensitive data

### ELEVATION OF PRIVILEGES

- custom web app,
- external software-as-service,
- storage-as-service

The component exposes an administration interface or manages different roles

### TAMPERING

- custom web app,
- external software-as-service,
- storage-as-service

The component does not have mechanisms to validate links

### INFORMATION DISCLOSURE

- custom web app,
- external software-as-service,
- storage-as-service
<table>
<thead>
<tr>
<th><strong>Sniffing Storage Traffic</strong></th>
<th>The traffic generated by a storage service on dedicated or shared networks can be observed revealing data, metadata and information on adopted protocols.</th>
<th>TAMPERING</th>
<th>• storage-as-service</th>
<th>The component sends storage traffic</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Snooping on Buffer Cache</strong></td>
<td>Many file systems use buffer caches to read/write storage blocks from/to the storage support. This is independent from the technology used. The buffer caches are assigned on demand. A malicious user may read the cache and access storage blocks, to access the information for which he is not authorized.</td>
<td>TAMPERING</td>
<td>• storage-as-service</td>
<td>The component uses caching.</td>
</tr>
<tr>
<td><strong>Snooping on Deleted Storage Blocks</strong></td>
<td>In most file systems, storage blocks are allocated to files on demand. When a file is deleted, the storage block contents are not necessarily erased. Rather, most of the storage systems implement file deletion by erasing the file name and links from metadata and deleting the file i-node. Thus, data contents can be left un-erased in deleted and now free storage blocks. By accessing these storage blocks, it is possible for an attacker to gain access to sensitive data</td>
<td>TAMPERING</td>
<td>• storage as service</td>
<td></td>
</tr>
<tr>
<td><strong>Snooping on DEALlocated Memory</strong></td>
<td>Although most modern software deallocate data in memory after its last usage, it is possible for attackers to snoop on deallocated memory because the content of freed memory stays intact until it gets overwritten.</td>
<td>TAMPERING</td>
<td>• custom web app, • external software-as-service, • storage-as-service</td>
<td></td>
</tr>
</tbody>
</table>
Chow et al. point out in [9] that after deallocation, sensitive data such as passwords, social security numbers, and credit card numbers, often remain in memory indefinitely, possibly for days. This increases the risk of exposing sensitive data when a system is compromised, or of data being accidentally leaked due to unexpected feature interactions such as core dumps, logging, etc. One solution to this problem is to reduce data lifetime by zeroing at time of deallocation.

<table>
<thead>
<tr>
<th>File System Profiling</th>
<th>TAMPERING</th>
</tr>
</thead>
<tbody>
<tr>
<td>File system profiling attacks attempt to use access type, timestamps of last modification, file names, and other file system metadata to gain insight about storage system operation. For example, if a set of files are accessed in regular patterns, the attacker may infer the importance, function, and possibly even the content of these files.</td>
<td>• custom web app, storage-as-service</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Modifying Metadata</th>
<th>TAMPERING</th>
<th>storage-as-service</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modifying metadata will disrupt a storage system. In any file system, if the i-node or file table are corrupted, the storage linked to the metadata cannot be accessed.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Subversion Attacks</th>
<th>TAMPERING</th>
<th>custom web app, storage-as-service</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attacks which modify operating system (OS) commands, kernel system calls, and/or storage system drivers to cause the wrong files, metadata or blocks to be modified or deleted.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Exhausting Log, Data and Metadata Space

Storage systems use different types of logging. In log-structured file systems, the whole file system is a series of logs. An attacker can create a large number of small modifications to fill up the log space and lock up the system. Moreover, an attacker can create a large number of files with random content to use up the available disk space. Finally, an attacker may create many empty/small/hidden files. While each file uses only a small amount of metadata space, a large number of metadata entries will degrade storage system performance.

**DENIAL OF SERVICE**
- storage-as-service

The storage service offers a logging system.

### Creating Redundant Versions

Some versioning file systems, like S4 and Elephant create multiple versions of objects. Taking advantage of this, an attacker may launch a DoS attack by creating multiple versions of objects with minimal changes that will eventually exhaust storage space.

**DENIAL OF SERVICE**
- storage-as-service

The storage system creates multiple versions of objects.

### Exhausting File Handles

In most storage systems, file handles are used to access files, and these are locked until the file is closed. Also, file systems usually have a fixed number of file handles. An attacker may create a DoS by opening up multiple files but not closing them, thereby holding the file handle and degrading storage system performance.

**DENIAL OF SERVICE**
- storage-as-service

The component (storage service) uses a limited number of file handles.

### Deletion of Data

Deleting data or metadata is an extreme DoS attack but also one that is easily detectable.

**DENIAL OF SERVICE**
- storage-as-service
and possibly recoverable given versioning or backups in time or space. If the deleted data is unrecoverable, the cost may range from insignificant to incalculable. Deleting system and network logs is commonly used by attackers to cover their attack traces.

| Storage Device Masquerading | An attack storage device authenticates as a legitimate storage device to the OS in order to access/modify/deny data or metadata | SPOOFING | • storage-as-service | The component allows to add supplementary storage devices |
D2.1: Initial SbD methods for multi-cloud applications

Appendix B. The TSM Engine component Security SLA

```xml
<?xml version="1.0" encoding="UTF-8"?>
<wsag:AgreementOffer
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:wsag="http://schemas.ggf.org/graap/2007/03/ws-agreement"
 xmlns:specs="http://www.specs-project.eu/resources/schemas/xml/SLAtemplate"
 xmlns:MUSA="http://www.specs-project.eu/resources/schemas/xml/SLAtemplate"
 xmlns:nist="http://www.specs-project.eu/resources/schemas/xml/control_frameworks/nist"
 xsi:schemaLocation="http://schemas.ggf.org/graap/2007/03/ws-agreement wsag.xsd
 http://www.specs-project.eu/resources/schemas/xml/SLAtemplate SLAtemplate.xsd
 http://www.specs-project.eu/resources/schemas/xml/control_frameworks/nist nist.xsd">

<wsag:Name>MUSA_SLA_TEMPLATE</wsag:Name>
<wsag:Context>
<wsag:AgreementInitiator>$SPECS-CUSTOMER</wsag:AgreementInitiator>
<wsag:AgreementResponder>$SPECS-APPLICATION</wsag:AgreementResponder>
<wsag:ServiceProvider>AgreementResponder</wsag:ServiceProvider>
<wsag:ExpirationTime>2014-02-02T06:00:00</wsag:ExpirationTime>
<wsag:TemplateName>SPECS_TEMPLATE_v1</wsag:TemplateName>
</wsag:Context>

<wsag:Terms>
<wsag:All>
<wsag:ServiceDescriptionTerm wsag:Name="TSM engine" wsag:ServiceName="custom web app">
<specs:serviceDescription>
<MUSA:Components>
<MUSA:Component name="TSM engine" type="custom web app">
<MUSA:ComponentProperty name="TSM engine"/>
<MUSA:Description>undefined</MUSA:Description>
<MUSA:Threats>
<MUSA:Threat name="Injection" source="OWASP TOP 10 2013"/>
<MUSA:Threat name="Broken Authentication and Session Management" source="OWASP TOP 10 2013"/>
<MUSA:Threat name="Insecure Direct Object References" source="OWASP TOP 10 2013"/>
<MUSA:Threat name="Cross-Site Request Forgery (CSRF)" source="OWASP TOP 10 2013"/>
<MUSA:Threat name="Cross-Site Scripting (XSS)" source="OWASP TOP 10 2013"/>
<MUSA:Threat name="Denial of Service" source="CSA Survey Cloud Computing Top Threats 2015"/>
<MUSA:Threat name="Unauthorized access to admin interface" source="CSA Survey Cloud Computing Top Threats 2015"/>
<MUSA:Threat name="Over-privileged application and accounts" source="CSA Survey Cloud Computing Top Threats 2015"/>
</MUSA:Threats>
</MUSA:Component>
</MUSA:Components>
</specs:serviceDescription>
</MUSA:Component>
</specs:serviceDescription>
</wsag:ServiceDescriptionTerm>
</wsag:All>
</wsag:Terms>
</wsag:AgreementOffer>
```
D2.1: Initial SbD methods for multi-cloud applications

Request for SbD methods for component TSM engine:

- Vulnerability Scanning
- Authenticator Management
- Password-based Authentication
- PKI-based Authentication
- In-Person or Trusted Third-Party Registration
- Hardware Token-based Authentication
- Session Authenticity
- Session Identifiers at Logout
- Unique Session Identifiers with Randomization
D2.1: Initial SbD methods for multi-cloud applications

<nist:description>
</nist:description>

<nist:importance_weight>MEDIUM</nist:importance_weight>
</specs:NISTsecurityControl>

<specs:NISTsecurityControl id="SC-23(5)" name="SESSION AUTHENTICITY | ALLOWED CERTIFICATE AUTHORITIES" control_family="SC" securityControl="23" control_enhancement=""/>

<nist:description>
</nist:description>

<nist:importance_weight>MEDIUM</nist:importance_weight>
</specs:NISTsecurityControl>

<specs:NISTsecurityControl id="RA-5(1)" name="VULNERABILITY SCANNING | UPDATE TOOL CAPABILITY" control_family="RA" securityControl="5" control_enhancement=""/>

<nist:description>
</nist:description>

<nist:importance_weight>MEDIUM</nist:importance_weight>
</specs:NISTsecurityControl>

<specs:NISTsecurityControl id="SI-4" name="INFORMATION SYSTEM MONITORING" control_family="SI" securityControl="4" control_enhancement=""/>

<nist:description>
</nist:description>

<nist:importance_weight>LOW</nist:importance_weight>
</specs:NISTsecurityControl>

<specs:NISTsecurityControl id="SC-5" name="DENIAL OF SERVICE PROTECTION" control_family="SC" securityControl="5" control_enhancement=""/>

<nist:description>
</nist:description>

<nist:importance_weight>MEDIUM</nist:importance_weight>
</specs:NISTsecurityControl>

<specs:NISTsecurityControl id="SC-5(3)" name="DENIAL OF SERVICE PROTECTION | DETECTION / MONITORING" control_family="SC" securityControl="5" control_enhancement=""/>

<nist:description>
</nist:description>

<nist:importance_weight>MEDIUM</nist:importance_weight>
</specs:NISTsecurityControl>

<specs:NISTsecurityControl id="SC-12" name="CRYPTOGRAPHIC KEY ESTABLISHMENT AND MANAGEMENT" control_family="SC" securityControl="12" control_enhancement=""/>

<nist:description>
</nist:description>

<nist:importance_weight>LOW</nist:importance_weight>
</specs:NISTsecurityControl>

<specs:NISTsecurityControl id="SC-12(2)" name="CRYPTOGRAPHIC KEY ESTABLISHMENT AND MANAGEMENT | SYMMETRIC KEYS" control_family="SC" securityControl="12" control_enhancement=""/>

<nist:description>
</nist:description>

<nist:importance_weight>LOW</nist:importance_weight>
</specs:NISTsecurityControl>

<specs:NISTsecurityControl id="SC-12(3)" name="CRYPTOGRAPHIC KEY ESTABLISHMENT AND MANAGEMENT | ASYMMETRIC KEYS" control_family="SC" securityControl="12" control_enhancement=""/>

<nist:description>
</nist:description>

<nist:importance_weight>LOW</nist:importance_weight>
D2.1: Initial SbD methods for multi-cloud applications

<spects:NISTsecurityControl id="CA-7" name="CONTINUOUS MONITORING" control_family="CA" securityControl="7" control_enhancement=""/>

<nist:description/>

<nist:importance_weight>LOW</nist:importance_weight>

<spects:NISTsecurityControl id="SC-13" name="CRYPTOGRAPHIC PROTECTION" control_family="SC" securityControl="13" control_enhancement=""/>

<nist:description/>

<nist:importance_weight>LOW</nist:importance_weight>

<spects:security_metrics>
  <spects:Metric name="Vulnerability Measure" referenceId="">
    <spects:MetricDefinition>
      <spects:unit name="%">
        <spects:intervalUnit>
          <spects:intervalItemsType>integer</spects:intervalItemsType>
          <spects:intervalItemStart>0</spects:intervalItemStart>
          <spects:intervalItemStop>100</spects:intervalItemStop>
          <spects:intervalItemStep>1</spects:intervalItemStep>
        </spects:intervalUnit>
      </spects:unit>
      <spects:scale>
        <spects:Quantitative>Ratio</spects:Quantitative>
      </spects:scale>
      <spects:expression>(N/T) *100</spects:expression>
      <spects:definition>It measures the efficiency in percentage (%) of high vulnerabilities mitigated within organizationally defined time periods after discovery.

      Strategic Goal: Ensure an environment of comprehensive security and accountability for personnel, facilities, and products.

      Information Security Goal: Ensure all vulnerabilities are identified and mitigated.</spects:definition>
    </spects:MetricDefinition>
  </spects:Metric>
</spects:security_metrics>
D2.1: Initial SbD methods for multi-cloud applications

<specs:ParameterDefinition name="T" referenceId="">
  <specs:definition>T: Number of high vulnerabilities identified within the time period</specs:definition>
  <specs:parameterType>integer</specs:parameterType>
  <specs:note></specs:note>
</specs:ParameterDefinition>

<specs:MetricRules>
  <specs:MetricRule>
    <specs:ruleDefinitionId></specs:ruleDefinitionId>
    <specs:value></specs:value>
    <specs:note></specs:note>
  </specs:MetricRule>
</specs:MetricRules>

<specs:MetricParameters>
  <specs:MetricParameter>
    <specs:parameterDefinitionId>Result</specs:parameterDefinitionId>
    <specs:value>0</specs:value>
    <specs:note>It is the percent parameter.</specs:note>
  </specs:MetricParameter>
</specs:MetricParameters>

<specs:Metric name="Risk Assessment Vulnerability Measure" referenceId="">
  <specs:MetricDefinition>
    <specs:unit name="%">
      <specs:intervalUnit>
        <specs:intervalItemsType>integer</specs:intervalItemsType>
        <specs:intervalItemStart>0</specs:intervalItemStart>
        <specs:intervalItemStop>100</specs:intervalItemStop>
        <specs:intervalItemStep>1</specs:intervalItemStep>
      </specs:intervalUnit>
    </specs:unit>
    <specs:scale>
      <specs:Quantitative>Ratio</specs:Quantitative>
    </specs:scale>
    <specs:expression>(N/T) *100</specs:expression>
    <specs:definition>It measures the percentage (%) of vulnerabilities remediated within organization-specified time frames.</specs:definition>
    <specs:note></specs:note>
  </specs:MetricDefinition>
</specs:Metric>

<specs:AbstractMetricRuleDefinition>
  <specs:RuleDefinition name="" referenceId="">
    <specs:definition></specs:definition>
    <specs:note></specs:note>
  </specs:RuleDefinition>
</specs:AbstractMetricRuleDefinition>

<specs:AbstractMetricParameterDefinition>
  <specs:ParameterDefinition name="N" referenceId="">
  </specs:ParameterDefinition>
</specs:AbstractMetricParameterDefinition>
<specs:definition>N: Number of vulnerabilities remediated according to POA&M schedule</specs:definition>

<specs:ParameterDefinition name="T" referenceId="">
<specs:definition>T: Total number of POA&M documented vulnerabilities identified through vulnerability scans</specs:definition>
<specs:parameterType>integer</specs:parameterType>
<specs:note></specs:note>
</specs:ParameterDefinition>
</specs:AbstractMetricParameterDefinition>

<specs:MetricRules>
<specs:MetricRule>
<specs:ruleDefinitionId></specs:ruleDefinitionId>
<specs:value></specs:value>
<specs:note></specs:note>
</specs:MetricRule>
</specs:MetricRules>
<specs:MetricParameters>
<specs:MetricParameter>
<specs:parameterDefinitionId>Result</specs:parameterDefinitionId>
<specs:value>0</specs:value>
<specs:note>It is the percent parameter.</specs:note>
</specs:MetricParameter>
</specs:MetricParameters>
<specs:note></specs:note>
</specs:Metric>

<specs:Metric name="M13-Scanning Frequency - Basic Scan" referenceId="">
<specs:MetricDefinition>
<specs:unit name="number">
<specs:intervalUnit>
<specs:intervalItemsType>integer</specs:intervalItemsType>
<specs:intervalItemStart>1</specs:intervalItemStart>
<specs:intervalItemStop>24</specs:intervalItemStop>
<specs:intervalItemStep>1</specs:intervalItemStep>
</specs:intervalUnit>
</specs:unit>
<specs:scale>
<specs:Quantitative>Ratio</specs:Quantitative>
</specs:scale>
<specs:expression></specs:expression>
<specs:definition>This metric sets the frequency (in hours) of the basic software vulnerability scanning. For example, for Scanning Frequency - Basic Scan = 24h, SPECS ensures that software vulnerability scanning will be performed at least once every day.</specs:definition>
<specs:note></specs:note>
</specs:MetricDefinition>
D2.1: Initial SbD methods for multi-cloud applications

<specs:AbstractMetricRuleDefinition>
  <specs:RuleDefinition name="" referenceId="">
    <specs:definition></specs:definition>
    <specs:note></specs:note>
  </specs:RuleDefinition>
</specs:AbstractMetricRuleDefinition>

<specs:AbstractMetricParameterDefinition>
  <specs:ParameterDefinition name="" referenceId="">
    <specs:definition></specs:definition>
    <specs:parameterType></specs:parameterType>
    <specs:note></specs:note>
  </specs:ParameterDefinition>
</specs:AbstractMetricParameterDefinition>

<specs:MetricRules>
  <specs:MetricRule>
    <specs:ruleDefinitionId></specs:ruleDefinitionId>
    <specs:value></specs:value>
    <specs:note></specs:note>
  </specs:MetricRule>
</specs:MetricRules>

<specs:MetricParameters>
  <specs:MetricParameter>
    <specs:parameterDefinitionId></specs:parameterDefinitionId>
    <specs:value>24</specs:value>
    <specs:note></specs:note>
  </specs:MetricParameter>
</specs:MetricParameters>

<specs:Metric name="M22-Scanning Frequency - Extended Scan" referenceId="">
  <specs:MetricDefinition>
    <specs:unit name="number">
      <specs:intervalUnit>
        <specs:intervalItemsType>integer</specs:intervalItemsType>
        <specs:intervalItemStart>1</specs:intervalItemStart>
        <specs:intervalItemStop>24</specs:intervalItemStop>
        <specs:intervalItemStep>1</specs:intervalItemStep>
      </specs:intervalUnit>
    </specs:unit>
    <specs:scale>
      <specs:Quantitative>Ratio</specs:Quantitative>
    </specs:scale>
    <specs:expression> </specs:expression>
    <specs:definition>This metric sets the frequency (in hours) of the extended software vulnerability scanning. For example, for Scanning Frequency - Extended Scan = 48h, SPECS ensures that

USA
software vulnerability scanning will be performed at least once every two days. Scanning is performed with two scanners and both scanning reports are joint.

```xml
<specs:MetricDefinition>
  <specs:Note/>
</specs:MetricDefinition>
<specs:AbstractMetricRuleDefinition>
  <specs:RuleDefinition name="" referenceId=""/>
  <specs:Note/>
</specs:RuleDefinition>
</specs:AbstractMetricRuleDefinition>
<specs:AbstractMetricParameterDefinition>
  <specs:ParameterDefinition name="" referenceId=""/>
  <specs:Note/>
</specs:ParameterDefinition>
</specs:AbstractMetricParameterDefinition>
<specs:MetricRules>
  <specs:MetricRule>
    <specs:RuleDefinitionId/>
    <specs:Value/>
    <specs:Note/>
  </specs:MetricRule>
  <specs:MetricRules>
  <specs:MetricParameter>
    <specs:ParameterDefinitionId/>
    <specs:Value>24</specs:Value>
    <specs:Note/>
  </specs:MetricParameter>
  <specs:MetricParameters>
  <specs:MetricParameter>
    <specs:ParameterDefinitionId/>
    <specs:Value>24</specs:Value>
    <specs:Note/>
  </specs:MetricParameter>
  <specs:MetricParameters>
  <specs:MetricParameter>
    <specs:ParameterDefinitionId/>
    <specs:Value>24</specs:Value>
    <specs:Note/>
  </specs:MetricParameter>
  <specs:MetricParameters>
  <specs:MetricParameter>
    <specs:ParameterDefinitionId/>
    <specs:Value>24</specs:Value>
    <specs:Note/>
  </specs:MetricParameter>
  <specs:MetricParameters>
  <specs:MetricParameter>
    <specs:ParameterDefinitionId/>
    <specs:Value>24</specs:Value>
    <specs:Note/>
  </specs:MetricParameter>
  <specs:MetricParameters>
  <specs:MetricParameter>
    <specs:ParameterDefinitionId/>
    <specs:Value>24</specs:Value>
    <specs:Note/>
  </specs:MetricParameter>
  <specs:MetricParameters>
  <specs:MetricParameter>
    <specs:ParameterDefinitionId/>
    <specs:Value>24</specs:Value>
    <specs:Note/>
  </specs:MetricParameter>
  <specs:MetricParameters>
  <specs:MetricParameter>
    <specs:ParameterDefinitionId/>
    <specs:Value>24</specs:Value>
    <specs:Note/>
  </specs:MetricParameter>
  <specs:MetricParameters>
  <specs:MetricParameter>
    <specs:ParameterDefinitionId/>
    <specs:Value>24</specs:Value>
    <specs:Note/>
  </specs:MetricParameter>
  <specs:MetricParameters>
  <specs:MetricParameter>
    <specs:ParameterDefinitionId/>
    <specs:Value>24</specs:Value>
    <specs:Note/>
  </specs:MetricParameter>
  <specs:MetricParameters>
  <specs:MetricParameter>
    <specs:ParameterDefinitionId/>
    <specs:Value>24</specs:Value>
    <specs:Note/>
  </specs:MetricParameter>
  <specs:MetricParameters>
  <specs:MetricParameter>
    <specs:ParameterDefinitionId/>
    <specs:Value>24</specs:Value>
    <specs:Note/>
  </specs:MetricParameter>
  <specs:MetricParameters>
  <specs:MetricParameter>
    <specs:ParameterDefinitionId/>
    <specs:Value>24</specs:Value>
    <specs:Note/>
  </specs:MetricParameter>
  <specs:MetricParameters>
  <specs:MetricParameter>
    <specs:ParameterDefinitionId/>
    <specs:Value>24</specs:Value>
    <specs:Note/>
  </specs:MetricParameter>
  <specs:MetricParameters>
  <specs:MetricParameter>
    <specs:ParameterDefinitionId/>
    <specs:Value>24</specs:Value>
    <specs:Note/>
  </specs:MetricParameter>
  <specs:MetricParameters>
  <specs:MetricParameter>
    <specs:ParameterDefinitionId/>
    <specs:Value>24</specs:Value>
    <specs:Note/>
  </specs:MetricParameter>
  <specs:MetricParameters>
  <specs:MetricParameter>
    <specs:ParameterDefinitionId/>
    <specs:Value>24</specs:Value>
    <specs:Note/>
  </specs:MetricParameter>
  <specs:MetricParameters>
  <specs:MetricParameter>
    <specs:ParameterDefinitionId/>
    <specs:Value>24</specs:Value>
    <specs:Note/>
  </specs:MetricParameter>
  <specs:MetricParameters>
  <specs:MetricParameter>
    <specs:ParameterDefinitionId/>
    <specs:Value>24</specs:Value>
    <specs:Note/>
  </specs:MetricParameter>
  <specs:MetricParameters>
  <specs:MetricParameter>
    <specs:ParameterDefinitionId/>
    <specs:Value>24</specs:Value>
    <specs:Note/>
  </specs:MetricParameter>
  <specs:MetricParameters>
  <specs:MetricParameter>
    <specs:ParameterDefinitionId/>
    <specs:Value>24</specs:Value>
    <specs:Note/>
  </specs:MetricParameter>
  <specs:MetricParameters>
  <specs:MetricParameter>
    <specs:ParameterDefinitionId/>
    <specs:Value>24</specs:Value>
    <specs:Note/>
This metric sets the frequency (in hours) of checks for updates and upgrades of vulnerable installed libraries. At the implementation time, SPECS generates vulnerability list, performs the vulnerability scan of the system, and generates scanning report. Then periodically checks for available updates and upgrades of libraries on which vulnerabilities have been detected. For example, for Up Report Frequency = 24h, SPECS ensures that checks for updates and upgrades are performed at least once every day.

```
<specs:Metric name="Resilience to attacks" referenceId=""/>
<specs:MetricDefinition>
  <specs:unit name="n/a">
    <specs:enumUnit>
      <specs:enumItemsType>string</specs:enumItemsType>
      <specs:enumItems>
        <specs:enumItem>
          <specs:value>yes</specs:value>
          <specs:description/>
        </specs:enumItem>
      </specs:enumItems>
    </specs:enumUnit>
  </specs:unit>
</specs:MetricDefinition>
```
D2.1: Initial SbD methods for multi-cloud applications

Vulnerability and malware

The application should be attack tolerant.

USA
D2.1: Initial SbD methods for multi-cloud applications

<specs:enumUnit>
  <specs:enumItemsType>string</specs:enumItemsType>
  <specs:enumItems>
    <specs:enumItem>
      <specs:value>yes</specs:value>
      <specs:description></specs:description>
    </specs:enumItem>
    <specs:enumItem>
      <specs:value>no</specs:value>
      <specs:description></specs:description>
    </specs:enumItem>
  </specs:enumItems>
</specs:enumUnit>

<specs:unit>
  <specs:scale>
    <specs:Qualitative>Nominal</specs:Qualitative>
  </specs:scale>
</specs:scale>

<specs:definition>The application provides the date and time of last malware and vulnerability scan as well as the number of malwares and vulnerabilities found during the scan</specs:definition>
</specs:MetricDefinition>

<specs:RuleDefinition name="" referenceId="">
  <specs:definition></specs:definition>
  <specs:note></specs:note>
</specs:RuleDefinition>

<specs:RuleDefinition name="" referenceId="">
  <specs:definition></specs:definition>
  <specs:note></specs:note>
</specs:RuleDefinition>

<specs:ParameterDefinition name="" referenceId="">
  <specs:definition></specs:definition>
  <specs:parameterType></specs:parameterType>
  <specs:note></specs:note>
</specs:ParameterDefinition>

<specs:MetricRules>
  <specs:MetricRule>
    <specs:ruleDefinitionId></specs:ruleDefinitionId>
    <specs:value></specs:value>
    <specs:note></specs:note>
  </specs:MetricRule>
</specs:MetricRules>

<specs:MetricParameters>
  <specs:MetricParameter>
    <specs:parameterDefinitionId></specs:parameterDefinitionId>
    <specs:value>yes</specs:value>
    <specs:note></specs:note>
  </specs:MetricParameter>
</specs:MetricParameters>
D2.1: Initial SbD methods for multi-cloud applications

<specs:Metric name="M2-Level of Diversity" referenceId="">
  <specs:MetricDefinition>
    <specs:unit name="number">
      <specs:intervalUnit>
        <specs:intervalItemsType>integer</specs:intervalItemsType>
        <specs:intervalItemStart>1</specs:intervalItemStart>
        <specs:intervalItemStop>100</specs:intervalItemStop>
        <specs:intervalItemStep>1</specs:intervalItemStep>
      </specs:intervalUnit>
    </specs:unit>
    <specs:scale>Ratio</specs:scale>
    <specs:expression/>
    <specs:definition>This metric sets the number of different Web Container types available on target VMs. For example for Level of Diversity=2 SPECS ensures that there are at least two different types of Web Container available.</specs:definition>
    <specs:note/>
  </specs:MetricDefinition>
  <specs:AbstractMetricRuleDefinition>
    <specs:RuleDefinition name="" referenceId=""/>
    <specs:definition/>
    <specs:note/>
  </specs:RuleDefinition>
  <specs:AbstractMetricRuleDefinition>
    <specs:RuleDefinition name="" referenceId=""/>
    <specs:definition/>
    <specs:note/>
  </specs:RuleDefinition>
  <specs:AbstractMetricParameterDefinition>
    <specs:ParameterDefinition name="" referenceId=""/>
    <specs:definition/>
    <specs:parameterType/>
    <specs:note/>
  </specs:ParameterDefinition>
  <specs:MetricRules>
    <specs:MetricRule>
      <specs:ruleDefinitionId></specs:ruleDefinitionId>
      <specs:value></specs:value>
      <specs:note></specs:note>
    </specs:MetricRule>
  </specs:MetricRules>
  <specs:MetricParameters>
    <specs:MetricParameter>
      <specs:parameterDefinitionId></specs:parameterDefinitionId>
      <specs:value>1</specs:value>
      <specs:note></specs:note>
    </specs:MetricParameter>
  </specs:MetricParameters>
</specs:Metric>
D2.1: Initial SbD methods for multi-cloud applications

<specs:Metric name="M14-List Update Frequency" referenceId="">
<specs:MetricDefinition>
    <specs:unit name="number">
        <specs:intervalUnit>
            <specs:intervalItem>integer</specs:intervalItem>
            <specs:intervalItemStart>1</specs:intervalItemStart>
            <specs:intervalItemStop>24</specs:intervalItemStop>
            <specs:intervalItemStep>1</specs:intervalItemStep>
        </specs:intervalUnit>
    </specs:unit>
    <specs:scale>
        <specs:Quantitative>Ratio</specs:Quantitative>
        <specs:expression></specs:expression>
    </specs:scale>
    <specs:definition>This metric sets the frequency (in hours) of updates of the list of known/disclosed vulnerabilities from OVAL/NVD databases. For example, for List Update Frequency=12h, SPECS ensures that the list of known/disclosed vulnerabilities will be updated at least once every 12 hours.</specs:definition>
</specs:MetricDefinition>
<specs:MetricRules>
    <specs:MetricRule>
        <specs:ruleDefinitionId></specs:ruleDefinitionId>
        <specs:value></specs:value>
        <specs:note></specs:note>
    </specs:MetricRule>
</specs:MetricRules>
<specs:MetricParameters>
    <specs:MetricParameter>
        <specs:parameterDefinitionId></specs:parameterDefinitionId>
    </specs:MetricParameter>
</specs:MetricParameters>
</specs:Metric>
D2.1: Initial SnD methods for multi-cloud applications

<s:Metric name="Personal data disclosure" referenceId="">
  <s:MetricDefinition>
    <s:unit name="n/a">
      <s:enumUnit>
        <s:enumItemsType>string</s:enumItemsType>
        <s:enumItems>
          <s:enumItem>
            <s:value>yes</s:value>
            <s:description></s:description>
          </s:enumItem>
          <s:enumItem>
            <s:value>no</s:value>
            <s:description></s:description>
          </s:enumItem>
        </s:enumItems>
      </s:enumUnit>
    </s:unit>
    <s:scale>
      <s:Qualitative>Nominal</s:Qualitative>
    </s:scale>
    <s:expression/>
    <s:definition>Capability of a provider of correlating hosted sensitive information to people they belong to</s:definition>
  </s:MetricDefinition>
  <s:AbstractMetricRuleDefinition>
    <s:RuleDefinition name="" referenceId="">
      <s:definition/>
    </s:RuleDefinition>
  </s:AbstractMetricRuleDefinition>
  <s:AbstractMetricParameterDefinition>
    <s:ParameterDefinition name="" referenceId="">
      <s:definition/>
    </s:ParameterDefinition>
  </s:AbstractMetricParameterDefinition>
</s:Metric>

<s:Metric name="" referenceId=""/>
  <s:MetricDefinition>
    <s:unit name=""></s:unit>
    <s:scale>
      <s:Qualitative> </s:Qualitative>
    </s:scale>
    <s:expression/>
    <s:definition/>
  </s:MetricDefinition>
  <s:AbstractMetricRuleDefinition>
    <s:RuleDefinition name="" referenceId="">
      <s:definition/>
    </s:RuleDefinition>
  </s:AbstractMetricRuleDefinition>
  <s:AbstractMetricParameterDefinition>
    <s:ParameterDefinition name="" referenceId="">
      <s:definition/>
    </s:ParameterDefinition>
  </s:AbstractMetricParameterDefinition>
</s:MetricRules>
<specs:MetricRule>
</specs:MetricRules>
<specs:MetricParameters>
<specs:MetricParameter>
<specs:parameterDefinitionId></specs:parameterDefinitionId>
<specs:value>yes</specs:value>
<specs:note></specs:note>
</specs:MetricParameter>
</specs:MetricParameters>
</specs:Metric>

<specs:Metric name="Database activity monitoring" referenceId="">
<specs:MetricDefinition>
<specs:unit name="n/a">
<specs:enumUnit>
<specs:enumItemsType>string</specs:enumItemsType>
<specs:enumItems>
<specs:enumItem>
<specs:value>yes</specs:value>
<specs:description></specs:description>
</specs:enumItem>
<specs:enumItem>
<specs:value>no</specs:value>
<specs:description></specs:description>
</specs:enumItem>
</specs:enumUnits>
</specs:unit>
<specs:scale>
<specs:Qualitative>Nominal</specs:Qualitative>
</specs:scale>
<specs:expression></specs:expression>
<specs:definition>Monitors the activity of privileged users (superusers) in databases and recognizes abnormal behaviour</specs:definition>
</specs:Metric>
</specs:MetricDefinition>
<specs:RuleDefinition name="" referenceId="">
<specs:definition></specs:definition>
</specs:RuleDefinition>
</specs:MetricRule>
<specs:AbstractMetricRuleDefinition>
</specs:RuleDefinition>
</specs:AbstractMetricRuleDefinition>
<specs:AbstractMetricParameterDefinition>
</specs:ParameterDefinition>
</specs:MetricDefinition>
D2.1: Initial SbD methods for multi-cloud applications

```xml
<specs:Metric name="M1-Level of Redundancy" referenceId="">  
  <specs:MetricDefinition>  
    <specs:unit name="number">  
      <specs:intervalUnit>  
        <specs:intervalItemsType>integer</specs:intervalItemsType>  
        <specs:intervalItemStart>1</specs:intervalItemStart>  
        <specs:intervalItemStop>100</specs:intervalItemStop>  
        <specs:intervalItemStep>1</specs:intervalItemStep>  
      </specs:intervalUnit>  
    </specs:unit>  
    <specs:scale>  
      <specs:Quantitative>Ratio</specs:Quantitative>  
    </specs:scale>  
    <specs:expression>  
      This metric sets the number of Web Container replicas. For example, for Level of Redundancy = 3, SPECS ensures that there are at least three Web Containers running.</specs:definition>  
    <specs:definition></specs:definition>  
    <specs:note></specs:note>  
  </specs:MetricDefinition>  
  <specs:AbstractMetricRuleDefinition>  
    <specs:RuleDefinition name="" referenceId="">  
      <specs:definition></specs:definition>  
      <specs:note></specs:note>  
    </specs:RuleDefinition>  
    <specs:AbstractMetricRuleDefinition>  
    </specs:AbstractMetricRuleDefinition>  
  </specs:AbstractMetricRuleDefinition>  
  <specs:AbstractMetricParameterDefinition>  
  </specs:AbstractMetricParameterDefinition>  
</specs:Metric>
```
D2.1: Initial SbD methods for multi-cloud applications

<specs:AbstractMetricParameterDefinition>
  <specs:ParameterDefinition>
    <specs:MetricRules>
      <specs:MetricRule>
        <specs:ruleDefinitionId>
        <specs:value>
        <specs:note>
      </specs:MetricRule>
    </specs:MetricRules>
    <specs:MetricParameters>
      <specs:MetricParameter>
        <specs:parameterDefinitionId>
        <specs:value>3</specs:value>
        <specs:note>
      </specs:MetricParameter>
    </specs:MetricParameters>
  </specs:MetricDefinition>
  <specs:unit name="n/a">
    <specs:enumUnit>
      <specs:enumItemsType>string</specs:enumItemsType>
      <specs:enumItems>
        <specs:enumItem>
          <specs:value>yes</specs:value>
          <specs:description>
        </specs:enumItem>
        <specs:enumItem>
          <specs:value>no</specs:value>
          <specs:description>
        </specs:enumItem>
      </specs:enumItems>
    </specs:enumUnit>
    <specs:scale>
      <specs:Qualitative>Nominal</specs:Qualitative>
    </specs:scale>
    <specs:expression>
    </specs:expression>
    <specs:definition>
      It is a property ensuring that a session key derived from a set of long-term keys cannot be compromised if one of the long-term keys is compromised in the future.
    </specs:definition>
    <specs:note>
  </specs:MetricDefinition>
D2.1: Initial ShD methods for multi-cloud applications

</specs:MetricDefinition>
<specs:AbstractMetricRuleDefinition>
    <specs:RuleDefinition name="" referenceId=""">
        <specs:definition></specs:definition>
        <specs:note></specs:note>
    </specs:RuleDefinition>
</specs:AbstractMetricRuleDefinition>
<specs:AbstractMetricParameterDefinition>
    <specs:ParameterDefinition name="" referenceId=""/>
        <specs:definition></specs:definition>
        <specs:parameterType></specs:parameterType>
        <specs:note></specs:note>
    </specs:ParameterDefinition>
</specs:AbstractMetricParameterDefinition>
<specs:MetricRules>
    <specs:MetricRule>
        <specs:ruleDefinitionId></specs:ruleDefinitionId>
        <specs:value></specs:value>
        <specs:note></specs:note>
    </specs:MetricRule>
</specs:MetricRules>
<specs:MetricParameters>
    <specs:MetricParameter>
        <specs:parameterDefinitionId></specs:parameterDefinitionId>
        <specs:value>yes</specs:value>
        <specs:note></specs:note>
    </specs:MetricParameter>
</specs:MetricParameters>
</specs:Metric>

<specs:Metric name="M19-Client-side Encryption Certification" referenceId=""/>
<specs:MetricDefinition>
    <specs:unit name="n/a">
        <specs:enumUnit>
            <specs:enumItemsType>string</specs:enumItemsType>
            <specs:enumItems>
                <specs:enumItem>
                    <specs:value>yes</specs:value>
                    <specs:description></specs:description>
                </specs:enumItem>
                <specs:enumItem>
                    <specs:value>no</specs:value>
                    <specs:description></specs:description>
                </specs:enumItem>
            </specs:enumItems>
        </specs:enumUnit>
    </specs:unit>
</specs:MetricDefinition>
This metric ensures that the E2EE Client component available at the provided address is certified by ??? and thus grants the security of the encryption.<br>

<specs:Metric name="System and Communication Protection Measure" referenceId="" value="yes"></specs:Metric>
D2.1: Initial SbD methods for multi-cloud applications

It measures the percentage of mobile computers and devices that perform all cryptographic operations using FIPS 140-2 validated cryptographic modules operating in approved modes of operation. Strategic Goal: Accelerate the development and use of an electronic information infrastructure. Information Security Goal: Allocate sufficient resources to adequately protect electronic information infrastructure.

**Ratio**

\[(N/T) \times 100\]

It measures the percentage of mobile computers and devices that perform all cryptographic operations using FIPS 140-2 validated cryptographic modules operating in approved modes of operation. Strategic Goal: Accelerate the development and use of an electronic information infrastructure. Information Security Goal: Allocate sufficient resources to adequately protect electronic information infrastructure.

**Metric Definition**

**Rule Definition**

**Parameter Definition**

**Metric Rules**

**Metric Parameters**

**Result**

0

It is the percent parameter.
<specs:Metric name="M3-TLS Cryptographic Strength" referenceId=""
 specs:MetricDefinition>
 <specs:unit name="level">
  <specs:intervalUnit>
   <specs:intervalItemsType>integer</specs:intervalItemsType>
   <specs:intervalItemStart>1</specs:intervalItemStart>
   <specs:intervalItemStop>8</specs:intervalItemStop>
   <specs:intervalItemStep>1</specs:intervalItemStep>
  </specs:intervalUnit>
 </specs:unit>
 <specs:scale>
  <specs:Quantitative>Ratio</specs:Quantitative>
 </specs:scale>
 <specs:expression> </specs:expression>
 <specs:definition>It is a measure of the expected number of operations required to
 defeat a cryptographic mechanism. The values (level 1-8) are based on ECRYPT II recommendations
 2012.</specs:definition>
 <specs:note></specs:note>
 </specs:MetricDefinition>
 <specs:AbstractMetricRuleDefinition>
  <specs:RuleDefinition name="" referenceId=""
   <specs:definition>
    <specs:note></specs:note>
   </specs:definition>
  </specs:RuleDefinition>
 </specs:AbstractMetricRuleDefinition>
 <specs:AbstractMetricParameterDefinition>
  <specs:ParameterDefinition name="Level" referenceId=""
   <specs:definition>This integer indicates the chosen level.</specs:definition>
   <specs:parameterType></specs:parameterType>
   <specs:note></specs:note>
  </specs:ParameterDefinition>
 </specs:AbstractMetricParameterDefinition>
 <specs:MetricRules>
  <specs:MetricRule>
   <specs:ruleDefinitionId></specs:ruleDefinitionId>
   <specs:value></specs:value>
   <specs:note></specs:note>
  </specs:MetricRule>
 </specs:MetricRules>
 <specs:MetricParameters>
  <specs:MetricParameter>
   <specs:parameterDefinitionId></specs:parameterDefinitionId>
   <specs:value>6</specs:value>
   <specs:note></specs:note>
  </specs:MetricParameter>
 </specs:MetricParameters>
 <specs:note></specs:note>
 </specs:Metric>
D2.1: Initial SbD methods for multi-cloud applications

<specs:Metric name="SQL injection" referenceId="" />
<specs:MetricDefinition>
<specs:unit name="n/a"
<specs:enumUnit>
<specs:enumItemsType>string</specs:enumItemsType>
<specs:enumItems>
<specs:enumItem>
<specs:value>yes</specs:value>
<specs:description></specs:description>
</specs:enumItem>
<specs:enumItem>
<specs:value>no</specs:value>
<specs:description></specs:description>
</specs:enumItems>
</specs:enumUnit>
</specs:unit>
<specs:scale>
<specs:Qualitative>Nominal</specs:Qualitative>
</specs:scale>
<specs:expression></specs:expression>
<specs:definition>By monitoring the queries, it is possible to identify SQL injection attempts</specs:definition>
<specs:note></specs:note>
</specs:MetricDefinition>
<specs:AbstractMetricRuleDefinition>
<specs:RuleDefinition name="" referenceId="" />
<specs:definition></specs:definition>
<specs:note></specs:note>
</specs:RuleDefinition>
</specs:AbstractMetricRuleDefinition>
<specs:AbstractMetricParameterDefinition>
<specs:ParameterDefinition name="" referenceId="" />
<specs:definition></specs:definition>
<specs:parameterType></specs:parameterType>
<specs:note></specs:note>
</specs:ParameterDefinition>
</specs:AbstractMetricParameterDefinition>
<specs:MetricRules>
<specs:MetricRule>
<specs:ruleDefinitionId></specs:ruleDefinitionId>
<specs:value></specs:value>
<specs:note></specs:note>
</specs:MetricRule>
</specs:MetricRules>
<specs:MetricParameters>


USA
D2.1: Initial ShD methods for multi-cloud applications

<specs:MetricParameter>
  <specs:parameterDefinitionId>
    <specs:value>yes</specs:value>
    <specs:note></specs:note>
  </specs:MetricParameter>
</specs:MetricParameters>

<specs:Metric name="Identity Assurance" referenceId="">
  <specs:MetricDefinition>
    <specs:unit name="level">
      <specs:intervalUnit>
        <specs:intervalItemsType>integer</specs:intervalItemsType>
        <specs:intervalItemStart>0</specs:intervalItemStart>
        <specs:intervalItemStop>4</specs:intervalItemStop>
        <specs:intervalItemStep>1</specs:intervalItemStep>
      </specs:intervalUnit>
    </specs:unit>
    <specs:scale>
      <specs:Quantitative>Ratio</specs:Quantitative>
    </specs:scale>
    <specs:expression></specs:expression>
    <specs:definition>This metric describes the quality of the authentication mechanisms in place.</specs:definition>
  </specs:MetricDefinition>
  <specs:AbstractMetricRuleDefinition>
    <specs:RuleDefinition name="" referenceId="">
      <specs:definition></specs:definition>
      <specs:note></specs:note>
    </specs:RuleDefinition>
  </specs:AbstractMetricRuleDefinition>
  <specs:AbstractMetricParameterDefinition>
    <specs:ParameterDefinition name="Level" referenceId="">
      <specs:definition>This integer indicates the chosen level.</specs:definition>
      <specs:parameterType></specs:parameterType>
      <specs:note></specs:note>
    </specs:ParameterDefinition>
  </specs:AbstractMetricParameterDefinition>
  <specs:MetricRules>
    <specs:MetricRule>
      <specs:ruleDefinitionId></specs:ruleDefinitionId>
      <specs:value></specs:value>
      <specs:note></specs:note>
    </specs:MetricRule>
  </specs:MetricRules>
</specs:MetricParameters>
D2.1: Initial SbD methods for multi-cloud applications

```xml
<specs:MetricParameter>
    <specs:parameterDefinitionId/>
    <specs:value>2</specs:value>
    <specs:note/>
</specs:MetricParameter>
</specs:MetricParameters>
</specs:Metric>

</specs:security_metrics>
</specs:serviceDescription>
</wsag:ServiceDescriptionTerm>

<wsag:GuaranteeTerm wsag:Name="/specs:capability[@name=TSM engine]" wsag:Obligated=""

<wsag:ServiceLevelObjective>
<wsag:CustomServiceLevel>
<specs:objectiveList>

    <specs:SLO SLO_ID="0">
        <specs:MetricREF>Vulnerability Measure</specs:MetricREF>
        <specs:SLOexpression>
            <specs:oneOpExpression operator="ge (>=)" operand="0"/>
        </specs:SLOexpression>
        <specs:importance_weight>MEDIUM</specs:importance_weight>
    </specs:SLO>

    <specs:SLO SLO_ID="1">
        <specs:MetricREF>Risk Assessment Vulnerability Measure</specs:MetricREF>
        <specs:SLOexpression>
            <specs:oneOpExpression operator="ge (>=)" operand="0"/>
        </specs:SLOexpression>
        <specs:importance_weight>MEDIUM</specs:importance_weight>
    </specs:SLO>

    <specs:SLO SLO_ID="2">
        <specs:MetricREF>M13-Scanning Frequency - Basic Scan</specs:MetricREF>
        <specs:SLOexpression>
            <specs:oneOpExpression operator="eq (=)" operand="24"/>
        </specs:SLOexpression>
        <specs:importance_weight>MEDIUM</specs:importance_weight>
    </specs:SLO>

    <specs:SLO SLO_ID="3">
        <specs:MetricREF>M22-Scanning Frequency - Extended Scan</specs:MetricREF>
        <specs:SLOexpression>
            <specs:oneOpExpression operator="eq (=)" operand="24"/>
        </specs:SLOexpression>
        <specs:importance_weight>MEDIUM</specs:importance_weight>
    </specs:SLO>
```

USA
<specs:importance_weight>MEDIUM</specs:importance_weight>
</specs:SLO>

<specs:SLO SLO_ID="4">
<specs:MetricREF>M23-Up Report Frequency</specs:MetricREF>
<specs:SLOexpression>
<specs:oneOpExpression operator="eq (=)" operand="24"/>
</specs:SLOexpression>
<specs:importance_weight>MEDIUM</specs:importance_weight>
</specs:SLO>

<specs:SLO SLO_ID="5">
<specs:MetricREF>Resilience to attacks</specs:MetricREF>
<specs:SLOexpression>
<specs:oneOpExpression operator="eq (=)" operand="yes"/>
</specs:SLOexpression>
<specs:importance_weight>MEDIUM</specs:importance_weight>
</specs:SLO>

<specs:SLO SLO_ID="6">
<specs:MetricREF>Vulnerability and malware</specs:MetricREF>
<specs:SLOexpression>
<specs:oneOpExpression operator="eq (=)" operand="yes"/>
</specs:SLOexpression>
<specs:importance_weight>MEDIUM</specs:importance_weight>
</specs:SLO>

<specs:SLO SLO_ID="7">
<specs:MetricREF>M2-Level of Diversity</specs:MetricREF>
<specs:SLOexpression>
<specs:oneOpExpression operator="ge (>=)" operand="1"/>
</specs:SLOexpression>
<specs:importance_weight>MEDIUM</specs:importance_weight>
</specs:SLO>

<specs:SLO SLO_ID="8">
<specs:MetricREF>M14-List Update Frequency</specs:MetricREF>
<specs:SLOexpression>
<specs:oneOpExpression operator="eq (=)" operand="24"/>
</specs:SLOexpression>
<specs:importance_weight>MEDIUM</specs:importance_weight>
</specs:SLO>

<specs:SLO SLO_ID="9">
<specs:MetricREF>Personal data disclosure</specs:MetricREF>
<specs:SLOexpression>
<specs:oneOpExpression operator="eq (=)" operand="yes"/>
</specs:SLOexpression>
<specs:importance_weight>MEDIUM</specs:importance_weight>
</specs:SLO>
D2.1: Initial SbD methods for multi-cloud applications

<specs:SLO SLO_ID="10">
<specs:MetricREF>Database activity monitoring</specs:MetricREF>
<specs:SLOexpression>
<specs:oneOpExpression operator="eq (=)" operand="yes"/>
</specs:SLOexpression>
<specs:importance_weight>MEDIUM</specs:importance_weight>
</specs:SLO>

<specs:SLO SLO_ID="11">
<specs:MetricREF>M1-Level of Redundancy</specs:MetricREF>
<specs:SLOexpression>
<specs:oneOpExpression operator="ge (>=)" operand="3"/>
</specs:SLOexpression>
<specs:importance_weight>MEDIUM</specs:importance_weight>
</specs:SLO>

<specs:SLO SLO_ID="12">
<specs:MetricREF>M4-Forward Secrecy</specs:MetricREF>
<specs:SLOexpression>
<specs:oneOpExpression operator="eq (=)" operand="yes"/>
</specs:SLOexpression>
<specs:importance_weight>MEDIUM</specs:importance_weight>
</specs:SLO>

<specs:SLO SLO_ID="13">
<specs:MetricREF>M19-Client-side Encryption Certification</specs:MetricREF>
<specs:SLOexpression>
<specs:oneOpExpression operator="eq (=)" operand="yes"/>
</specs:SLOexpression>
<specs:importance_weight>MEDIUM</specs:importance_weight>
</specs:SLO>

<specs:SLO SLO_ID="14">
<specs:MetricREF>System and Communication Protection Measure</specs:MetricREF>
<specs:SLOexpression>
<specs:oneOpExpression operator="ge (>=)" operand="0"/>
</specs:SLOexpression>
<specs:importance_weight>MEDIUM</specs:importance_weight>
</specs:SLO>

<specs:SLO SLO_ID="15">
<specs:MetricREF>M3-TLS Cryptographic Strength</specs:MetricREF>
<specs:SLOexpression>
<specs:oneOpExpression operator="eq (=)" operand="6"/>
</specs:SLOexpression>
<specs:importance_weight>MEDIUM</specs:importance_weight>
</specs:SLO>

<specs:SLO SLO_ID="16"/>
D2.1: Initial SbD methods for multi-cloud applications

<specs:MetricREF>SQL injection</specs:MetricREF>
<specs:SLO>
  <specs:oneOpExpression operator="eq (=)" operand="yes"/>
</specs:SLO>
<specs:importance_weight>MEDIUM</specs:importance_weight>
</specs:SLO>

<specs:SLO SLO_ID="17">
<specs:MetricREF>Identity Assurance</specs:MetricREF>
<specs:oneOpExpression operator="eq (=)" operand="2"/>
<specs:importance_weight>MEDIUM</specs:importance_weight>
</specs:SLO>
</specs:objectiveList>
<wsag:CustomServiceLevel>
<wsag:ServiceLevelObjective>
<wsag:BusinessValueList></wsag:BusinessValueList>
<wsag:GuaranteeTerm>
<wsag:All>
<wsag:Terms>
<wsag:AgreementOffer>