



TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

SAMUEL OLAIYA AFOLARANMI
MULTI-CLOUD SECURITY MECHANISMS FOR SMART ENVI-
RONMENTS

Master of Science Thesis

Examiner: Prof. Jose L. Martinez
Lastra
Examiner and topic approved by the
Faculty Council of the Faculty of
Engineering Sciences
on 7th September, 2016.

ABSTRACT

SAMUEL OLAIYA AFOLARANMI: Multi-cloud Security Mechanisms for Smart Environments

Tampere University of technology

Master of Science Thesis, 66 pages, 10 Appendix pages

May 2018

Master's Degree Programme in Automation Engineering

Major: Factory Automation and Industrial Informatics

Examiner: Professor Jose L. Martinez Lastra

Supervisor: Dr. Borja Ramis Ferrer

Keywords: cloud computing, multi-cloud, security mechanisms, smart environments, security ontology, security monitoring, security metrics, threat identification, security controls, risk analysis, security measurement, transparency, security awareness

Achieving transparency and security awareness in cloud environments is a challenging task. It is even more challenging in multi-cloud environments (where application components are distributed across multiple clouds) owing to its complexity. This complexity opens doors to the introduction of threats and makes it difficult to know how the application components are performing and when remedial actions should be taken in the case of an anomaly. Nowadays, many cloud customers are becoming more interested in having a knowledge of their application status, particularly as it relates to the security of the application owing to growing cloud security concerns, which is multi-faceted in multi-cloud environments. This has necessitated the need for adequate visibility and security awareness in multi-cloud environments. However, this is threatened by non-standardization and diverse CSP platforms.

This thesis presents a security evaluation framework for multi-cloud applications. It aims to facilitate transparency and security awareness in multi-cloud applications through adequate evaluation of the application components deployed across different clouds as well as the entire multi-cloud application. This will ensure that the health, internal events and performance of the multi-cloud application can be known. As a result of this, the security status and information about the multi-cloud application can be made available to application owners, cloud service providers and application users. This will increase cloud customers' trust in using multi-clouds and ensure verification of the security status of multi-cloud components at any time desired. The security evaluation framework is based on threat identification and risk analysis, application modelling with ontology, selection of metrics and security controls, application security monitoring, security measurement, decision making and security status visualization.

PREFACE

This thesis was carried out in FAST-Lab, (<http://www.tut.fi/fast>), Tampere University of Technology (TUT), Tampere in the scope of Multi-cloud Secure Applications (MUSA) project (<https://musa-project.eu>). The examiner of this thesis has been Prof. Jose L. Martinez Lastra of Tampere University of Technology.

First of all, I give thanks to Almighty God, the source of my life, wisdom and inspiration, for seeing me through the successful completion of this thesis. I cannot quantify his love, support and guidance.

My appreciation goes to Prof. Jose L. Martinez Lastra for providing a conducive environment that supports and promotes top-notch scientific research. Special thanks to Anne Korhonen, Associate Prof. Andrei Lobov and Luis Enrique Gonzalez Moctezuma for believing in me and giving me the chance to work in FAST-Lab, in the MUSA Project. It was really an interesting experience for me. I thank all the members of MUSA project for their support.

My appreciation also goes to Dr. Borja Ramis Ferrer, my friend and colleague for reviewing this thesis and in particular, for the many collaborations we had via publication of scientific papers and journals. I learnt a lot from you as you really supported and motivated me to achieve success. Gracias amigo!

I appreciate my parents, Mr. Peter Olufemi Afolaranmi and Mrs. Motunrayo Aduke Afolaranmi for their moral and financial support, the prayers, calls and words of encouragement. May God bless you. Special thanks to my siblings; Olatunji, Yetunde and Olaoluwa Afolaranmi for their unending love and support. You guys are awesome and I am proud to have you as siblings. I appreciate my girlfriend, Doris for her patience, understanding and support during the course of this thesis. I love you dearie.

I also appreciate Mrs. Adeshola Adeyoyin, “*my second mum*”, for her prayers, encouragement and financial support at the point of embarking on the journey to Finland for my M.Sc. studies. May God bless you ma.

Finally, I dedicate this thesis to the memory of Mrs. Akinfosile. Your demise came as a big shock to me but God knows best. May God rest your soul. Amen.

Tampere, 22.05.2018

Samuel Olaiya Afolaranmi

LIST OF PUBLICATIONS

- I. J. Puttonen, S. O. Afolaranmi, L. G. Moctezuma, A. Lobov, and J. L. M. Lastra, “Security in Cloud-Based Cyber-Physical Systems” in *2015 10th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC)*, 2015, pp. 671-676.
- II. S. O. Afolaranmi, L. E. G. Moctezuma, M. Rak, V. Casola, E. Rios, and J. L. M. Lastra, “Methodology to Obtain the Security Controls in Multi-cloud Applications,” presented at the *6th International Conference on Cloud Computing and Services Science*, 2016, vol. 1, pp. 327–332.
- III. J. Puttonen, S. O. Afolaranmi, L. G. Moctezuma, A. Lobov, and J. L. M. Lastra, “Enhancing security in cloud-based Cyber-Physical Systems” *Journal of Cloud Computing Research (JCCR) 2016*.
- IV. S. O. Afolaranmi, B. R. Ferrer, W. M. Mohammed, J. L. M. Lastra, M. Ahmad, and R. Harrison, “Providing an access control layer to web-based applications for the industrial domain,” in *2017 IEEE 15th International Conference on Industrial Informatics (INDIN)*, 2017, pp. 1096–1102.
- V. B. R. Ferrer, S. O. Afolaranmi, and J. L. M. Lastra, “Principles and risk assessment of managing distributed ontologies hosted by embedded devices for controlling industrial systems,” in *IECON 2017 - 43rd Annual Conference of the IEEE Industrial Electronics Society*, 2017, pp. 3498–3505.

CONTENTS

1.	INTRODUCTION	1
1.1	Background	1
1.2	Motivation	3
1.3	Problem Statement	4
1.4	Objectives and Research Questions	4
1.5	Limitation	5
1.6	Thesis Structure	5
2.	THEORETICAL BACKGROUND	6
2.1	Smart Environments	6
2.1.1	Threat Modelling in Smart Environments	8
2.1.2	Self-healing and Self-recovery in Smart Environments	10
2.1.3	Managing Device Security in Smart Environments	12
2.2	Cloud Computing and Cloud Systems	14
2.2.1	Security Challenges in Cloud Computing	15
2.3	Multi-clouds and Multi-cloud Applications	16
2.3.1	Benefits of Multi-cloud Computing	18
2.3.2	Security Challenges in Multi-cloud Computing	20
2.4	Transparency and Security Awareness in Multi-cloud Environments	21
2.5	Summary of the State of the Art	22
3.	METHODOLOGY	24
3.1	Framework Operations	24
3.1.1	Threat Identification and Risk Analysis	24
3.1.2	Selection of Metrics and Security Controls	25
3.1.3	Application Modelling	25
3.1.4	Application Security Monitoring	27
3.1.5	Security Measurement	28
3.1.6	Decision making and Security Status Visualization	28
3.2	Framework Architectural View	29
3.3	Framework Components	29
3.3.1	Request Handling Engine (RHE)	29
3.3.2	Security Policy Engine (SPE)	31
3.3.3	Metrics Monitoring Engine (MME)	32
3.3.4	Security Measurement Engine (SME)	33
3.3.5	Decision & Analytics Engine (DAE)	35
4.	IMPLEMENTATION	38
4.1	Threat Identification and Risk Analysis	39
4.1.1	Threat Identification	39
4.1.2	Risk Analysis	40
4.2	Selection of Security Metrics and Security Controls	42
4.2.1	Selection of Security Metrics	42

4.2.2	Selection of Security Controls	44
4.3	Application Modelling	46
4.3.1	Classes.....	46
4.3.2	Properties	48
5.	RESULTS	50
6.	DISCUSSIONS	55
7.	CONCLUSION.....	57
	REFERENCES.....	59

LIST OF FIGURES

<i>Figure 1. Enterprise cloud strategy</i>	3
<i>Figure 2. The components of a smart environment</i>	6
<i>Figure 3. The lifecycle of self-healing behaviour in OSAD model</i>	11
<i>Figure 4. MARKS architecture</i>	11
<i>Figure 5. Self-healing unit architecture</i>	12
<i>Figure 6. Security adaptation process</i>	14
<i>Figure 7. Cloud computing service models</i>	15
<i>Figure 8. Security evaluation framework methodology</i>	24
<i>Figure 9. Entity-relationship diagram</i>	27
<i>Figure 10. Security evaluation framework architectural view</i>	29
<i>Figure 11. Request Handling Engine</i>	30
<i>Figure 12. Sample request sent by Requestor to SPE</i>	31
<i>Figure 13. Security Policy Engine</i>	32
<i>Figure 14. Metrics Monitoring Engine</i>	33
<i>Figure 15. Security Measurement Engine</i>	34
<i>Figure 16. Decision & Analytics Engine</i>	36
<i>Figure 17. Final security evaluation framework</i>	37
<i>Figure 18. Interactions between engines following a request for security evaluation</i>	37
<i>Figure 19. TSM Application architecture</i>	38
<i>Figure 20. TSM Application risk assessment</i>	41
<i>Figure 21. TSM Application security objectives</i>	42
<i>Figure 22. Ontology of the 'deletion of data' threat</i>	46
<i>Figure 23. TSM Application model properties</i>	48
<i>Figure 24. TSM Application model (showing classes)</i>	51
<i>Figure 25. TSM Application model (showing properties)</i>	52
<i>Figure 26. TSM Application model (showing instances of the CSP class)</i>	53
<i>Figure 27. A sample query/response from the TSM application model</i>	54

LIST OF TABLES

<i>Table 1. Application decomposition for the TSM application</i>	<i>39</i>
<i>Table 2. Identified threats for TSM components</i>	<i>40</i>
<i>Table 3. Selected security metrics for TSM application according to identified threats</i>	<i>43</i>
<i>Table 4. Selected security controls for TSM application threats</i>	<i>44</i>
<i>Table 5. Details of all classes in the TSM application model</i>	<i>47</i>
<i>Table 6. Details of all properties in the TSM application model</i>	<i>48</i>
<i>Table 7. A summary of classes, properties and instances in the TSM application model</i>	<i>49</i>

LIST OF SYMBOLS AND ABBREVIATIONS

6LowPAN	IPv6 over Low-Power Wireless Personal Area Networks
AIS	Application & Interface Security
API	Application Programming Interface
AWS	Amazon Web Service
CCM	Cloud Control Matrix
CEC	Consumption Estimator Calculator
CSA	Cloud Security Alliance
CSP	Cloud Service Provider
CPS	Cyber-Physical Systems
CVSS	Common Vulnerability Scoring System
DAE	Decision & Analytics Engine
DB	Database
DFD	Data Flow Diagram
DoS	Denial of Service
DDoS	Distributed Denial of Service
DREAD	Damage, Reproducibility, Exploitability, Affected users & Discoverability
EKM	Encryption & Key Management
GUI	Graphical User Interface
HiSPO	Hardware, intelligence, Software, Policies and Operation
HSTS	HTTP Strict Transport Security
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IAM	Identity & Access Management
IBM	The International Business Machines
IaaS	Infrastructure-as-a-Service
IP	Internet Protocol
IT	Information Technology
ITU-T	The ITU Telecommunication Standardization Sector
IVS	Infrastructure & Virtualization Security Network Architecture

KB	Knowledge Base
MARKS	Middleware Adaptability for Resource Discovery, Knowledge Usability and Self-healing
MME	Metrics Monitoring Engine
MJP	Multi-modal Journey Planner
NIST	National Institute of Standards and Technology
NVD	National Vulnerability Database
ORB	Object Request Broker
OS	Operating System
OSAD	On-demand Service Assembly and Delivery
OWASP	The Open Web Application Security Project
OWL	Ontology Web Language
PaaS	Platform-as-a-Service
RAM	Random Access Memory
RHE	Request Handling Engine
SaaS	Software-as-a-Service
SDL	Security Development Lifecycle
SLA	Service Level Agreement
SME	Security Measurement Engine
SPARQL	SPARQL Protocol and RDF Query Language
SPE	Security Policy Engine
SQL	Structured Query Language
STRIDE	Spoofing, Tampering, Repudiation, Information disclosure, Denial of service and Elevation of privileges
TLS	Transport Layer Security
TSM	Tampere Smart Mobility
TSM_e	TSM Engine
VM	Virtual Machine
WSN	Wireless Sensor Networks
XSS	Cross-site Scripting

1. INTRODUCTION

This chapter presents the thesis background, the motivation, the problem statement, the objectives and research questions, the limitation and the thesis structure.

1.1 Background

Globalization has brought humans closer to one another in relation to how they relate, interact and carry out their business activities. Nowadays, it has even extended to the point of bringing humans and objects (i.e., devices and systems) closer to one another. However, in order to ensure seamless interaction between humans and objects, the need to embed some form of intelligence into these objects came up and led to the creation and development of smart objects or devices. According to [1], a smart object is a physical thing that has a sensor, an actuator, a low power radio and a microcontroller. Smart objects are designed and developed with the capability of being autonomous, self-aware, self-sustaining, energy efficient and self-governing [2]. These qualities have made it possible to create an ambience for human-object interaction and relationship.

The embedding of microcontrollers in smart devices has given these devices capabilities to interact intelligently with other devices as well as the environment where they are deployed [2]. The sensor perceives the environment; the microcontroller processes the measured data, makes logical decision and provides appropriate response, which is effected by the actuator. Smart objects have the capabilities of being context aware i.e. conscious about their status, location and surrounding environment. They can administer control, respond adequately to changes within their environment and they are capable of learning their environment [3]. The ambience where the interaction between humans and these devices occurs is called a smart environment. The concept of smart environments has been in existence for some time and it resulted from the need for automation, security and energy efficiency i.e., reduction of energy, maintenance and operational costs.

The progresses recorded in supporting fields like sensor networks, robotics, artificial intelligence, mobile and pervasive computing have brought about further work and research in smart environments [3]. According to [4], a smart environment refers to an environment, which acquires and applies knowledge pertaining to the environment and its occupants so as to enhance their feeling within the environment. Smart environments have physical (sensors and actuators), communication, information and decision engines. Typical applications are found in smart buildings, smart factories, smart phones, smart mobility systems, smart homes and Cyber-Physical Systems (CPS), which embodies computational, communication and physical processes e.g. smart grid. Communication is an

integral aspect of smart environments as the different components and devices in the environment interact with one another through the exchange of data and information.

The advances recorded in the field of communication engineering led to the development of communication mechanisms like Wi-Fi, ZigBee, Bluetooth low energy [5], and communication standards like 6LowPAN [6], [7] and this has positively affected the interaction between humans and devices. Furthermore, the emergence of cloud computing has had a great effect on resource utilization, data storage and communication [8]. This is because with cloud computing, there is no need to increase or have a large storage space on local servers and host devices as there are remote servers, which handles and manages storage and processing of information. This has played a huge role in the deployment of smart devices because devices can be developed with a high processing power with little focus on storage and computational resources as these can be outsourced in the cloud thereby requiring the devices to pull the resources from different cloud platforms such as Google cloud, Amazon and Microsoft Azure [9].

Several resources such as email services, web servers, weather forecasts, database servers and search engines now run in the cloud. This means that smart devices have to connect to the cloud in order to utilize these resources. An example is when a user tries to use his mobile phone to access his information stored in Google drive. These resources are available as-a-service in the cloud. The goal of ensuring resource sharing or resource pooling gave rise to the concept of cloud computing. The National Institute of Standards and Technology (NIST) [10] defines cloud computing as a platform based on resource sharing, where resources e.g., applications are consumed with negligible Cloud Service Provider (CSP) participation. Common CSPs are IBM Softlayer, Microsoft Azure, Google cloud and AWS [11].

Nowadays, many enterprises utilize cloud resources for several purposes in order to fulfill their business needs. Notably is the use of multi-clouds, where resources from different cloud providers are combined. For example, an enterprise can deploy a web server on Microsoft Azure while the database server may be hosted on Google Cloud. This can be attributed to associated benefits such as cost optimization, improved quality of service, prevention of vendor lock-in, and increased flexibility through availability of choice just to mention a few. The adoption of multi-clouds has also brought about the development of multi-cloud applications i.e., applications whose components are distributed across multiple clouds. The goal of utilizing multi-clouds and developing multi-cloud applications is to maximize or leverage on the unique capabilities of different CSPs by selecting the best mix of cloud deployments that helps to satisfy diverse customer needs as well as application requirements.

According to the RightScale 2018 state of the cloud report [12], which was based on a survey carried out among 997 respondent organizations on their level of cloud adoption, it was discovered that 81% of the enterprises reported a multi-cloud strategy as shown in

Figure 1. In addition, the average amount of CSPs utilized by these enterprises was about five different clouds. This clearly shows that the use of multiple cloud resources has been enormously embraced by different enterprises for diverse business operations. However, even with numerous benefits of multi-cloud deployment and high level of adoption, the security of the multi-cloud environment remains a major challenge for many enterprises [12]. This is because the level of threats and vulnerabilities increases with increasing use of cloud resources and this poses some security concerns relating to the integrity, confidentiality and availability of the data stored or processed within the environment. As smart devices consume services provided by multi-cloud applications hosted on multiple clouds, they become exposed to different threats and vulnerabilities.

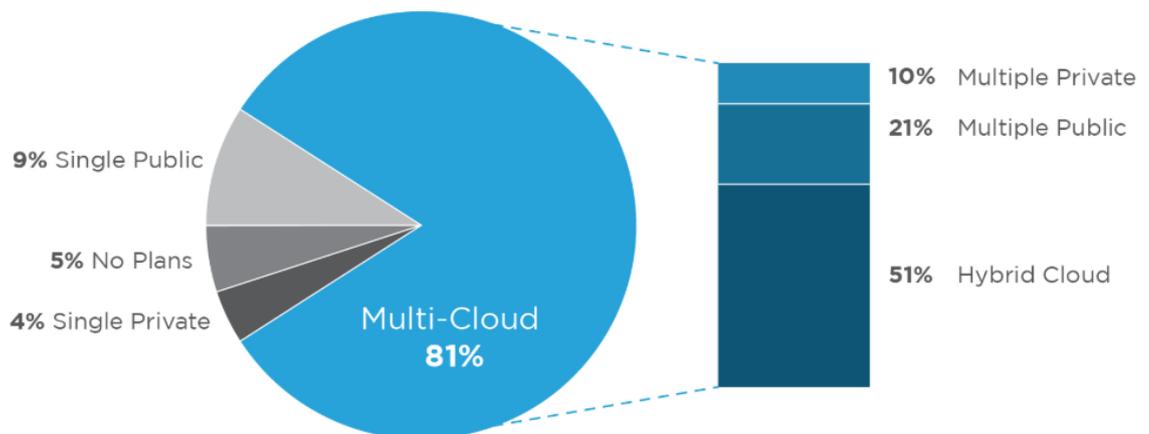


Figure 1. Enterprise cloud strategy [12]

1.2 Motivation

The internet plays a key role in cloud computing as interactions within the cloud and multi-cloud environment thrives on the availability of the internet. This has made it seamless for data and information exchange between diverse components in the multi-cloud environment. However, the open nature of the internet exposes devices, applications and multi-cloud components to various kinds of threats as malicious individuals may take advantage of it. In addition, the multi-cloud environment is heterogeneous in nature owing to diverse components spread across different clouds. This heterogeneous nature results in multiple attack surfaces and further exposes the environment to more threats. The open nature of the internet coupled with the heterogeneous nature of multi-clouds makes the multi-cloud environment complex.

The complexity of the multi-cloud environment involves a high level of interactions between environment components. It introduces threats and makes it difficult to ascertain what is happening within the environment particularly as it relates to the internal events and security status of the application components i.e., how the application components are performing and when remedial actions are needed to be taken in the case of an anomaly. Threat detection and determination of application components' performance require

multi-cloud environment visibility (i.e., transparency) and security state-awareness of multi-cloud application components and the entire multi-cloud environment. This will bring about the detection and mitigation of threats, which may impact the multi-cloud environment. The need for transparency and security awareness in multi-cloud environment has necessitated research in the field. This will surely be beneficial to multi-cloud application and infrastructure owners, as it will ease the burden of security management in multi-clouds environments.

1.3 Problem Statement

There is a growing need amongst cloud customers to become fully aware of performance, health and security status of applications hosted across different clouds. In particular, customers are interested in being able to ascertain that CSPs are truly fulfilling the agreed Service Level Agreement (SLA) as it concerns application security. In addition, multi-cloud adoption is also increasing and so is the level of complexity of the multi-cloud environment. Therefore, there is need to ensure adequate environment transparency and security awareness in order to address this need. However, ensuring transparency and security awareness in multi-cloud environment can be challenging owing to non-standardization, diverse CSP platforms with different modes of operation and the dynamic nature of the multi-cloud environment. In view of this, a multi-faceted but yet holistic security approach is required.

1.4 Objectives and Research Questions

The realization of transparency and security awareness in multi-cloud environments will increase cloud consumers' trust in using multi-clouds. It will also enable the verification of the security of multi-cloud components at any time required. The goal of this thesis is to demonstrate how transparency and security awareness can be achieved in multi-cloud environments. The hypothesis is that through adequate security evaluation, transparency and security awareness can be achieved in multi-cloud environments. This thesis thus proposes a framework that can be used to carry out security evaluation in multi-cloud applications. The security evaluation framework consists of different building blocks called engines, which provide several functionalities that include application modelling, security metrics monitoring, security measurement, decision making and security status visualization.

The proposed security evaluation framework will bring about detection of threats, application and environment monitoring and threat mitigation. This will help to achieve the desired level of transparency and security awareness within the multi-cloud environment and enable cloud customers and infrastructure owners to get adequate information about the health, performance and security status of their application. In summary, the research problem for the thesis can be formulated with the following bullet points;

- How to evaluate security for multi-cloud environment and get this information to the end-user of the application at the device?
- How to implement transparency of multi-cloud systems in order to get awareness on the resources used for the application and their contribution to the security of the overall system?

1.5 Limitation

The scope of this thesis shall be within the domain of multiple clouds services i.e., the interaction and communication between multi-clouds offering different services and resources. The focus shall not be on the mode of deployment like public, private or hybrid clouds.

1.6 Thesis Structure

This thesis is structured as follows; Chapter 2 introduces the thesis theoretical background while Chapter 3 presents the methodology detailing the proposed framework for security evaluation. In Chapter 4, the framework is validated using the TSM application as case study. Chapter 5 shows the results, Chapter 6, the discussion and finally in Chapter 7, the conclusions and future work is presented.

2. THEORETICAL BACKGROUND

This chapter presents the review of existing literature and technology in smart environments, cloud computing and cloud systems, and multi-clouds. In addition, a review of different sub-areas of the main literatures, such as threat modelling, self-healing and self-recovery is also presented. This chapter is divided into the following sections: Smart environments, Cloud computing and cloud systems, Multi-clouds and multi-cloud applications, Transparency and security awareness in multi-cloud environments and Summary of the state of the art.

2.1 Smart Environments

Smart environments acquire and apply knowledge about themselves and their occupants in order to enhance the feelings of their occupants within the environment. They possess sensing and reasoning attributes, which enhances knowledge acquisition and application. The acquisition and application of knowledge is based on data, which helps to generate information about the environment, which is used for decision making. Smart environments consist of different components, namely physical (sensors and actuators), communication, information and decision engines. They collaborate to make sensing and decision making possible. Figure 2 shows the basic architecture of the components and layers of a smart environment.

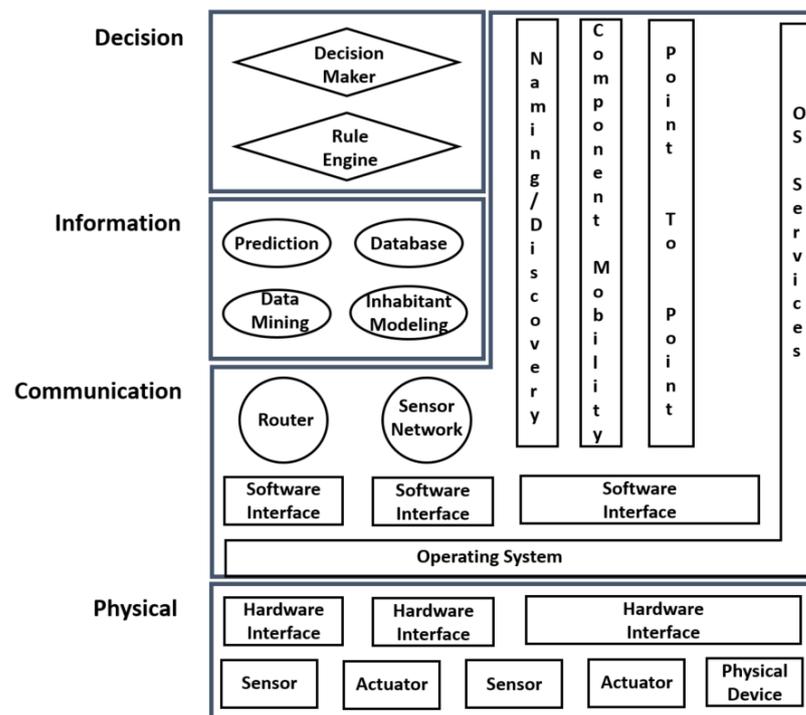


Figure 2. The components of a smart environment [2]

There are several technologies utilized across the different layers of a smart environment. In the physical layer, sensor technology is employed for the detection and acquisition of sensory data. In addition, Wireless Sensor Networks (WSN¹) enhances the sharing of the data acquired between different sensors to make data available for processing and decision making. In the communication layer, wireless communication protocols such as Infrared², Bluetooth and Wi-Fi [5] are used to transmit data between different components within the environment. In the information layer, dedicated prediction algorithms and data mining techniques are used to process, analyze and interpret the data to make meaningful information that will form the basis for decision making in the decision layer. The desired response is provided through device actuation utilizing technologies such as power line control [13]. Other supporting technologies utilized in smart environments include speech recognition, pattern recognition, adaptive control etc.

Typical practical applications of smart environments include adaptive homes, smart rooms, smart offices, assistive environments for the elderly and individual with special needs, smart homes, smart buildings and smart cities. It has become very common to also to refer to these applications as smart systems [14]. With the advent of internet of things, cyber-physical systems and advances in technologies associated with smart systems, many devices are now even more connected than before [15] and huge amount of data is transferred and exchanged seamlessly between these devices. This exchange even extends to the collection and use of personal data of users in the environment and it has raised several security concerns for the users particularly as it affects their trust and confidence. This has called for the need for transparency, security awareness and integration of adequate security measures to meet these challenges.

The main security challenges in smart environments include device integrity, communication channel security [16] and unauthorized access and user privacy [17]. As it concerns device integrity, challenges arise owing to device mobility between different smart environments as devices may become exposed to threats in one environment, and may introduce such threats to another environment. On the issue of privacy, data owners are mostly concerned about the use of their data owing to lack of trust and absence of knowledge of how their data is being used by different entities within the environment. This therefore requires that personal data be adequately protected from unauthorized use and disclosure. To achieve privacy, environment transparency and user awareness is required as data owners need to know how their data is being used, as it will form the basis upon which the consent will be given by the data owner to process and disclose his data. Lastly, on communication channel security, many devices in smart environments communicate using wireless protocols such as Bluetooth, ZigBee, Wi-Fi [5] and Infrared. However, as outlined in [16], these protocols are vulnerable and susceptible to different attacks such as eavesdropping due to their open nature, clear visibility and easy detection by other

¹ <http://www.ni.com/white-paper/7142/en/>

² <https://www.elprocus.com/communication-using-infrared-technology/>

entities that may be around the environment. These entities may introduce threats into the environment if they have malicious intentions. Therefore, adequate security of these communication channels is required to mitigate any possible threats.

In view of the aforementioned challenges, security thus becomes a necessity in smart environments, as it is important to provide adequate security to ensure data protection, user privacy, device integrity and protection of communication channels. This will bring about an increase in user trust and confidence within the environment. The procedure for achieving this involves implementing a security approach that involves threat identification and enforcement of effective countermeasures to mitigate identified threats. Precisely, it involves threat modelling, effective management of device security and the incorporation of self-healing capabilities in smart environments.

2.1.1 Threat Modelling in Smart Environments

Threat modelling is a methodology for carrying out the security analysis of a system in order to discover the threats and malicious events that are likely to affect the system. It helps to ascertain areas and system components where mitigation actions should be applied in order to maintain system security. Several studies have been carried out in modelling threats in smart environments and different methodologies have been proposed. A survey of some of them is provided in the next paragraphs.

Malik et al. [18] propose a 7-step approach for modelling threats in pervasive environments. Preliminary work begins with the definition of users and their roles to ensure authorized access. The main approach involves identification of system domain, identification of trust levels, identification of threats, quantification and estimation of risk based on financial implications of threat, and specification and selection of the most cost effective countermeasures. The approach concludes with the detection of new threats and vulnerabilities earlier unidentified in the system by using a tool such as Common Vulnerability Scoring System (CVSS). This approach is iterative and expansive in nature as any emerging threat can be easily discovered and appropriate countermeasures can be applied.

Martins et al. [19] propose a threat modelling methodology for cyber-physical systems. In this methodology, threat modelling is performed based on component-to-component interactions (i.e. data exchange and communication) by using Data Flow Diagrams (DFD) to model the interactions. It is followed by the identification of threats for each of the interactions using the STRIDE (*Spoofing, Tampering, Repudiation, Information disclosure, Denial of Service and Elevation of privileges*) methodology [20] to model the threats for each interaction. The authors recommended that controls and countermeasures from prominent industrial standards such as NIST “SP 800-82 Rev. 2” [21] should be applied to mitigate the threats. This methodology further widens the threat identification process by identifying threats at the communication layer.

Wang et al. [22] presents a methodology for performing threat modeling in smart city systems from both technical and business perspective (i.e. software, hardware, policies and business operations). The approach named Hardware, intelligence, Software, Policies and Operation (HiSPO) further extends the process of threat modelling by identifying threats on the network, host, application, security policy and operational security levels. The STRIDE methodology is used to categorize the threats and is followed by a risk assessment process. Furthermore, with the aid of a specialized algorithm, the threat factor is computed and it represents the security level of the system. Mitigation strategies are also provided based on the threat factor. This threat modeling process is iterative and is aimed at reducing the threat factor as much as possible.

Beckers et al. [23] presents an approach for a systematic threat analysis in a smart metering gateway system. It includes scope definition, asset identification, domain knowledge consideration, description of attackers, identification of threats and general documentation. The domain knowledge contains information (facts and assumptions) about the possible protection of the assets and it is used to generate the textual documentation. This documentation represents the threat model, which gives the security analysis of the smart metering gateway system. This approach is also iterative in nature as each step is continuously checked in order to accommodate any vital activity not incorporated earlier on. It also provides a detail description of the attacker's abilities.

The approach presented in [23] was extended in [24] with the inclusion of an additional step i.e., identification of entry points and vulnerability analysis. The approach was then applied to identify security threats in smart homes. The approach also relies on the use of Microsoft security development lifecycle (SDL) and context-pattern for scenario description. Based on the assets identified, the possible entry points through which an attacker can access the assets are identified and the STRIDE methodology is used to specify the threats for each entry points. Following this, the possible actions of the attacker on the assets are analyzed based on the entry points and STRIDE threats, using an attack path DFD. The path DFD diagrams help to show how the attacker can harm the asset.

The different studies reviewed in the foregoing paragraphs highlight different techniques and methodologies for modelling threats in smart environments. The methodologies have some steps in common which are asset identification, threat identification and provision of countermeasures. It therefore means that identifying the assets in a smart environment is very relevant for the identification of threats and provision of countermeasures. It follows also that efficient mechanisms are integrated to make the environment detect threats and possible attacks. In essence, the environment should be equipped with capabilities to discover, mitigate and recover from possible attacks.

2.1.2 Self-healing and Self-recovery in Smart Environments

According to Ghosh et al. [25], Self-healing features makes it possible for a system to know when it is not in correct operating condition and performs necessary alteration to regularize itself through reconfiguration and fault recovery. In more details, it is the potential of a system to detect an anomaly in its mode of operation, examine the anomaly and carry out actions to repair the anomaly and restore itself back to normal working condition without the need for any human intervention. This means that such a system is ably equipped with functionalities to identify what an abnormal working operation is, prevent and react to such anomaly and apply the set of rules governing the repair and recovery from such faults in the system in real-time. The main attributes of self-healing systems are detecting, diagnosing and recovering [26] i.e. automatic discovery of faults and correction of faults, errors or anomalies.

The attributes of self-healing systems also form its building blocks and extend beyond automatic discovery and correction of faults. Other attributes such as analysis and planning are also part of the self-healing process. The self-healing process follows the approach of autonomic computing system defined by IBM in [27]. It includes monitoring (sensing), analyzing (prediction or estimation), planning (selection) and executing (recovery) together with a knowledge base, which serves as a repository of rules and policies for applying and enacting the four processes. Some studies have been conducted in the area of self-healing systems. However, at the state of the art, there is no particular universally accepted scope, standard, architecture or model for representing the building blocks of a self-healing system [28]. Different architectures and frameworks have been presented and proposed but a common occurrence is that they are all a spin-off of the four main processes of autonomic computing systems.

Pereira et al. [28] presents a self-healing middleware used to develop distributed applications based on On-demand Service Assembly and Delivery (OSAD) model. Middleware aids interoperability in a distributed environment. The implementation of self-healing based on the OSAD model utilizes component redundancy. It consists of four main blocks/processes, namely application monitoring, failure detection, alternative component discovery and replacement of failed component. It starts with continuous monitoring of applications running on virtual containers in order to identify the application properties, relationships and inter-dependencies. At the failure of any of the application components, a replacement is immediately searched using the lookup and discovery service and the appropriate alternative component is invoked and the failed component replaced in order to ensure normalcy. The model is shown in Figure 3.

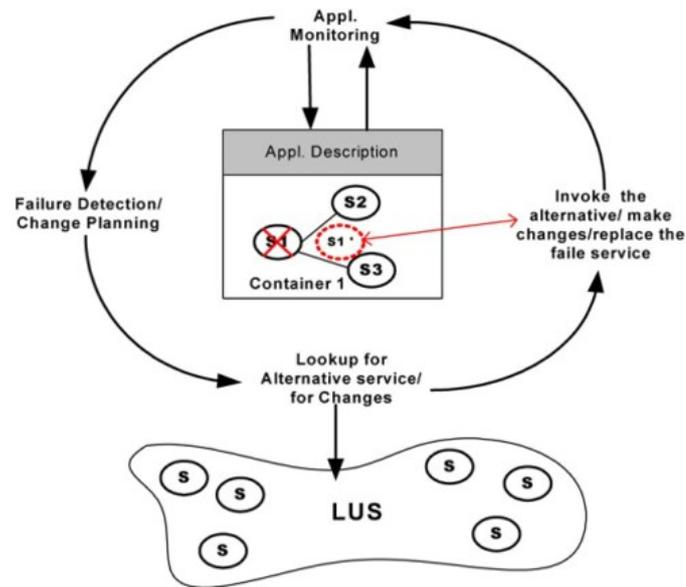


Figure 3. The lifecycle of self-healing behaviour in OSAD model [28]

Sharmin et al. [29] also presents a middleware called Middleware Adaptability for Resource Discovery, Knowledge Usability and Self-healing (MARKS), that is well suited for pervasive computing environment. It consists of components (object request broker (ORB), universal service access unit, trust management, and resource discovery) and services (self-healing, knowledge usability and context-service). Figure 4 shows the MARKS architecture.

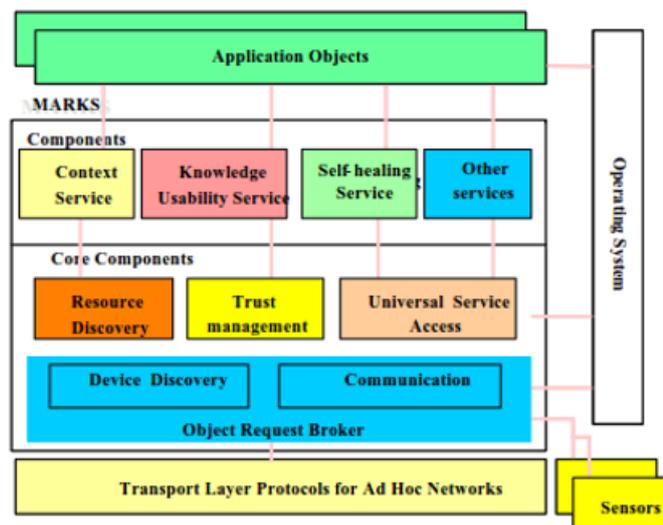


Figure 4. MARKS architecture [29]

The Self-healing unit of MARKS consists of the healing manager and resource manager, and the unit's process involves fault detection, fault notification and resource recovery. It continuously monitors the devices by querying and analyzing the 'rate of change of status' messages generated from the devices. This could be 'OK' or 'SOS'. If the device returns no message or 'SOS' message, it translates to a fault. The unit then notifies the

healing manager to initiate recovery procedure, which begins with isolating the faulty device and searching for an alternative. At the same time, the information on the faulty device is shared among active devices to service restoration and continuity. The self-healing architecture is shown in Figure 5.

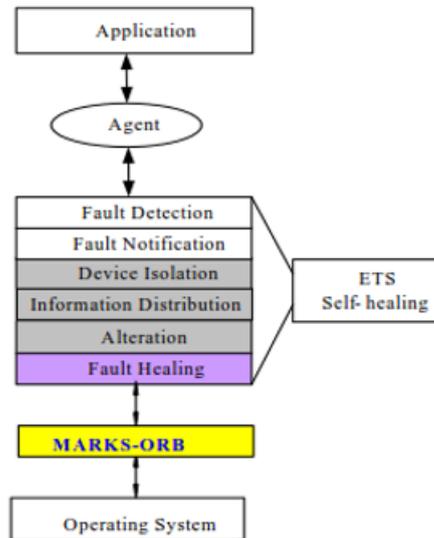


Figure 5. Self-healing unit architecture [29]

The studies reviewed in the foregoing paragraphs show different approaches or methods for achieving self-healing in smart environments. It revealed that the basic functional requirements for bringing about self-healing in smart environments are monitoring, detection, notification and recovery & restoration. These functionalities have to be enshrined in the operational setup of smart environments to ensure anomaly detection and continuous operation of the environment. In addition to self-healing, it is also important to have measures in place to manage device security in a bid enhance proactivity regarding security management.

2.1.3 Managing Device Security in Smart Environments

In smart environments, the application and enforcement of security occurs at different layers; one of such is the device layer. Smart environments accommodate several devices, some of which are very mobile such as the ones carried by humans who regularly make use of the environment. Owing to this mobility, these devices may be transported and used in other smart environments and as a result, they become vulnerable and exposed to threats that may be available in the environment. In the event of a device being compromised by threats in one environment, it becomes capable of introducing such threats when taken into another environment. As such, it becomes necessary therefore to ensure that the security of devices is effectively managed in order to protect them against any form of threats or attacks as well as the possible introduction of such threats into the smart

environment. This includes the enforcement of policies and application of relevant security mechanisms. The next paragraphs present some relevant works.

McAvoy et al. [30] propose the use of ontology for context-management of smart environments, which also includes ontological modelling of sensor data. It utilizes semantic descriptions for enabling automatic identification of sensors when added to the environment. The proposed context-management system consists of five components, namely device, sensor ontology, enrichment, semantic repository, Application Programming Interface (API) query & retrieval. Sensor ontology component enables the addition of new devices based on the data it receives from them and the enrichment component contains a reasoning engine, which processes the data before sending it to the semantic repository component. Applications can then make use of the processed data through API queries. In addition, the authors implemented a new "plug-n-measure" procedure that ensures acquisition of data and update of contextual data to the semantic repository.

This approach presented in [30] ensures that new devices can be efficiently added to smart environments with less amount of time and effort. However, it fails to address the issue of security of the added device. An improvement on this approach could be the inclusion of device security configuration. In addition, information about the required level of security that has to be maintained in the environment by all the devices could be specified in the semantic repository. This information is made available as security metrics. Therefore, when a new device is added, the sensor ontology component receives data from the device including its security information and then sends it to the enrichment component for comparison with the security metrics stored in the semantic repository component. If the required metrics is met, the device is immediately added but if not, a rejection is made immediately.

Evesti and Ovaska [31] suggest an ontology-based security adaptation at run-time for the security management of smart spaces. The adaptation process involves two phases namely, start-up phase and run-time phase. The start-up phase is initiated when a new device joins the smart space. At the introduction of the new device, the security requirements, security levels and the security mechanisms that support the requirements for the operation of the smart space are retrieved from the security ontology. The applicable security mechanisms are selected and the security level of the device is measured. If the measured levels do not meet the security requirements specified in the security ontology, the run-time adaptation is executed. The initiation of the run-time phase adaptation implies non-conformance and detection of threat within the environment and when this occurs, the environment adjusts by applying the appropriate security mechanism needed to mitigate the threat. The security adaptation process is illustrated in Figure 6.

The above studies have highlighted the importance and effectiveness of ontology for security management as seen in its use for the modelling and specification of security requirements, security metrics, security levels, security mechanisms and other security parameters.

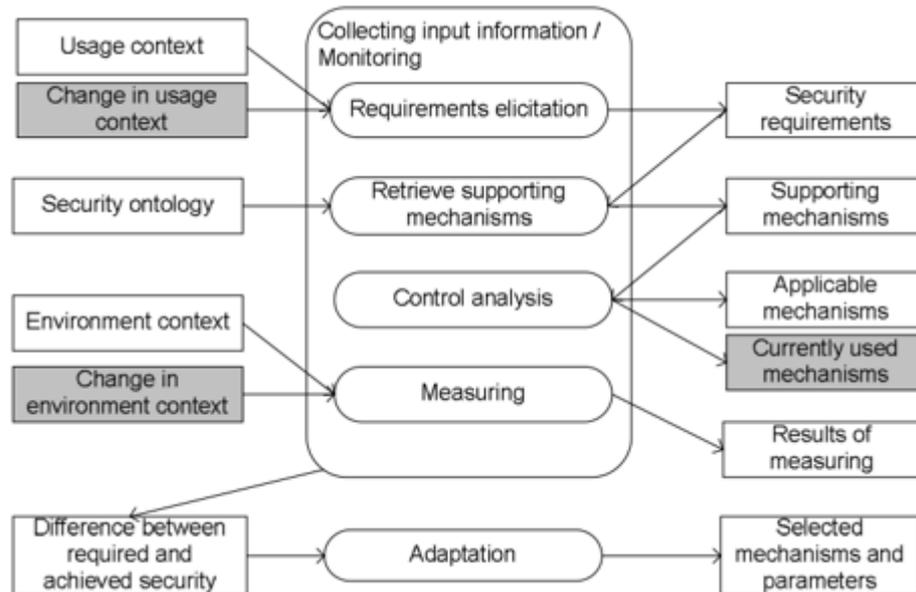


Figure 6. Security adaptation process [31]

2.2 Cloud Computing and Cloud Systems

Cloud computing involves making computing services available to users at the desired time, location and quantity [32]. The associated cost depends on the amount of resources consumed and this is of great benefit to individuals and establishments, as they do not have to spend heavily on maintaining IT infrastructures. In simple terms, cloud computing is based on a pay-as-you-consume model and it allows establishments to focus on their core expertise. According to NIST [33], there are three fundamental cloud computing service models, namely Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS) and Software-as-a-Service (SaaS) as shown in Figure 7. Nowadays, new services are emerging such as network-as-a-service specified by ITU-T and data-as-a-service as defined in ISO/IEC 17826:2016 [34]. This has been largely due to the benefits of cloud computing.

However, even with the cloud computing benefits, the issue of security still affects its usage by enterprises. As presented in [10], security is a major cloud challenge, particularly relating to privacy risks and data loss. This is because privacy and data security are of utmost concern for many cloud users. Therefore, identifying the security challenges in cloud computing becomes very important. This can be achieved by evaluating cloud computing bearing in mind an attacker's intention, as it will help to reveal his most likely

actions. Identifying and analyzing security challenges will facilitate countermeasure determination. This will enable the proper selection of techniques, tools and mechanisms needed to mitigate the security challenges and through this; enterprises will have confidence using cloud resources.

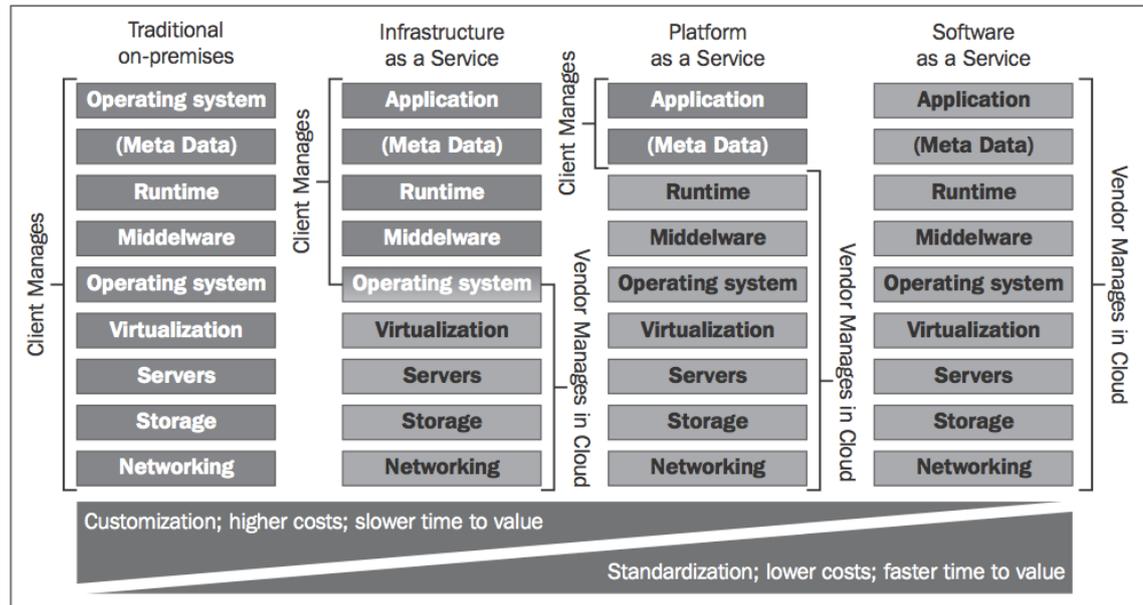


Figure 7. Cloud computing service models [35]

2.2.1 Security Challenges in Cloud Computing

Security is concerned with the provision and enforcement of safeguards to protect a resource of great value. Traditionally and as it applies to information security, the basic goal is to ensure confidentiality, integrity and availability. In cloud computing, the provision of security extends beyond these basic objectives as it becomes vital to manage and control access to data stored in the cloud, incorporate measures for attack resilience and detection of threats etc. This becomes necessary as the risk of exposure grows with increasing numbers of cloud consumers as well as the discovery of new threats. Several cloud security challenges have been identified through different studies and different mitigation techniques and mechanisms have also been proposed to address the challenges. A review of relevant studies for identifying cloud security challenges is presented in the next paragraphs.

The Open Web Application Security Project (OWASP) [36] presents an approach for detecting and combating threats in the development of software applications. The approach offers a procedure to identify, rate and tackle security risks in an application. It involves three steps i.e., the decomposition of application, discovery and rating of threats, and discovery of control measures. OWASP classifies the application threats as STRIDE. The foundation also proposes the use of encryption, hashing, authentication, the protection of secret data, use of privacy-enhanced protocols, authorization and use of digital

signatures as countermeasures to minimize identified risks. The approach presented by OWASP is generic but may be extended and used in cloud computing domain.

Another approach is presented in [37] where the issue of cloud security was studied from four aspects namely, cloud service delivery, cloud architecture, cloud stakeholders and cloud offered characteristics. The authors mention that cloud security issues are ingrained in isolation, access control, virtualization, security management and multi-tenancy. Following their analysis of cloud security issues, they recommended that the solutions to cloud computing issues should be adaptive, support integration with other security controls and use models, which ensures that users can only access their own security configurations.

Johnson E. Robert [38] states that the rate of consumption of cloud resources surpasses the rate of cloud security tools development. The author mentions that the use of cloud resources has brought about new security challenges that can be addressed by modifying conventional security controls. He identifies security threats like Structured Query Language (SQL) injection, Buffer Overflow and Cross Site Scripting. He mentions that privacy and auditing issues have become paramount in cloud computing. He proposes remote integrity monitoring, encryption and virtual private storage proxy as methods of preventing illegal access, increasing and augmenting the security in the cloud.

According to Akhil Behl [39], cloud computing benefits also attracts various threats. The author mentions that security responsibility should be handled by both CSP and cloud customer. He identifies loss of control, insider threats, data loss, multi-tenancy, service disruptions and outsider malicious attacks as cloud security challenges. He also proposes the use of strong authentication, firewalls, authorization, intrusion detection systems, data integrity mechanisms and properly defined SLA as countermeasures to mitigate the identified security challenges.

Security breaches will have an effect on many cloud customers. Its prevention and control involve identifying cloud security challenges and developing security mechanisms and tools for tackling the challenges. The studies reviewed in the previous paragraphs have brought about the identification of the major cloud computing security challenges as well as the methods and mechanisms required to counter them. These mechanisms can be developed into security tools that can be used with cloud applications to address security issues.

2.3 Multi-clouds and Multi-cloud Applications

Multi-clouds refer to the utilization of multiple cloud services by an enterprise to meet a business objective. This may involve using different CSPs for the same service model, for different cloud service models or for distributed applications. A simple example might

be an enterprise using multiple IaaS providers to host a particular application, an enterprise that uses different CSPs to meet its IaaS, PaaS and SaaS needs or an enterprise that decides to split an application into various components and host the components using different CSPs. Another example could be a combination of a personal data center (private cloud) and a public cloud provider. This may be necessary when an enterprise needs to have some form of control over the type of data or information they outsource to CSPs for storage.

Multi-cloud application on the other hand refers to an application, which is broken down into various components, with the components distributed across different clouds [40]. Multi-cloud applications follow the concept of distributed computing where components are dispersed but made to communicate and interact in a consolidated or coordinated fashion in order to achieve a desired goal. These components may be services, containers or micro-services. The entire process occurs as if it were just a single application. Multi-cloud computing allows multi-cloud applications to adopt the use of different cloud services (IaaS, PaaS and SaaS) from different CSPs. This means that the distributed components can be deployed in the separate clouds for their operations as CSPs can be selected according to the components and application requirements.

The development of a multi-cloud application involves several stages, which involves design, development, deployment and run-time [41]. The multi-cloud application creation begins with the design of the application. In the design stage, the application is modelled by specifying the application architecture (i.e., the application components, interactions and mode of communication) and the specification of the requirements (i.e., hardware, cloud resources, location, operating system etc.). This will form the basis for the development of the multi-cloud application SLA. The next stage is development and it involves building the multi-cloud application components as specified in the design stage using relevant software and technologies e.g. Java, JavaScript.

The process continues with the deployment of the multi-cloud application in cloud environment. It involves installation, configuration, testing and provisioning of cloud resources e.g. servers, virtual machines. In this stage, the services and CSPs that satisfy the application requirements are specified. This results in the creation of the deployment script, which contains all information needed for deployment execution. Common tools used for deployment are Chef, Puppet and Ansible [42], [43]. The last stage is run-time. In this stage, the multi-cloud application runs on different cloud infrastructures where it has been deployed. The application is monitored to observe its performance and detect any anomaly or violation. In the event of an anomaly, notification is sent to trigger and enforce appropriate remedial actions i.e., countermeasure or security control.

Multi-cloud deployments present an approach for maximizing the benefits and potentials of different cloud providers. It offers the opportunity to have the best mix of CSP offerings that best satisfies business objectives and application requirements. Through this,

key factors such as performance and cost can be optimized. The benefits of multi-cloud computing are presented in the next section.

2.3.1 Benefits of Multi-cloud Computing

There are several reasons why an enterprise may decide to use multi-clouds for its operation. These reasons are entrenched in the benefits of multi-clouds, which include the prevention of vendor lock-in, reduction of cost, flexibility of choice of CSPs, disaster recovery, service redundancy, cost optimization, load balancing, improvement of quality of service etc. [44], [45]. In order to fully enjoy the benefits of multi-clouds, it is important that enterprises clearly understand their needs and how to make the appropriate selections to fulfill them. This begins with identifying business objectives (e.g., operational and financial) and application requirements (e.g., functional and technical) and then determining and selecting the appropriate cloud service model(s) and CSPs that satisfies the requirements. The service mix and CSP selection may require the sound knowledge and expertise of in-house IT personnel. The benefits of multi-clouds are discussed in the next paragraphs.

Multi-clouds help in the prevention of vendor lock-in [44]. This clearly points to the prevention of over-dependence on one cloud provider. In this setup, two or more CSPs are used and cloud consumers can leave one cloud provider to another at any time when they feel the need to do so owing to cost, unsatisfactory service, unavailability of service or the discovery of better service offering by a different cloud provider. It offers them flexibility of choice that provides them with the opportunity to transit to the CSP that best suits them. The main issue to address here is interoperability between CSPs as it has a great influence on the smooth transition of data between different cloud providers.

Multi-cloud deployment enables cloud consumers to control their risks by distributing it across different clouds [45]. This is evident in the case of data availability, disaster recovery and fail-over services. For example, if several instances of a particular service are deployed in about three different public clouds, it is possible to ensure the continuous provisioning and running of the service even in the event of the primary cloud being shut down or unavailable as the service continues to run in the other clouds. This is a simple scenario of fail-over. Generally, this can be achieved using clusters [46], [47]. Just as in the case of data availability, the main data is stored in a primary cloud and its replicas created and stored in two other different clouds (secondary clouds) provided by a different CSP. Continuous availability is ensured when the data is provided by secondary clouds when the primary cloud fails. With this setup, data availability may be guaranteed whenever the data owner needs to access it.

Multi-clouds make it possible to take advantage of the strengths of various cloud providers [44]. The strengths of CSPs are seen as their core competence, and are entrenched in the types of cloud models, and cloud services that they offer. e.g., *CSP 'A' may be a better*

provider of IaaS than CSP 'B' probably because CSP 'B' focuses more on providing PaaS. In the provision of these services, variations such as performance, costs and coverage exist, and which are what cloud users capitalize on and use as metrics for measuring and determining the suitability of CSPs for their operations. This makes it possible therefore for cloud consumers to take advantage of these differences to get the best of cloud service offerings by different CSPs. This will even provide cloud consumers with the opportunity to get the best mix of solutions and services offered by different CSPs according to how best they satisfy their needs. Through this, the best CSP satisfying a particular need can be selected and enterprises can reduce their costs and achieve a good balance in the consumption of cloud services.

In the aspect of data location and in particular where data governance applies in the location of data, the use of multiple clouds may also be beneficial. For instance, in a situation where user data (relating to the use of a particular application) has to be stored within certain locations owing to the requirements of data protection laws, it becomes very important to ensure that the data resides within that location. However, in the event that the desired CSP does not have a data center within the location, it becomes a problem fulfilling data governance (data storage) requirements. The use of multi-clouds helps to address this problem as user data can be stored in the facilities of a CSP within the region while the other application components and underlying services are hosted in the other desired CSP. With this arrangement, effective communication between the different CSPs must be adequately addressed particularly in the areas of coordination and security of communication channels.

Another reason for the utilization of multi-clouds is the need for confidentiality and reduction of latency. For instance, even with the previously highlighted benefits of multi-clouds, certain enterprises might still decide not to put certain data or information in public clouds owing to data privacy. For this, they might decide to keep very sensitive data and information on-premise (i.e., in their private cloud facility) while they outsource less sensitive information to desired CSPs. This gives the enterprise great control over their data. As it concerns latency reduction, multi-clouds may be used to achieve this particularly when an enterprise' customers are domiciled in different regions. Latency reduction is achieved by bringing the service closer to the customers through the use of different CSPs located within customers' regions. This will bring about faster response time and customer satisfaction.

In the foregoing paragraphs, the benefits of multi-clouds have been highlighted and discussed. These benefits have been the reason for its growing adoption. However, it is important to mention that despite its increasing adoption, security is still a concern as stated in [12]. This is associated with the increasing level of risks owing to the distributed nature of the multi-cloud environment. This increasing level of risks amounts to increasing threats and vulnerabilities, which contributes to the security challenges in multi-clouds.

It is important to properly identify these security challenges as well as the countermeasures needed to tackle them. Through this, the multi-cloud environment becomes more secure and consumers' confidence in using multi-clouds increases. In this next section, the main security challenges in multi-clouds are discussed.

2.3.2 Security Challenges in Multi-cloud Computing

The implementation of multi-clouds requires the enforcement of multi-protection [48]. This is largely attributed to the dynamism and complexity of the multi-cloud environment, which involves the use of different interfaces and endpoints to establish interactions and communication between different environment components. Increased number of interfaces and endpoints contribute to increased risks and vulnerabilities as the attack surfaces become multiplied and can be exploited by adversaries to perpetuate different attacks and malicious activities. Hence the need for multiple protection to secure the multiple attack surfaces. As a way of addressing this need, different attempts have been made towards identifying the security challenges in multi-clouds. This will serve as a great input for providing the needed level of protection in the multi-cloud environment. The security challenges that have been identified through different studies are presented in the next paragraphs.

As outlined in [49], the main security challenges in multi-clouds are establishing trust among CSPs, data privacy, loss of control over data and policy heterogeneity. On the issue of trust, the challenge arises as many cloud customers delegate total security of their assets (e.g., applications) to their CSPs. This exposes the assets to risks such as insider threats, which could even multiply as data is transferred between CSPs and other third-party organizations. This clearly raises trust concerns as cloud customers lose control and visibility, and become unaware of who might be accessing the asset as well as how security is being administered over it. As it concerns data privacy, when sharing and using multiple customer data for research and analytical purposes for instance, it is important that data privacy is ensured so as not to reveal sensitive and personal information of different customers as required by data protection laws. At the same time, measures must also be taken to ensure data usability. Lastly on policy heterogeneity, different CSPs have unique security policies, which mostly leads to conflicts and breaches during integration owing to non-standardization. It is important to ensure policy harmonization through adequate standardization to detect conflicts and resolve policy inconsistencies.

To address the aforementioned challenges, the authors proposed the use of proxies as intermediaries for enabling effective cooperation between applications distributed across multiple clouds. They stated that the use of proxies will help to achieve trust between cloud customers and CSPs. It will also help to tackle policy heterogeneity through detection and resolution of policy anomalies. The authors also proposed the use of data perturbation (addition of noise to data) as a means of ensuring data privacy.

In [50], the main multi-cloud security challenge highlighted is the inappropriate access to personal data and VMs, which may result in the breach of data privacy. The need for the protection of private data cannot be over-emphasized as already highlighted in previous paragraphs. To address this challenge, the authors proposed a model-driven architecture to ensure security and grant users their personal spaces within the multi-cloud environment. The PaaSage multi-cloud platform [50] was used as a case study. The proposed solution tackles inappropriate access to personal data and VMs through the enforcement of adequate authentication and authorization when consuming services by the PaaSage platform. It allocates dedicated information spaces to organization and their users and provides APIs that allows each organization's administrators to securely manage their information space within the platform (i.e., management of security policies, users, roles, permissions etc.).

In the foregoing paragraphs, multi-cloud security challenges have been identified following the study of some research works. The studies show that the main security challenges in multi-clouds are data privacy, trust, loss of control over data and policy heterogeneity. However, beyond the identified challenges, the need for visibility (transparency) and security awareness in multi-cloud environments was also highlighted. This will ensure that application owners will have a clear knowledge of activities and events taking place within the multi-cloud environment and will be able to react accordingly to achieve the desired level of security in the environment. In the next section, the transparency and security awareness in multi-clouds are presented.

2.4 Transparency and Security Awareness in Multi-cloud Environments

As illustrated in the introductory chapter, the complexity of the multi-cloud environment and the growing need of cloud customers to become more aware of the security of their application has mandated the need for transparency and security awareness. It has become necessary to provide a means through which application owners and cloud customers can ascertain the occurrences and security situations of their assets (e.g., applications, data, information) hosted or stored across multiple clouds. Specifically, cloud customers want to be sure that CSPs are respecting agreed SLA especially regarding application security. Transparency refers to a state of visibility, clarity or openness. It ensures that nothing is hidden. Security awareness refers to a condition of being conscious about the state of security of an entity e.g., an asset or a space.

Transparency will ensure that all internal events, activities and interactions between the entities in multi-cloud environments are visible to application owners and cloud customers. Security awareness will ensure that application owners and cloud customers can verify the security state or status of the assets and the entire environment. It involves both

user-awareness and component-awareness. Through transparency and security awareness, multi-cloud application components can be duly monitored, anomalies can be prevented, detected and mitigated; and normalcy restored within the environment. This will be beneficial to multi-cloud users, as it will enhance control over personal assets and ease the burden of security management in multi-clouds environments. However, it requires implementing processes that are adaptable and CSP independent.

Transparency and security awareness requires that efficient mechanisms for specifying multi-cloud application requirements, monitoring multi-cloud application, detecting threats, applying countermeasures, determining and reporting security status should be integrated into the multi-cloud environment. This can be achieved through security evaluation of the multi-cloud environment. This is because evaluating the multi-cloud components and the multi-cloud environment will bring about the determination of the true status of the components and the environment in terms of health, performance and security. This will ensure visibility and knowledge about the environment and its entities. In this thesis, a framework for enabling security evaluation of multi-clouds is proposed.

The framework will facilitate anomaly detection, multi-cloud environment monitoring and application of appropriate countermeasures to mitigate threats. Specifically, it will provide the following functions, namely application modelling, security monitoring, security measurement, decision making and security status visualization. The combination of these functions will bring about a transparent and security-aware multi-cloud environment, which will improve customers' trust and confidence in using multi-clouds.

2.5 Summary of the State of the Art

In the foregoing sub-sections, the most significant topics relating to this thesis have been presented. The technological advancements and the different research conducted in the fields of smart environments, cloud computing and multi-clouds have formed a good basis for developing multi-cloud applications and this has brought about an increase in the level of use of multiple cloud resources, particularly as it concerns choosing the best mix of the resources. However, the multifaceted nature of the multi-cloud environment raises several security concerns, particularly regarding transparency and security awareness, which has made it burdensome to determine the security condition of the environment and the applications it hosts, and has therefore necessitated the need to develop an approach for evaluating security in multi-cloud environments as presented in this thesis.

The proposed security evaluation framework ensures that the security levels of components hosted across different clouds as well as the entire multi-cloud environment can be realized. This addresses the security concerns surrounding transparency and security awareness as cloud customers, application owners and administrators will have full knowledge of the security state and wellbeing of their application and the entire multi-cloud environment. It will further ensure that threats can be identified, mitigated, and

required level of performance of the application and the multi-cloud environment is maintained. Security evaluation in multi-cloud environments will promote visibility, security awareness and upgrade customers' credence, using multiple cloud resources. The procedures for realizing this framework are explained in the next chapter.

3. METHODOLOGY

The proposed security evaluation framework will provide a methodology for evaluating security in multi-cloud applications. It is made up of different components called engines, which are responsible for providing different operations. In this section, the framework operations, architectural view and the components' functionalities are described.

3.1 Framework Operations

The proposed security evaluation framework consists of the following operations, namely threat identification and risk analysis, selection of metrics and security controls, application modelling, application security monitoring, security measurement, decision making and security status visualization. Figure 8 shows a diagrammatic illustration of the security evaluation framework methodology. The description of each operation is also provided in the next sections.

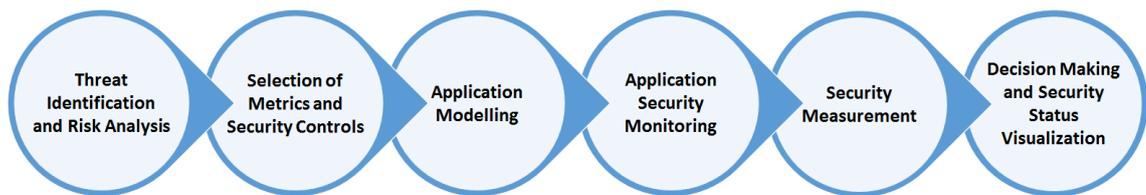


Figure 8. Security evaluation framework methodology

3.1.1 Threat Identification and Risk Analysis

Threat identification and risk analysis are the first operations in the proposed security evaluation framework. Threat identification involves the specification of all factors affecting the security of the application, leading to the determination of threats while risk analysis involves ranking the identified threats to determine their level of criticality. Threat identification depends on the peculiarity of the application (i.e., whether it is a software or storage application) and it is carried out by analyzing the application based on its components and their interactions i.e., identifying threats by specifying the likely actions and factors that may impact the components as well as their mode of interactions. Threat identification is achieved through threat modelling. A review of different threat modelling techniques is presented in [51]. The approach recommended by OWASP [36], which is based on STRIDE methodology [20] will be used in this thesis.

Risk analysis follows the identifications of threats. It is performed on each threat and per application component. It involves evaluating and rating each identified threat in order to determine the potential extent of the damage to the component in the event of an attack. This is very essential as it helps to identify the critical component(s) of the application

that requires adequate protection. Specifically, risk analysis is carried out by estimating the probability of occurrence and impact of each threat using quantitative risk models (where risk is a product of probability and impact) or value-based models such as DREAD (*Damage potential, Reproducibility, Exploitability, Affected users and Discoverability*) [36]. Threat identification and risk analysis are important steps for identifying threats, determining critical components and selecting security controls. The approach presented in [52] illustrates how this can be achieved in multi-cloud applications.

3.1.2 Selection of Metrics and Security Controls

The next operation that follows the identification of threats is the selection of metrics and security controls. Metrics selection involves specifying monitorable parameters while the selection of controls involves specifying countermeasures for threat mitigation. Metrics are quantifiable parameters that can be monitored or measured to get certain information about the application performance, characteristics or behaviours. Metrics are defined and selected according to the desires of the application owner. They are aligned to certain properties or defined security objectives that clearly reflect the application characteristics or behaviours that are of great interest to the application owner. For instance, *Availability* can be a property, with *average response time* as the metric. Metrics also have threshold values, which when violated could result in a threat. The research works in [53]–[55] propose a set of metrics that can be used for monitoring the security level in multi-cloud applications.

The selection of security controls is an essential step required for the mitigation of identified threats. It involves identifying and selecting the countermeasures needed to tackle anomalies and restore the application back to a safe condition. The selection of security controls requires a good understanding of the identified threats. This is because having a good knowledge of the identified threats will aid the selection of security controls that are adequate for mitigating the threats. Security controls can be selected by adopting methodologies formulated by different security control frameworks and standards such as NIST SP 800-53 Rev.4 [56], Cloud Security Alliance “Cloud control matrix” (CCM) [57].

3.1.3 Application Modelling

Application modelling involves specifying the multi-cloud application requirements. These requirements include application component description (name, type and number of replicas), CSP information, geographical location, organization (i.e., application owner) details, threats, security metrics, security controls, violation thresholds, VM requirements (hardware, OS, RAM and number of cores), communication ports and modes of interaction. Application modelling is very important as it provides a means to describe

the application and its requirements thereby enabling proper understanding of the application functionalities and its interactions. Simply put, it ensures application definition and system understanding. Application modelling can be done using modelling languages such as CloudML [58], OWL (Ontology Web Language) [59], UMLSec [60], SecAM [61]. In this thesis, multi-cloud application is modelled with ontology using OWL.

Ontologies express knowledge in a formal manner [62]. It encapsulates knowledge about a specific domain and helps to describe its logical structures. It promotes knowledge sharing and reuse among different entities and allows making informed inferencing, facilitates reasoning and aids quality decision making. Ontologies are made up of concepts (classes), relationships, attributes and instances and have a wide application in software engineering, artificial intelligence and semantic web. Ontologies are suitable for modelling dynamic systems (i.e., systems with constantly changing information). In multi-cloud environments for instance, the use of ontology will ensure that rapidly changing information (entities, events, interactions and security status) in the environment can be adequately captured and updated. This will bring about responsiveness, self-awareness and self-learning in the system.

Ontologies have been applied in modelling information systems security and have resulted in the development of security ontologies as seen in [63]–[68], where ontologies have been applied for risk management, vulnerabilities detection, intrusion detection and presentation of general security overview. Furthermore, in the field of cloud computing, ontologies have been applied for modelling interactions, security measures, factors and controls as presented in [69]–[71]. Security ontologies provide a means for representing security domain knowledge. It begins the definition of security concepts and description of the relationships between them. As it concerns the multi-cloud environment, the concepts are the multi-cloud application requirements, which includes asset or application component, CSP, organization, threats, security metrics, security controls, VM requirements etc., while the relationships are the interactions that exist between the concepts. Figure 9 shows a snippet of the entity-relationship diagram for this thesis.

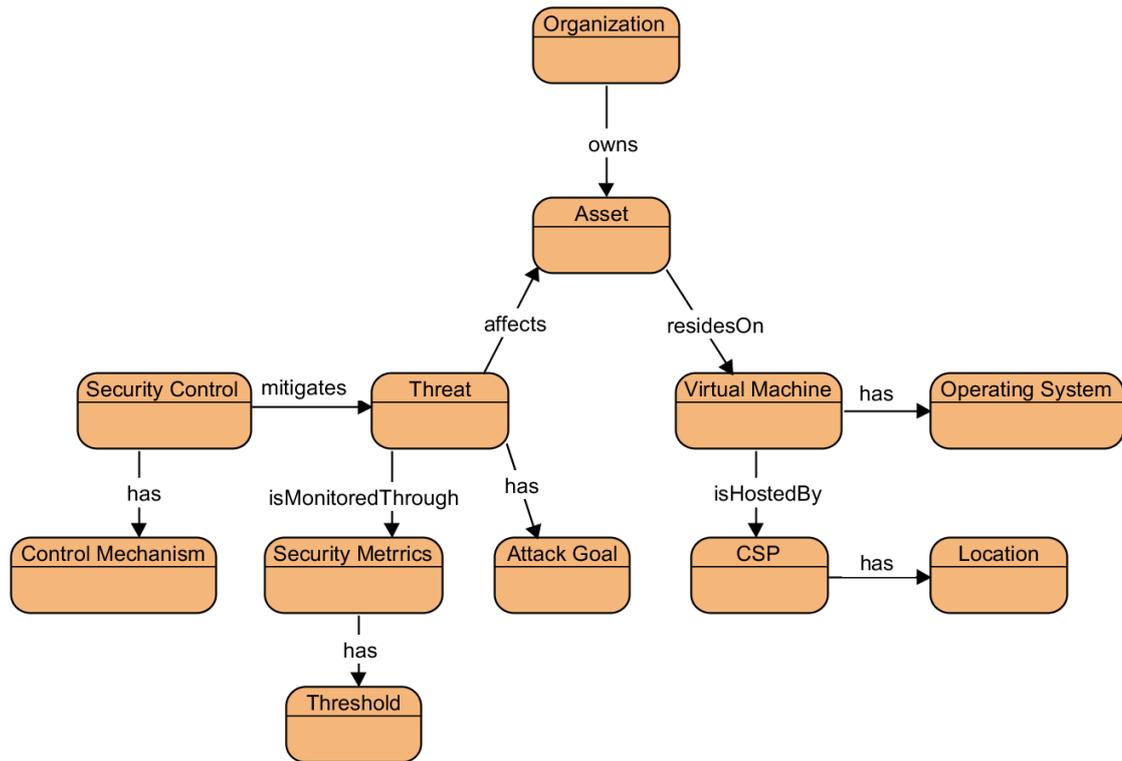


Figure 9. Entity-relationship diagram

The result of the application modelling operation is the multi-cloud application security policy, which facilitates security awareness and offers great support for application security monitoring, security measurement, decision making and security status visualization.

3.1.4 Application Security Monitoring

Application security monitoring refers to the security surveillance of an application to detect any unusual behavior. It involves monitoring a set of metrics that can help to ascertain whether the application is behaving as desired or whether there has been a case of metrics violation that could lead to a threat. Application security monitoring may be achieved through the use of security monitoring probes [40] and relevant security libraries. A security monitoring probe is a software program, which monitors different security variables (i.e., security metrics) on a network, an application or operating system and then transmits the monitored data to a server for further actions. Security monitoring probes are installed in the VMs hosting the application components to monitor different sets of metrics that helps to establish the status of application properties like availability, performance, communication traffic etc. at any time desired. Security libraries may be integrated into the application components to monitor internal events.

Application security monitoring also extends to the management of all security monitoring probes installed across different VMs. This is important in order to achieve overall multi-cloud application security monitoring. The values of the monitored metrics reported

by each monitoring probe are compiled and stored (i.e., the composite metrics), before being sent for further measurement, calculation, evaluation and analysis. The analysis may involve the use of machine learning techniques as mentioned in [48], [72]. Application security monitoring will ensure that security metrics and relevant security events are duly monitored, captured and reported. An approach for monitoring security in multi-cloud applications is presented in [73].

3.1.5 Security Measurement

The security measurement operation involves the computation of the percentage level of fulfillment of each metric (expressed within a range of 0 and 1) based on defined thresholds and the security quotient (i.e., a measure of the level of security) of the multi-cloud application. It makes use of specified rules and mathematical formulae to carry out security measurements and calculations. Specifically, it applies the rules and formulae on the security metrics values reported by the monitoring probes to calculate the security quotient of the multi-cloud application. Security measurement is a combination of security objectives [74]. Thus, the security quotient is calculated in terms of the three basic security objectives (i.e., confidentiality, integrity and availability) for individual multi-cloud application components as well as the overall multi-cloud application based on the request for security evaluation. For instance, the security quotient for “availability” of the multi-cloud application is calculated as the weighted average of all availability security metrics i.e., $\text{Summation of (all availability metrics (in percentage units)) / number of metrics.}$

3.1.6 Decision making and Security Status Visualization

The decision-making and security status visualization process is the last step of the security evaluation framework. Decision making involves the assessment of the security measurements and verdict provision while security status visualization involves displaying the security status of the application. In the decision-making process, the security measurements are evaluated to ascertain if normal operating conditions have been met or if a breach has occurred. Specifically, the measurements are compared with the specified thresholds of the metrics and security quotient to determine conformance (desired conditions) or non-conformance (breach or violation). In the case of a breach, decision is taken to initiate the process for applying the appropriate security control(s) needed to mitigate the threat and restore normalcy.

Following the evaluation of the security measurements, the result of the evaluation is reported. This is called the security status visualization. It involves generating statistics and presenting the security status of the multi-cloud application in a user-friendly and easy-to-understand manner, specifically using graphs, bar charts, pie charts etc. (i.e., GUI). This may be achieved using relevant data visualization JavaScript libraries such as

d3.js³ in conjunction with front-end JavaScript frameworks such as Vue.js⁴. The aim of this is to bring about seamless notification and simple presentation of security evaluation results (for individual multi-cloud application components and entire application) to the end user (application owner, system administrators and IT team).

3.2 Framework Architectural View

The architectural view of the proposed security evaluation framework is presented in Figure 10.

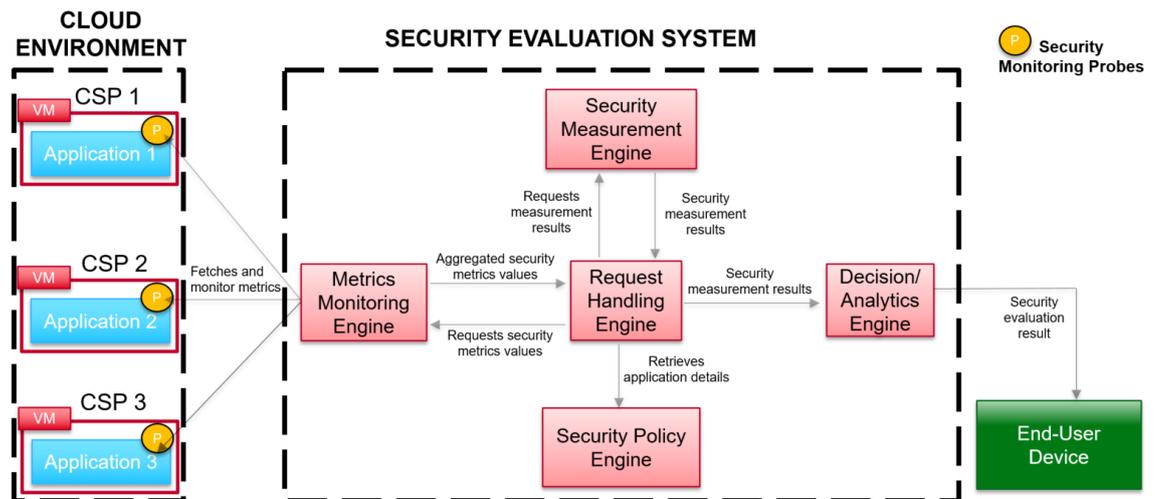


Figure 10. Security evaluation framework architectural view

3.3 Framework Components

The main components of the proposed security evaluation framework are request handling engine, security policy engine, security measurement engine, metrics monitoring engine, and decision & analytics engine. The description of each component is provided in the next sections.

3.3.1 Request Handling Engine (RHE)

The request handling engine is the entry port of the security evaluation framework. It is the coordinator of the security evaluation system and acts an intermediary for communication/interactions between other engines. It receives the request to evaluate the security of the multi-cloud application and according to the request; it makes a call by forwarding the request to the appropriate framework component for processing. Following the processing, the response is sent back to the RHE and the RHE sends it to the engine from where the request came from or to another engine for further actions as the case may be.

³ <https://d3js.org/>

⁴ <https://vuejs.org/>

It continues this process until the final evaluation result has been computed and is now ready to be made available to the end-user.

The RHE (shown in Figure 11) is a software component, which consists of three modules, namely *Receiver*, *Directory* and *Requestor*. The *Receiver* module receives/accepts the request sent by the end-user via the user web-interface or other requests coming from other components and validates the request by checking the *Directory* for correctness. The *Directory* contains a list of mappings for permissible incoming/outgoing requests in the RHE. This makes it possible for the RHE to identify the incoming request, the request initiator, the corresponding outgoing request as well as the component required to process the request. Following this, the corresponding outgoing request is fetched from the *Directory* and sent to the appropriate component by the *Requestor*.

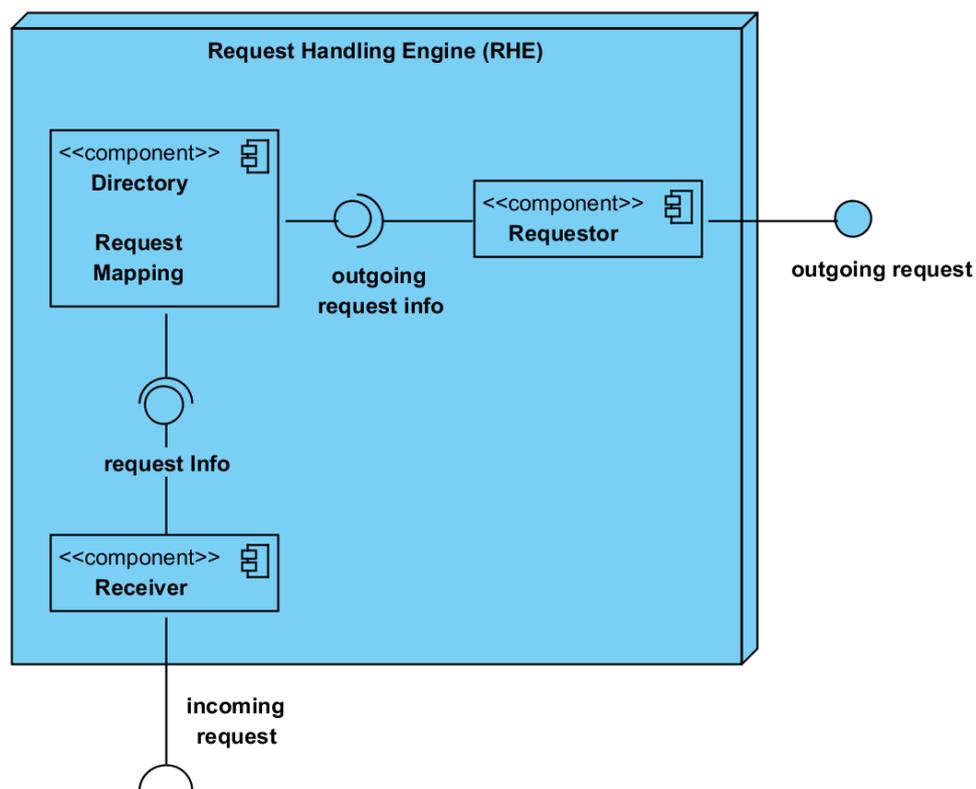
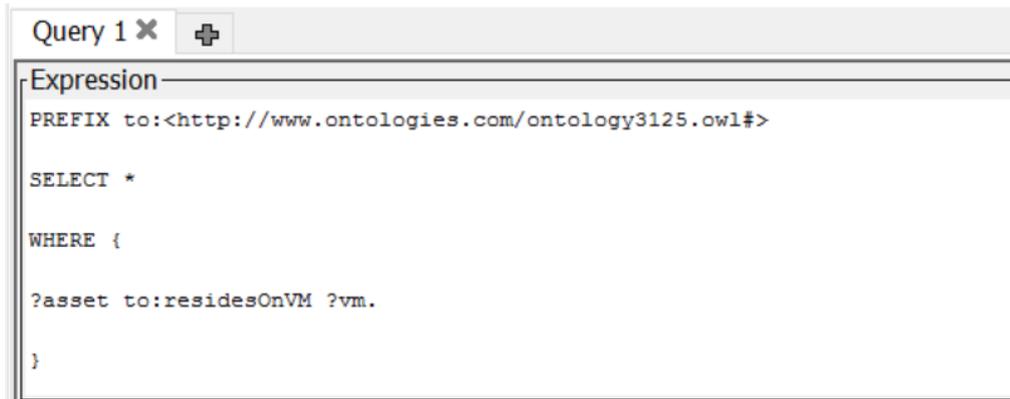


Figure 11. Request Handling Engine

For the purpose of illustration, the request for security evaluation would normally begin with the RHE communicating with the SPE to extract relevant application details pertaining to the request (e.g., component name, VM details etc.). This is achieved by sending SPARQL⁵ queries from the *Requestor* to the SPE to extract the required details from the SPE. The response provided by the SPE is received by the *Receiver* and serves as input for the next action. An example of a typical query sent to the SPE to retrieve application details is shown in Figure 12.

⁵ <https://www.w3.org/TR/sparql11-update/>



```

Query 1 × +
Expression
PREFIX to:<http://www.ontologies.com/ontology3125.owl#>

SELECT *

WHERE {

?asset to:residesOnVM ?vm.

}

```

Figure 12. Sample request sent by Requestor to SPE

In the above query, the Requestor is sending a query to the SPE to request all assets and their VMs.

3.3.2 Security Policy Engine (SPE)

The security policy engine provides a means for modelling and specifying the requirements of the requirements and information about the application as well as information about other engines in the framework. The security policy engine contains information about the multi-cloud application (i.e. detailing each component) and all other engines making up the security evaluation system. Specifically, it contains information about assets (i.e. application components), cloud service providers, infrastructure type, threats, vulnerabilities, organization, metrics, security controls, thresholds, violations etc. The SPE is based on the use of ontologies to model the multi-cloud application properties and requirements, which are stored in the knowledge base.

The SPE (shown in Figure 13) consists of four modules, namely *Inquirer*, *Result*, *Reasoner* and the *Knowledge base (KB)*. The *Inquirer* accepts incoming requests (SPARQL queries) from the RHE and transmits it to the *Reasoner* for further actions. The *Reasoner* module is responsible for querying and inferencing. It queries the *Knowledge Base* (a store of security ontology (multi-cloud application security information)) and performs inferencing to acquire the query answers, which are presented through the *Result* module. Specifically, the reasoner utilizes forward and backward chaining process [75] for making inferences and first-order logic [76] for making deductions based on the facts and axioms (i.e., security ontologies) already defined in the *Knowledge Base*. Through this, the reasoner is able to extract new facts from the ontology.

The flexibility of ontological models makes it easy to update the knowledge base whenever required in order to accommodate new changes. In this thesis, the Olingvo tool, which is based on OWL is used to develop the required security ontology (security requirements, interactions, concepts, relationships etc.), which represents the security policy of the security evaluation system.

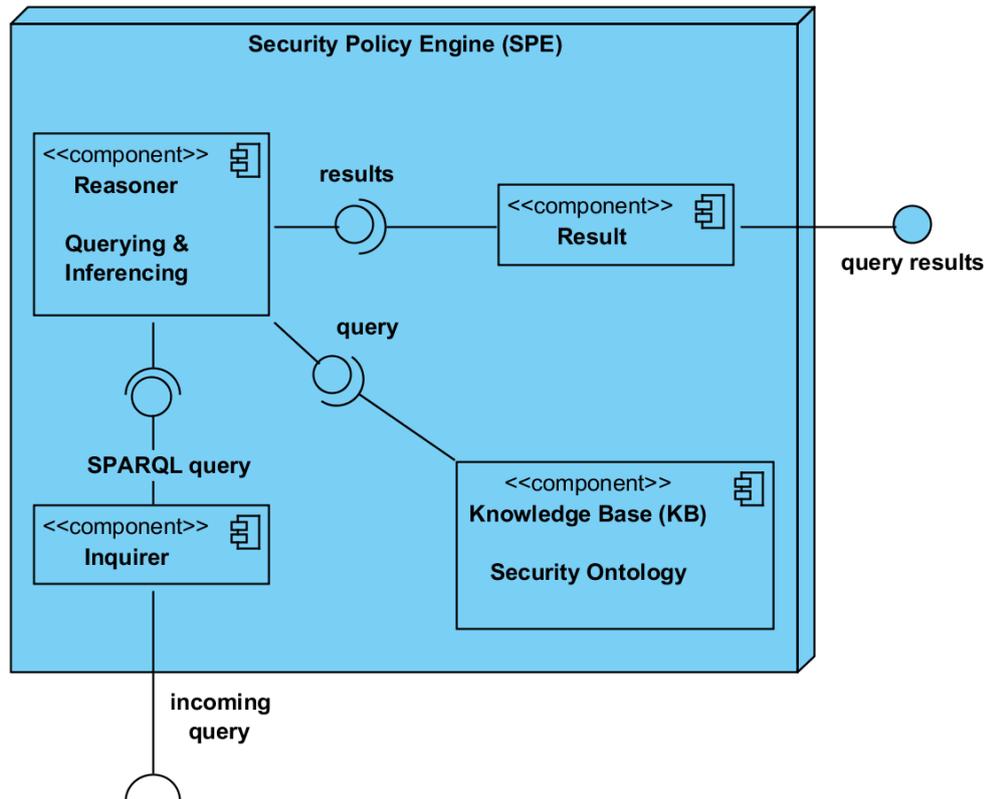


Figure 13. Security Policy Engine

3.3.3 Metrics Monitoring Engine (MME)

The metrics monitoring engine is responsible for the security surveillance management of the multi-cloud application. It performs this task to identify any anomaly in the system. It manages the activities of all the security monitoring probes (monitoring agents) installed on the VMs hosting the multi-cloud application components in the CSP environment. It receives, stores and aggregates the monitored set of metric values reported by the different security monitoring probes. The aggregated metric values are then forwarded via the RHE to the SME for security measurement.

The MME (shown in Figure 14) is a software component, which consists of three modules, namely *monitoring server*, *metrics aggregator* and the *database*. The *monitoring server* manages the operations of all security monitoring probes. It continuously receives the security metric values from the monitoring probes and sends them to the *database* for storage. The metric values together with their time stamps are stored in the database for the purpose of data logging (i.e., auditing). The *metrics aggregator* fetches the metric values from the *database*, aggregates them according to the security evaluation request and sends them through the RHE to the SME for security measurements and calculations.

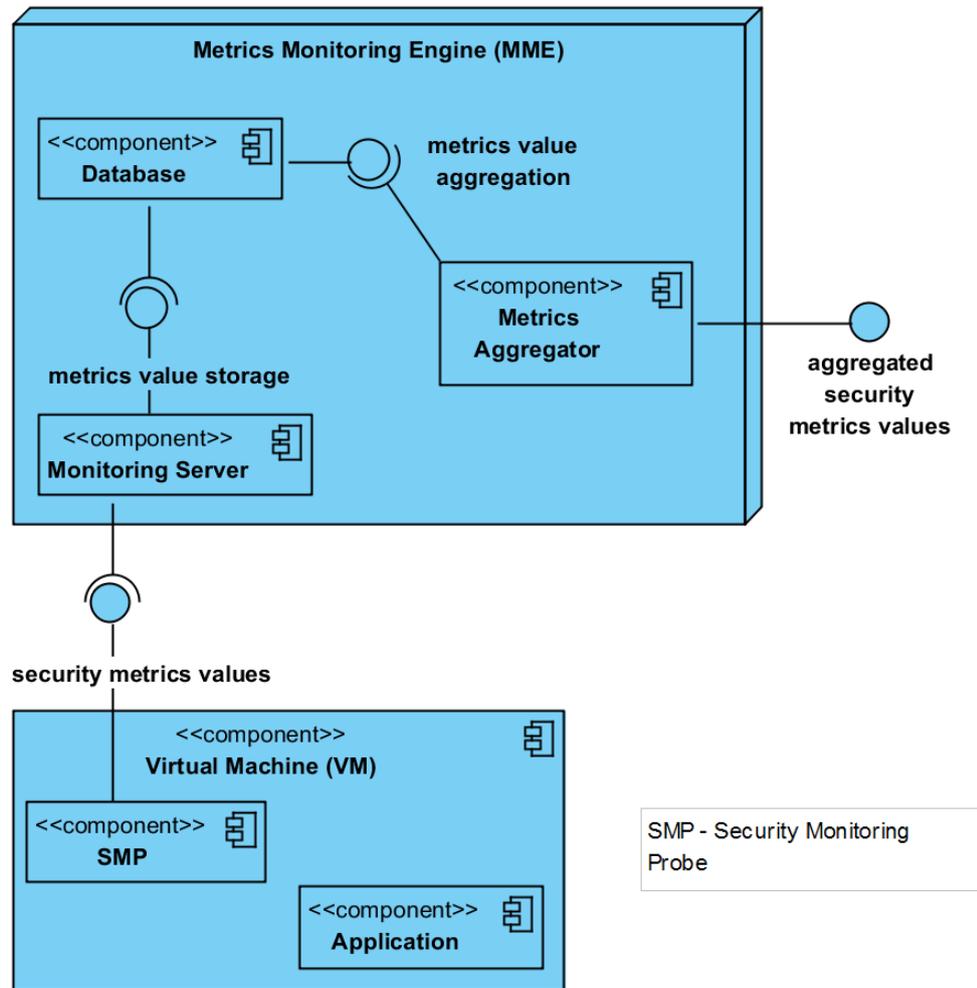


Figure 14. Metrics Monitoring Engine

3.3.4 Security Measurement Engine (SME)

The security measurement engine calculates the percentage level of fulfillment of each metric and the security quotient of the multi-cloud application. Through this, the level of security in the multi-cloud application is determined. The SME carries out these calculations using predefined rules and mathematical formulae that are applied on the metric values. The security measurements are made according to the confidentiality, integrity and availability status of each component of the multi-cloud application. The measurements are also aggregated to derive the overall security measurement of the multi-cloud application. The rules and mathematical formulae are updated through the security policy engine. This allows for easy modification of the formulae whenever changes are necessary.

The SME (shown in Figure 15) is a software component, which consists of three modules, namely *Receiver*, *Security_calculator* and *Result*. The *Receiver* module accepts incoming requests for security measurements. The request includes the metric values reported by

the monitoring probes. The *Receiver* forwards the request to the *Security_calculator*, triggering the call for security measurement. The *Security_calculator* is purely a mathematical component containing several mathematical rules, functions and formulae for calculating the security measurement. According to the security measurement request, it invokes the appropriate mathematical formulae and functions required to calculate the percentage level of metric fulfillment and the security quotient of the security objective(s). The calculation is made for a single component or the entire multi-cloud application component according to the request. Following this calculation, the *Result* module retrieves the security measurement results for onward transmission to the RHE for further actions.

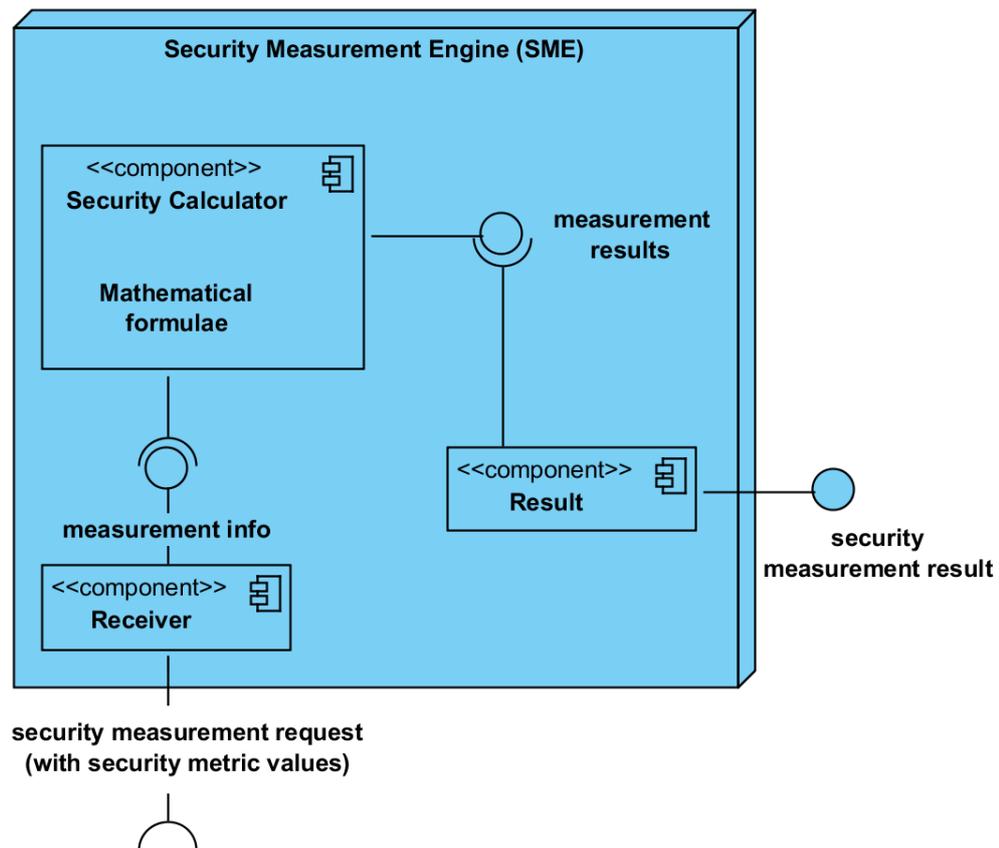


Figure 15. Security Measurement Engine

The mathematical formulae for making the different security measurements are presented below. First, the mathematical notations are defined;

L = percentage level of metrics fulfillment

m = measured metric value

n = total number of metrics

n_a = total number of availability metrics

n_c = total number of confidentiality metrics

n_i = total number of integrity metrics

S = specified metric threshold

SQ_o = Overall security quotient

SQ_a = Security quotient for availability

SQ_c = Security quotient for confidentiality

SQ_i = Security quotient for integrity

Then followed by the actual mathematical formulae;

$$L = \frac{m}{S} \quad (1)$$

$$SQ_a = \frac{\sum_{j=1}^{n_a} \left(\frac{m_j}{S_j} \right)}{n_a} \quad (2)$$

$$SQ_c = \frac{\sum_{j=1}^{n_c} \left(\frac{m_j}{S_j} \right)}{n_c} \quad (3)$$

$$SQ_i = \frac{\sum_{j=1}^{n_i} \left(\frac{m_j}{S_j} \right)}{n_i} \quad (4)$$

$$SQ_o = \frac{\{SQ_a + SQ_c + SQ_i\}}{3} = \frac{1}{3} \left\{ \frac{\sum_{j=1}^{n_a} \left(\frac{m_j}{S_j} \right)}{n_a} + \frac{\sum_{j=1}^{n_c} \left(\frac{m_j}{S_j} \right)}{n_c} + \frac{\sum_{j=1}^{n_i} \left(\frac{m_j}{S_j} \right)}{n_i} \right\} \quad (5)$$

3.3.5 Decision & Analytics Engine (DAE)

The decision & analytics engine handles decision-making and presentation of the security status of the multi-cloud application to the end-user. It receives the results of the security measurement performed by the SME and evaluates it based on predefined threshold values to establish a state of normalcy or violation, after which appropriate actions are taken. It also displays the results of the security evaluation exercise in the end-user device in an easily comprehensible way using graphs, bar charts and pie charts etc. Through this, the end-user becomes aware of the internal operations and performance of the multi-cloud application.

The DAE (shown in Figure 16) is a software component, which consists of three modules, namely *Receptor*, *Decision* and *Visualizer*. The *Receptor* module retrieves the results of the security measurements from the SME and initiates the decision-making process by forwarding the results to the *Decision* module. The *Decision* module is a comparator and

enforcement module. It receives the security measurement results and compares them with the specified thresholds of metrics and security quotient to determine if they are within/beyond the acceptable limits. This helps to estimate the security status of the multi-cloud application. In the event of a violation of the thresholds, the *Decision module* decides on the appropriate security control(s) and initiates the process for enforcing it to mitigate the anomaly and restore normalcy. Following this, the *Visualizer* presents the security status of the multi-cloud application. The *Visualizer* is basically a web interface through which the result of the security evaluation exercise i.e., the security status is presented to the end-user. The *Visualizer* also allows the end-user to interact with the security evaluation framework.

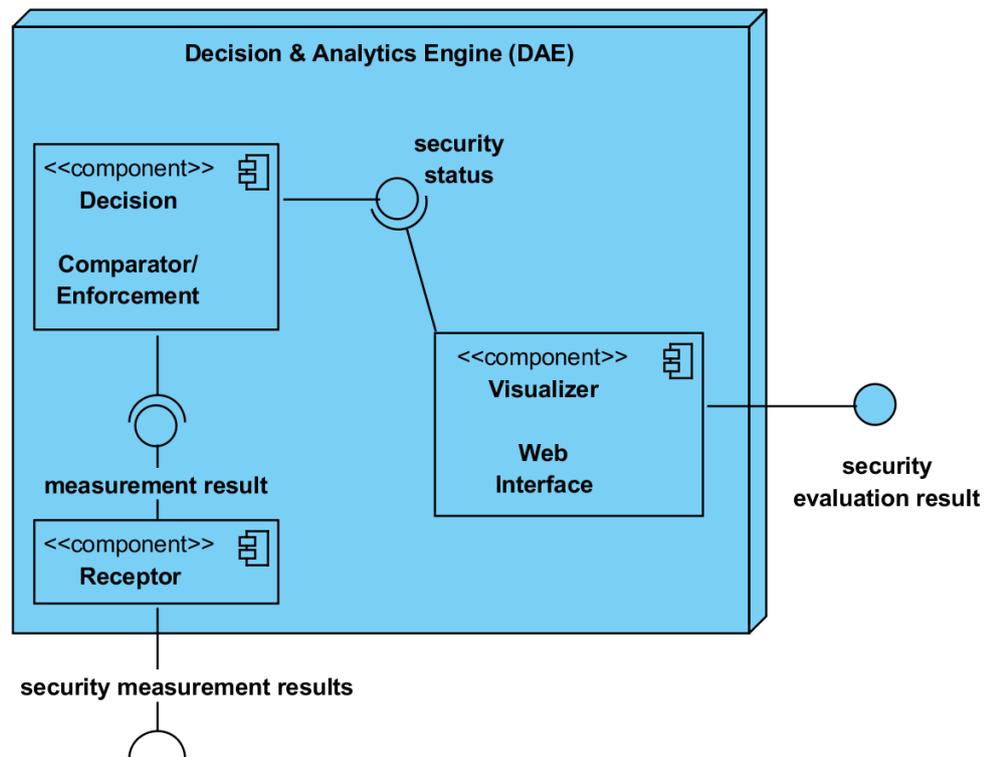


Figure 16. Decision & Analytics Engine

In this section, the security evaluation framework components have been described. The combination of all the framework components results in the final security evaluation framework, which is presented in Figure 17. It receives security evaluation request as input and provides security evaluation results as its output.

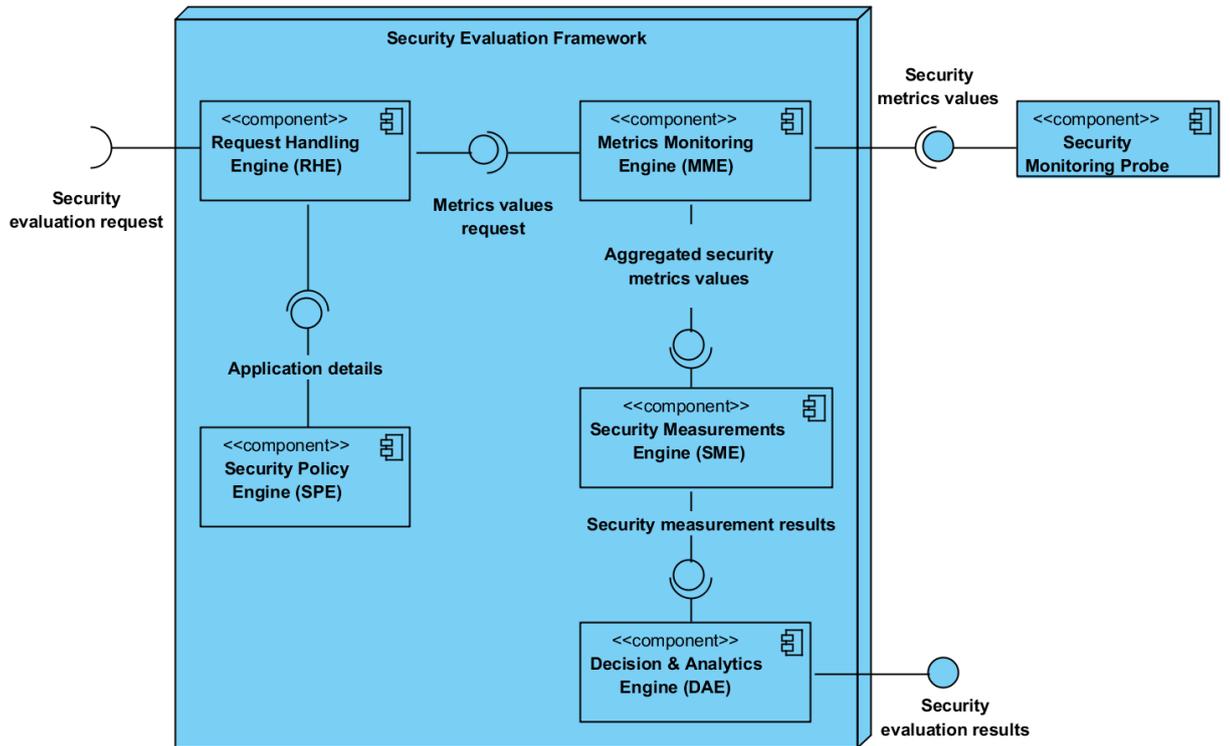


Figure 17. Final security evaluation framework

The security evaluation request is received by the RHE, which interacts with other engines to process the request until the evaluation results are provided by the DAE. The processing of the request involves several interactions between the framework engines. The sequence diagram shown in Figure 18 presents these interactions.

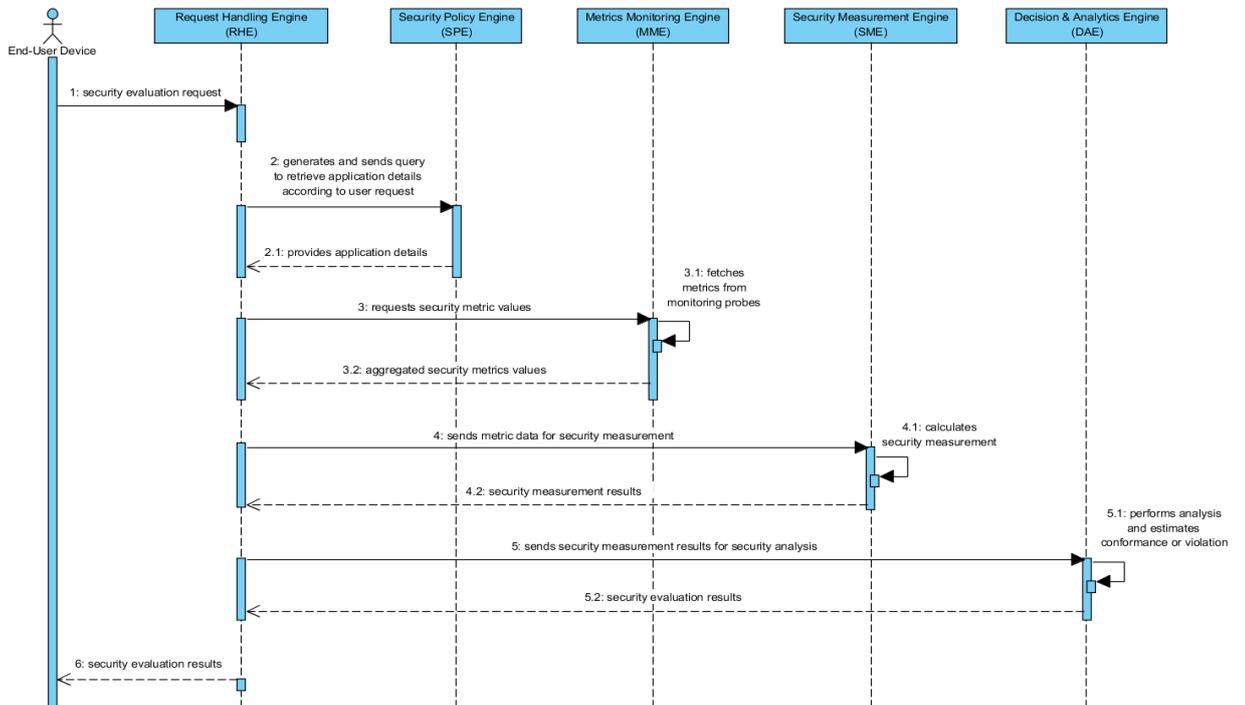


Figure 18. Interactions between engines following a request for security evaluation

4. IMPLEMENTATION

In the previous section, the methodology for performing security evaluation was presented. In this section, the methodology is validated using the TSM application as a case study. The TSM application [52] is “a smart mobility multi-cloud application that provides an energy efficient and smart mobility solution to Tampere (Finland) citizens”. It makes available personalized journey suggestions from which the application users can make their selections. It stores users’ personal data, activity and mobility profiles on the cloud and thus requires adequate protection to prevent breaches and abuses. The architecture of the TSM application is shown in Figure 19.

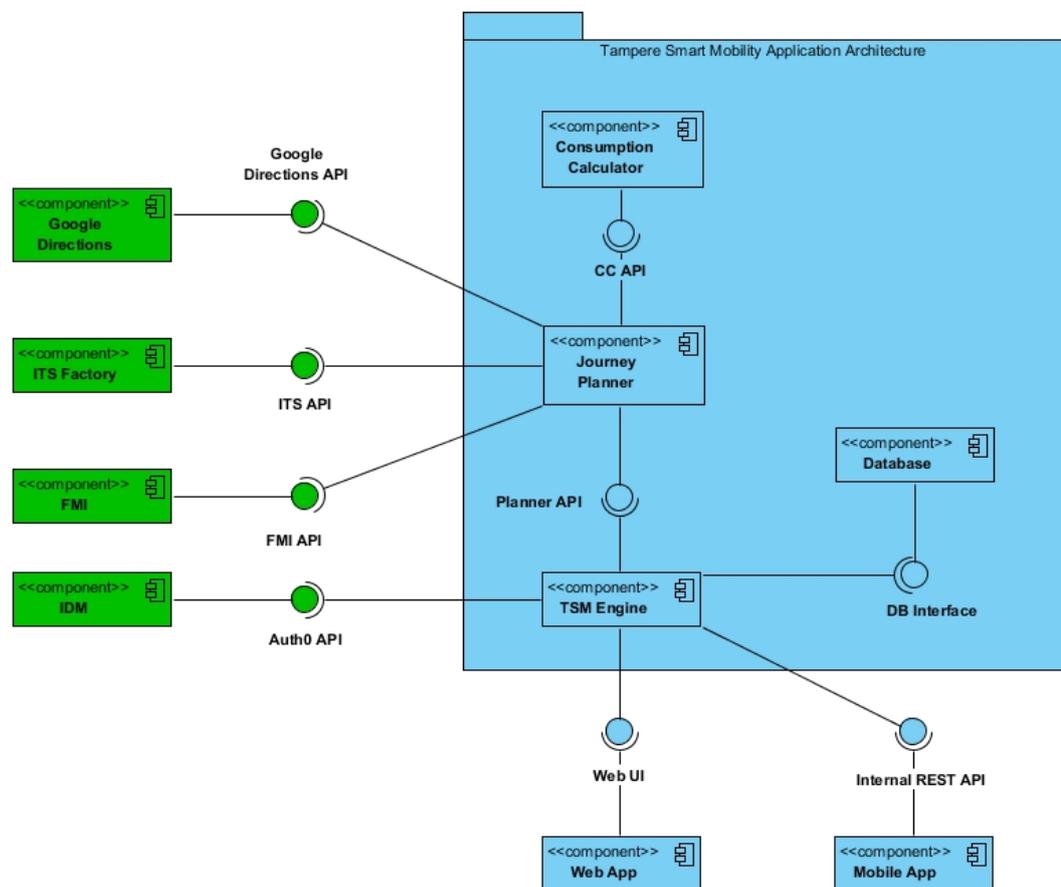


Figure 19. TSM Application architecture

The focus in this implementation will be on threat identification, risk analysis, selection of security metrics, selection of security controls and application modelling.

4.1 Threat Identification and Risk Analysis

Threat identification and risk analysis begins with the decomposition of the TSM application. This involves splitting the application into the constituents that makes up the application. This is essential as it helps to identify the application components and understand their nature, operation and interactions with external entities. Specifically, application decomposition involves identification of assets, entry points and trust levels. The assets represent the application components, which are the attack targets and thus need to be safeguarded. Entry points are the interfaces through which the components communicate with external entities while the trust levels represent the access rights granted to external entities by the application. Application decomposition is the first step towards identifying potential threat targets. The result of the application decomposition exercise for the TSM application is presented in Table 1.

Table 1. Application decomposition for the TSM application

S/N	Asset (Component)	Description	Trust Level	Entry Point
1	TSM Engine (TSM _e)	TSM application aggregator and request handler	Administrator	Web server
			App users	HTTP port
2	Multimodal Journey Planner (MJP)	A web application that provides journey options	TSM components	HTTP Port
3	Consumption Estimator Calculator (CEC)	A web application that calculates energy consumption based on journey option	TSM components	HTTP Port
4	Database (DB)	A component that stores users' personal data and activity logs	TSM components	HTTP Port

The application decomposition exercise shows that the TSM application is made up of four major components. Apart from the Database, which is a storage component, all the other components are web applications.

4.1.1 Threat Identification

Following the decomposition of the TSM application is the identification of threats using the threat modelling technique recommended by OWASP [20]. The aim is to identify the threats that are most likely to impact the TSM application, particularly by analyzing the application bearing in mind the potential attacker's intention. It involves classifying the threats using the STRIDE methodology. The likely threats particular to each TSM application component are identified. The threats have been identified according to the nature and functionality of the TSM application components. Specifically, as the TSM application consists of web applications and storage components hosted in cloud environments, threats that affect web applications, storage components and cloud applications have been

identified using several web application security, storage and cloud security threat catalogues as presented in [77]–[81]. The threats most likely to affect each of the TSM application components are presented in Table 2.

Table 2. Identified threats for TSM components

Cat.	TSMc	MJP	CEC	Database
S	Broken authentication Sensitive data disclosure	Man in the middle	Man in the middle	Man-in-the-middle Sensitive data disclosure
T	Injection flaws Cross site scripting (XSS)	Injection flaws	Injection flaws	Modifying metadata Injection flaws
R	Overly permissive cross-domain whitelist		Overly permissive cross-domain whitelist	
I	Access token leaks Insecure direct object reference Obtain access tokens	Obtain access tokens Insecure direct object reference Weak identity, credential & access management	Obtain access tokens Weak identity, credential & access management	Data breaches Sniffing storage traffic
D	DoS DDoS	DoS DDoS	DoS	DoS Deletion of data
E	Unauthorized access to admin interface Over privileged applications and accounts Account hijacking		Unauthorized access to admin interface Over privileged applications and accounts	Unauthorized access to admin interface Over privileged applications and accounts Resource owner impersonation Account hijacking

In Table 2, the threats likely to affect each component have been presented according to the STRIDE categories. This ensures easy understanding of the nature of the threats as well as the intention of a potential attacker and likely effect of each threat following an attack. A detailed description of all the threats listed above is provided in Appendix A.

4.1.2 Risk Analysis

Following the identification and categorization of threats using the STRIDE methodology, the process continues with risk assessment i.e., the evaluation and ranking of the risk related to the threats to estimate the degree of potential harm and determine the most critical components in the TSM application. The risk is analyzed based on threat categories (i.e., using STRIDE categorization), thereby giving a general overview of the risk based on the threat category. For instance, the spoofing risk in an application component represents the risk associated with all the spoofing threats identified in that component.

The risk is computed using a qualitative risk model i.e., (Risk = Probability x Impact). The probability refers to the likelihood of the threat while the impact is the extent of damage resulting from the threat. Figure 20 shows the outcome of the risk analysis process.

	TSM Engine (TSMe)		Multimodal Journey Planner (MJP)		Consumption Estimator (CE)		Database (DB)	
Spoofing	TSMe.S	MJP.S	CE.S	DB.S				
	10	10	5	5	5	3	10	10
Tampering	TSMe.T	MJP.T	CE.T	DB.T				
	9	7	5	2	5	5	10	10
Repudiation	TSMe.R	MJP.R	CE.R	DB.R				
	5	8	2	2	5	5	3	5
Information disclosure	TSMe.I	MJP.I	CE.I	DB.I				
	10	8	6	2	7	5	9	10
Denial of Service	TSMe.D	MJP.D	CE.D	DB.D				
	10	10	5	5	5	5	8	8
Elevation of privileges	TSMe.E	MJP.E	CE.E	DB.E				
	8	8	5	2	7	8	10	10
Total Risk	447		86		181		469	

Figure 20. TSM Application risk assessment

In Figure 20, for each threat category and each component, the threat likelihood value is on the left, the level of impact of the threat is on the right while the computed risk is below the two values. The sum of all the calculated risks is presented at the bottom of the table. The database component has the highest risk while the multimodal journey planner has the lowest risk. This is quite understandable owing to the type of information that these components handle. The database stores sensitive and personal data of the application users and hence possesses a higher risk as it becomes a big target for potential attackers. The multimodal journey planner does not store any sensitive information and thus is of less interest to attackers. In conclusion, the risk analysis exercise shows that the TSM engine and the Database are the most critical components of the TSM application and thus must be adequately guarded. All the components of the TSM application must be protected to avoid any security ‘loophole’ through the application can be attacked. In the next section, the appropriate security controls are selected and presented.

4.2 Selection of Security Metrics and Security Controls

In this section, the relevant security metrics and security controls for the TSM application are selected. This is important because it helps to identify monitorable parameters for threat detection as well as adequate countermeasures to mitigate identified threats and restore normalcy. These selections are made using relevant security metrics and security controls standards and frameworks. The selected security metrics and security controls for the TSM application components are presented in the next section.

4.2.1 Selection of Security Metrics

Security metrics are selected according to the interest of the application owner, which is enshrined in the security objectives of the application (primarily confidentiality, integrity and availability). In addition, the metrics are also selected such that they can be used to monitor different application behaviors that could lead to the detection and identification of threats, particularly when their thresholds become violated. In this thesis, the basic security objectives were considered and relevant sources of security metrics [53]–[55], [82], [83] were consulted to select the security metrics for the TSM application. The initial analysis leading to the selection of the metrics for the TSM application began with the declaration of the security objectives. The Fishbone diagram presented in Figure 21 adequately captures this.

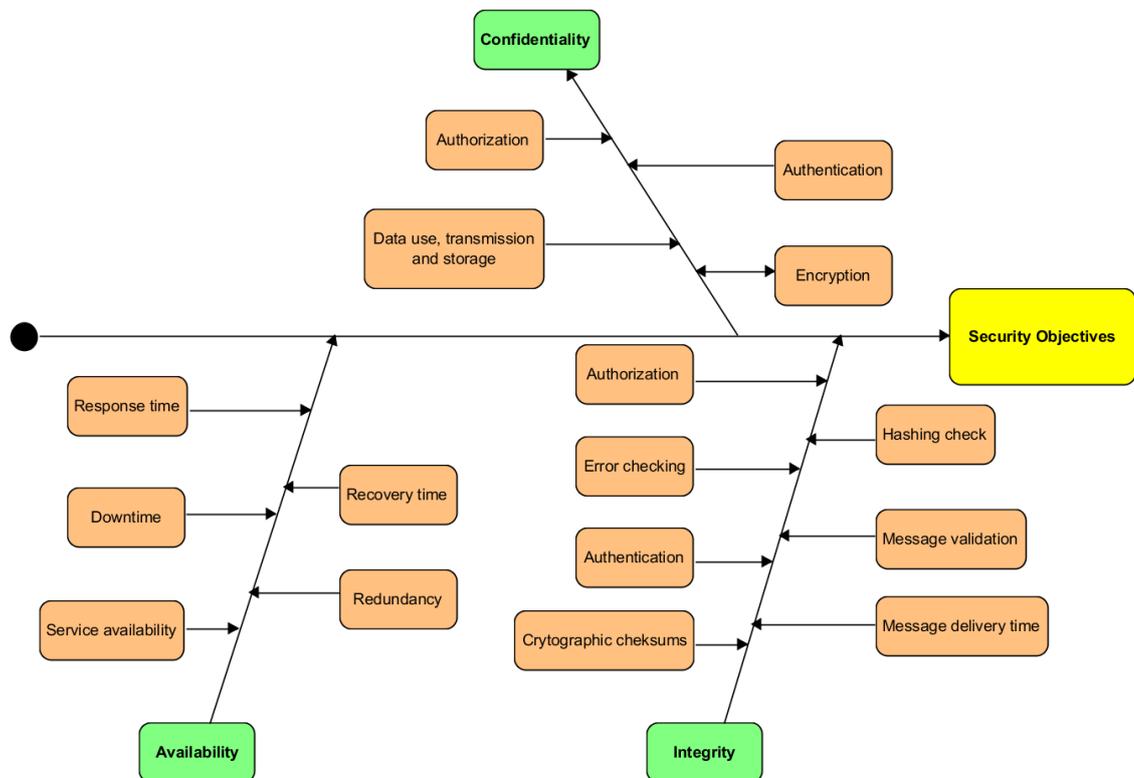


Figure 21. TSM Application security objectives

In the diagram above, the interest of the application owner is duly expressed according to the security objectives. The different sub-factors emanating from the core objectives (i.e., integrity, confidentiality and availability) represents the activities that are of monitorable or measurable interest, and based on this and in conjunction with the identified threats, the appropriate metrics are selected. For instance, for an availability sub-factor such as **redundancy** and an availability threat such as Denial of Service (**DoS**), a metric such as **level of redundancy** is selected to measure the capability of the application to handle multiple and concurrent requests before it becomes unavailable. The higher the level of redundancy, the more available the system is expected to be. The security metrics selected for the TSM application are presented in Table 3.

Table 3. Selected security metrics for TSM application according to identified threats

Identified threats	Selected Security metrics	Specified Thresholds
Broken authentication	Identity assurance	Level ≥ 3
Sensitive data disclosure	HSTS (HTTP Strict Transport Security)	Yes
	TLS cryptographic strength	7
	Data encryption	Yes
	Level of confidentiality	4
Man in the middle (MiTM)	Identity assurance	Level ≥ 3
	HSTS (HTTP Strict Transport Security)	Yes
	TLS cryptographic strength	7
	Data encryption	Yes
	Level of confidentiality	4
Injection flaws	Log unalterability	2
	SQL Injection	No
Cross site scripting (XSS)	Identity assurance	Level ≥ 3
	Level of confidentiality	4
	Vulnerability measure	100%
Modifying metadata	Log unalterability	2
Overly permissive cross-domain whitelist	Identity assurance	Level ≥ 3
Access token leaks	HTTP to HTTPS redirects	Yes
	Data encryption	Yes
Insecure direct object reference	Identity assurance	Level ≥ 3
	Access control and enforcement	Failed attempts <3
Obtain access tokens	Data encryption	Yes
	Identity assurance	Level ≥ 3
Weak Identity, credentials & access management	Identity assurance	Level ≥ 3
	Access control and enforcement	Failed attempts <3
	Data encryption	Yes

Data breaches	HSTS (HTTP Strict Transport Security)	Yes
	TLS cryptographic strength	7
	Data encryption	Yes
	Level of confidentiality	4
Sniffing storage traffic	HSTS (HTTP Strict Transport Security)	Yes
	TLS cryptographic strength	7
Denial of Service (DoS) & Distributed Denial of Service (DoS)	Level of redundancy	3
	Service availability	$\geq 99.99\%$
Deletion of data	Level of redundancy	3
Unauthorized access to admin interface Over privileged applications and accounts	Identity assurance	Level ≥ 3
	Access control and enforcement	Failed attempts <3
Account hijacking Resource owner impersonation	Identity assurance	Level ≥ 3
	Access control and enforcement	Failed attempts <3
	Personnel security screening measure	100%

In Table 3, the set(s) of security metrics required for monitoring the different identified TSM application threats is presented. In addition, thresholds have been specified for each metric. A violation of the specified metric threshold results in a threat. A detailed information about each of the selected metrics is provided in Appendix B.

4.2.2 Selection of Security Controls

Security controls are essential to mitigate threats. According to the threats identified in the TSM application, sufficient security controls have been selected to combat them. These controls have been selected using the Cloud Control Matrix (CCM) [57] and NIST SP 800-53 Rev. 4 [56] security control frameworks. The selected controls are aligned with the threat categories and identified threats for each component. Specifically, for each threat, the required security control property is identified using the OWASP methodology [36] and then based on this, the security control domain (i.e., control group) is identified and security control is selected. The security control domain is identified using the CCM standard and for each security control domain, several security controls have been recommended, out of which the most appropriate security control(s) needed to mitigate the threat is then selected. Additional NIST security controls may also be selected based on the peculiarity of each threat. The selected security controls are presented in Table 4.

Table 4. Selected security controls for TSM application threats

Identified threats	Recommended security control property (OWASP) [36]	Security control domain (CCM) [57]	Selected security control (NIST) [56]
Broken authentication	Strong authentication	IAM-12	IA-5

Sensitive data disclosure	Protection of secret data	EKM-03	SC-8(1), SC-13, SC-23
Cross site scripting (XSS)	Message integrity protocols	EKM-03	SC-8
Injection flaws	Message integrity protocols	AIS-03 & AIS-04	SI-10
Overly permissive cross-domain whitelist			SA-13
Insecure direct object reference	Strong authorization	IAM-09	AC-3, AC-6
Access token leaks	Message confidentiality protocols	EKM-03	SC-8
Obtain access tokens	Don't store secrets	EKM-04	SC-12
Denial of Service (DoS)		IVS-13	SC-5
Distributed Denial of Service (DoS)		IVS-13	SC-5(3)
Unauthorized access to admin interface	Use least privileges	IAM-05	AC-5, AC-6
Over privileged applications and accounts	Use least privileges	IAM-05	AC-5, AC-6
Account hijacking	Use least privileges	IAM-09	AC-5, AC-6
Weak identity, credential & access management	Strong encryption	EKM-03	AC-2, SC-13
Man-in-the-middle (MiTM)	Strong authentication	IAM-12	IA-5, SC-23
Modifying metadata	Tamper-resistant protocols	IAM-01	SA-18
Data breaches	Message confidentiality protocols	EKM-03	AC-2, SC-13
Sniffing storage traffic	Message confidentiality protocols	EKM-03	SC-8
Deletion of data		IVS-13	CP-9(6)
Resource owner impersonation		IAM-09	AC-2, IA-2(13)

Table 4 shows the security controls required for the mitigation of the TSM application identified threats. The recommendation provided by OWASP and the classification matrix given by CSA (i.e., CCM), has made it seamless to identify and select sufficient security controls needed to counter the identified threats likely to affect the TSM application. A detailed explanation of the security controls domain and the selected security controls is provided in Appendix C and Appendix D respectively.

4.3 Application Modelling

Application modelling is an important step required for developing and formulating the security policy, which is the lifeline of the security policy engine. In this section, the TSM application model is presented. The model has been developed using the Ontology Web Language (OWL) and it represents the security knowledge of the application, detailing application requirements (components, VMs, CSP information etc.), security requirements, meta-information, dependencies and interactions in the form of classes such as **asset**, **threat**, **security control**, properties such as **ownsAsset**, **hasThreat**, **hasAttackGoal**, constraints and several instances. The specific details of the classes and properties are presented in the next section.

4.3.1 Classes

The TSM application model (i.e., security policy ontology) consists of 24 classes, which includes asset, threat, threshold etc. The classes were defined based on the security requirements of the TSM application, thereby ensuring that the security knowledge of TSM application is adequately represented. Some of the classes have sub-classes, which help to provide more information and description about the parent classes. In addition, there are also **instances** defined, which represents class membership. For instance, the '*Threat*' class consists of six sub-classes, namely '*denial of service*', '*elevation of privileges*', '*information disclosure*', '*repudiation*', '*spoofing*' and '*tampering*' with several instances. For example, deletion of data threat is an instance of the denial of service class and its basic ontology is presented in Figure 22.

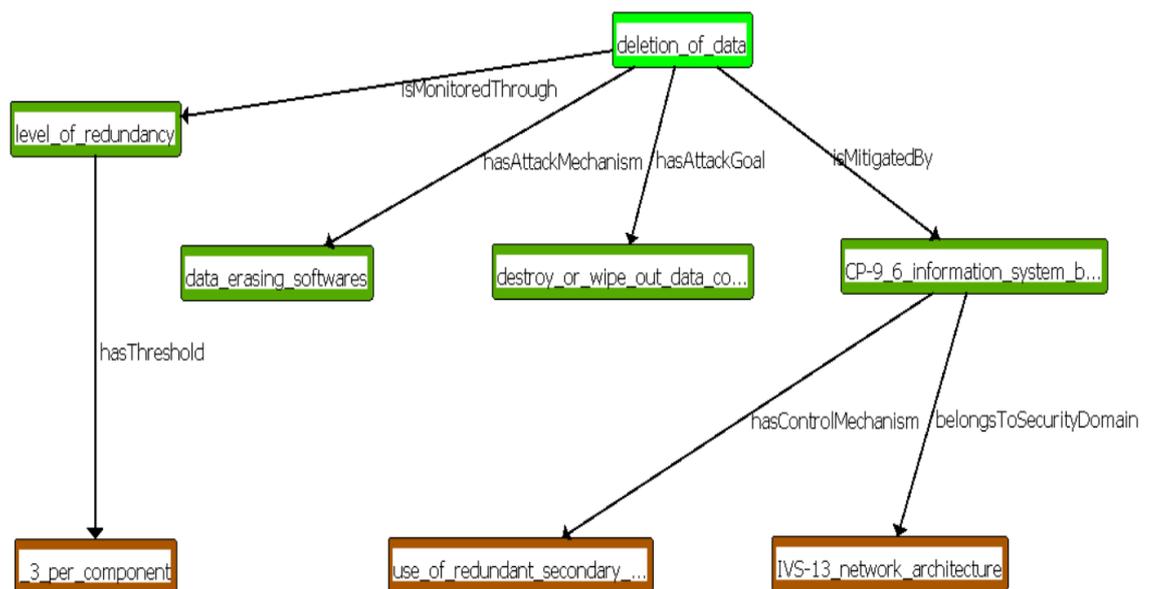


Figure 22. Ontology of the 'deletion of data' threat

The details of all the classes in the TSM application model are presented in Table 5.

Table 5. Details of all classes in the TSM application model

S/N	Classes	Description
1	Asset	An entity of great value belonging to an organization. It is also an attack target e.g., a database
2	Attack Goal	The intention behind perpetuating an attack e.g., to steal data
3	Attack Mechanism	The mode of perpetuating an attack e.g., brute force
4	Cloud Service Provider (CSP)	An organization that offers storage or software services through the cloud e.g., AWS
5	Communication Port	A physical or virtual interface for communication e.g., serial port
6	Deployment Model	A type of cloud environment e.g., public cloud
7	Entry Point	An interface through which an application component communicates with external entities e.g., HTTP port
8	Geographical Location	A physical location or region on the earth surface e.g., Africa
9	Hardware Requirement	The physical computer resources required by a software or operating system e.g., RAM
10	Operating System	A software platform that aids a computer's basic functions e.g., Ubuntu
11	Organization	An establishment, company or firm e.g., Tampere University of Technology
12	Risk Level	The level of criticality of a threat e.g., High
13	Security Monitoring Probe	A software program that monitors different security variables on a network, an application or operating system
14	Security Control	A countermeasure required for mitigating a threat e.g., access control
15	Security Control Mechanism	A technique for applying a security control e.g., use of certificates
16	Security Control Property	A required attribute of a security control e.g., strong authentication
17	Security Controls Domain	The security group to which a security control belongs e.g., encryption and key management
18	Security Metrics	A set of parameters for monitoring application performance, characteristics or behaviors e.g., level of redundancy
19	Security Objectives	The fundamental goals of security e.g., confidentiality
20	Service Model	A framework for providing cloud computing services e.g., PaaS
21	Threat	Anything that can cause harm to an asset or that can lead to an attack e.g., denial of service (DoS)
22	Threshold	A limit which when exceeded, indicates the likelihood of a threat
23	Trust Level	The access rights granted to external entities by an application e.g., admin
24	Virtual Machine (VM)	A software computer with an operating system

In Table 5, the description of each of the defined class for the TSM application model is given. This helps to provide a clear understanding of the classes as we as the major components of the TSM application model.

4.3.2 Properties

Following the definition of classes is the definition of properties. In the TSM application model, there are 21 properties, which describes the relationship between different classes. The properties were defined based on the classes and mainly represents the associations between them. For instance, the ‘ownsAsset’ property describes the binary relationship between the ‘Organization’ and ‘Asset’ classes. *Organization* is the domain while *Asset* represents the range. This simply means that *Assets* are owned by *Organizations*. Figure 23 shows all the properties in the TSM application model as defined on the Olingvo tool.

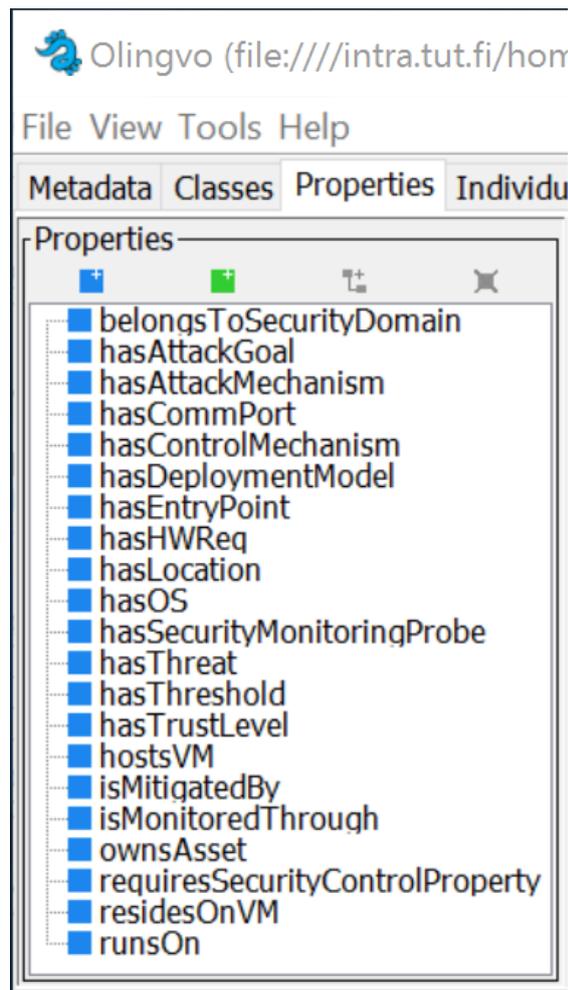


Figure 23. TSM Application model properties

The details of all the properties in the TSM application model are presented in Table 6.

Table 6. Details of all properties in the TSM application model

S/N	Properties	Description
1	belongsToSecurityDomain	An object property that associates security controls with security controls domain
2	hasAttackGoal	An object property that associates threats with attack goals

3	hasAttackMechanism	An object property that associates threats with attack mechanisms
4	hasCommPort	An object property that associates assets with communication port
5	hasControlMechanism	An object property that associates security controls with security control mechanisms
6	hasDeploymentModel	An object property that associates CSPs with cloud deployment models
7	hasEntryPoint	An object property that associates assets with entry point
8	hasHWReq	An object property that associates virtual machines with hardware requirements
9	hasLocation	An object property that associates CSPs with geographical location
10	hasOS	An object property that associates virtual machines with operating system
11	hasSecurityMonitoringProbe	An object property that associates assets with security monitoring probes
12	hasThreat	An object property that associates assets with threats
13	hasThreshold	An object property that associates security metrics with threshold
14	hasTrustLevel	An object property that associates assets with trust levels
15	hostsVM	An object property that associates CSPs with virtual machines
16	isMitigatedBy	An object property that associates threats with security controls
17	isMonitoredThrough	An object property that associates threats with security metrics
18	ownsAsset	An object property that associates organization with assets
19	requiresSecurityControlProperty	An object property that associates threats with security control property
20	residesOnVM	An object property that associates assets with virtual machines
21	runsOn	An object property that associates security monitoring probes with virtual machines

Table 6 shows the relationship that exists between different classes of the TSM application model. It clearly indicates the properties that bind different classes together. A summary of all the classes, properties and instances that make up the TSM application model is presented in Table 7.

Table 7. A summary of classes, properties and instances in the TSM application model

S/N	Properties	
1	Number of Classes	24
2	Number of Properties	21
3	Number of Individuals (Instances)	200

5. RESULTS

This section presents the results of the framework operations carried out in Chapter 4. It covers threat identification, risk analysis, selection of security metrics, selection of security controls and application modelling. The threat modelling exercise brought about the identification of 20 different threats (Table 2) that are likely to affect the TSM application, with the TSM engine and database components having the most number of threats, out of which injection flaws and denial of service (DoS) threats were the most significant threats. The risk analysis process also revealed the TSM engine and database components as having the highest risk levels amongst all the TSM application components. These results have showed that the TSM engine and the database components are the most critical components of the TSM application. This is largely attributed to the sensitive nature of the data and information processed by the two components.

A total number of 13 security metrics with specified thresholds (Table 3) and 18 security controls (Table 4) were selected to monitor the threats levels and mitigate the threats respectively. Other relevant properties such as organization, CSP information and communication ports were defined to completely specify and describe the security and application requirements of the TSM application. The specification of these properties resulted in the development of the TSM application model, which represents the security knowledge of the multi-cloud environment. In Appendix E, a snippet of the TSM application ontology is presented. The development of the TSM application model is an integral part of the steps required to achieve transparency and security awareness. It is a fundamental aspect of the security evaluation framework upon which all other engines depend. Without it, the engines cannot carry out their functions, security evaluation cannot be achieved, and thus transparency and security awareness becomes impossible.

The Olingvo tool developed by the FAST-Lab was utilized in formulating the TSM application model. The significant aspects of the TSM application model are the application/security requirements represented as '*classes*', the interactions between them represented as '*properties*' and the application/security requirements types represented as '*instances*'. These different aspects of the TSM application model are shown in Figure 24, Figure 25 and Figure 26.

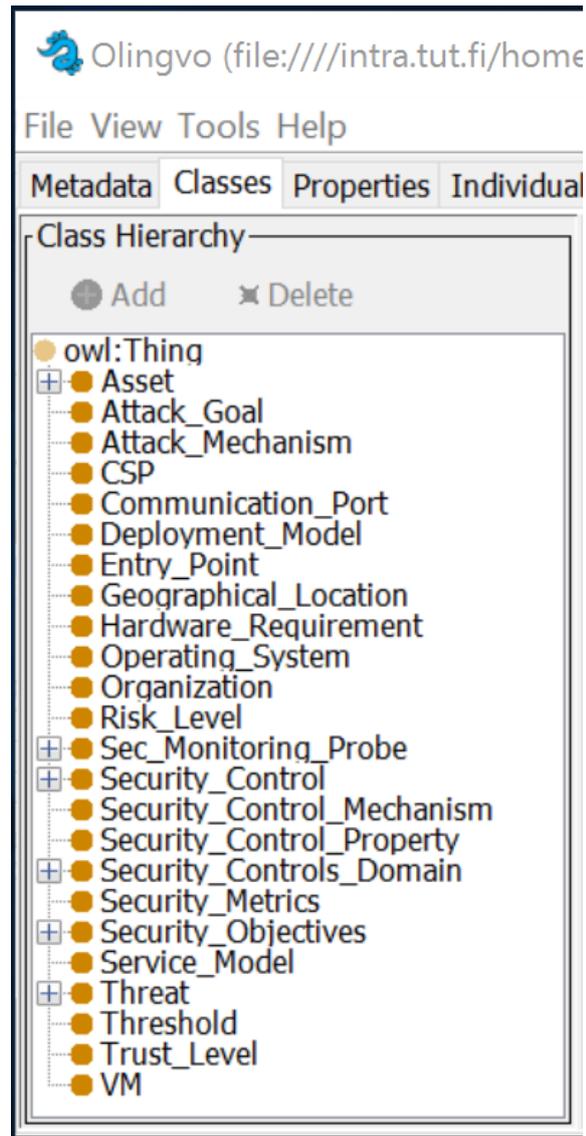


Figure 24. TSM Application model (showing classes)

Figure 24 presents all the TSM application model classes. They represent the major aspects of the TSM application required for describing the security requirements of the TSM application.

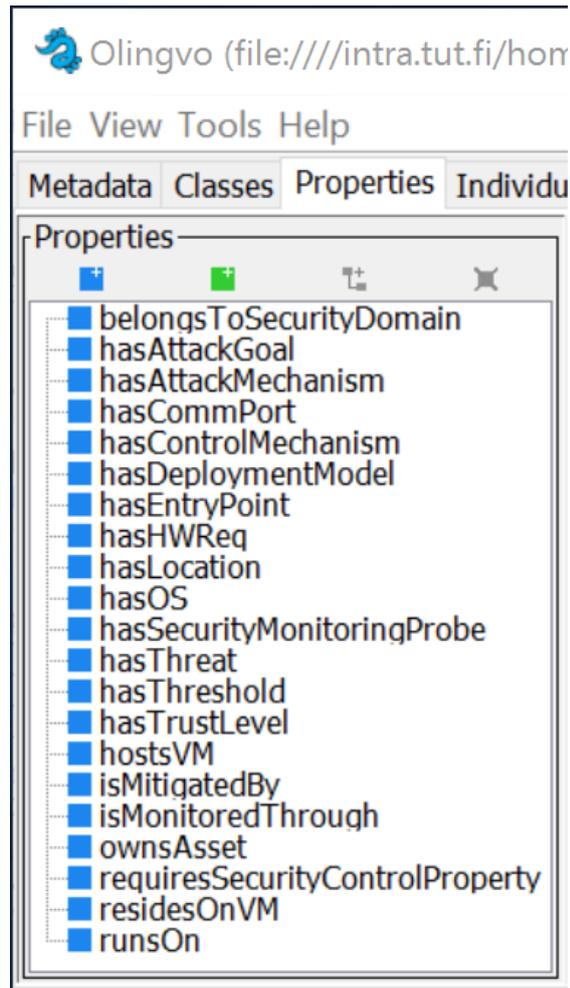


Figure 25. TSM Application model (showing properties)

In Figure 25, the most relevant properties of the TSM application model are shown. They represent the associations or relationships that exist between different classes and help in understanding how they interact with one another.

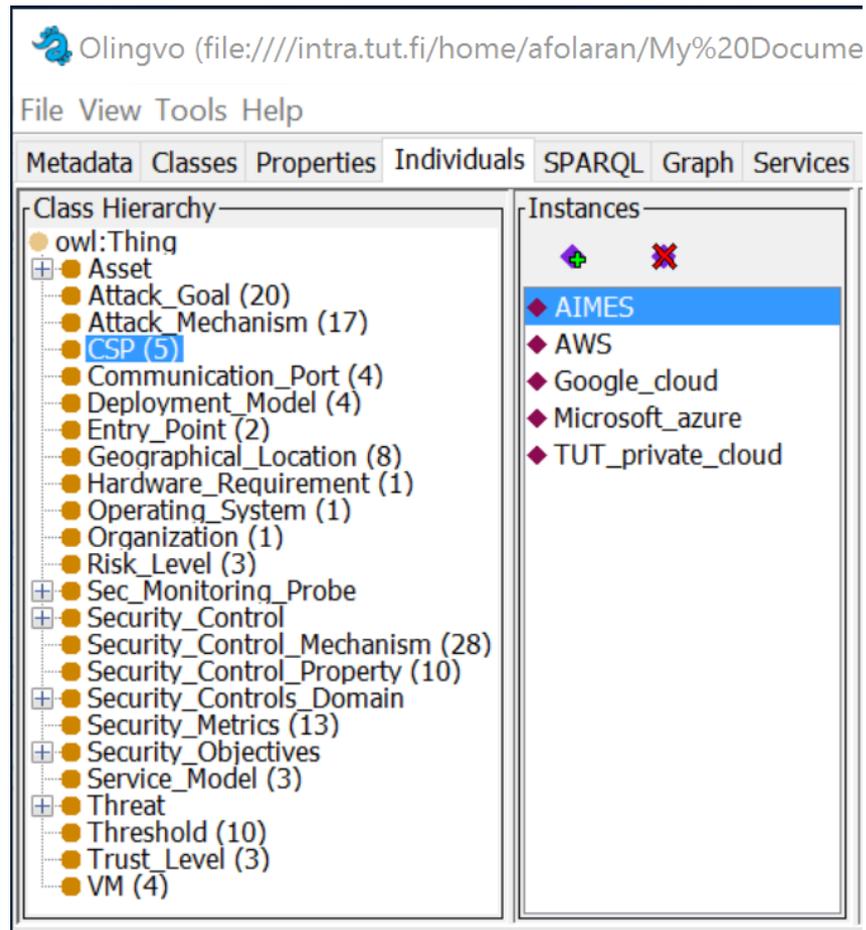
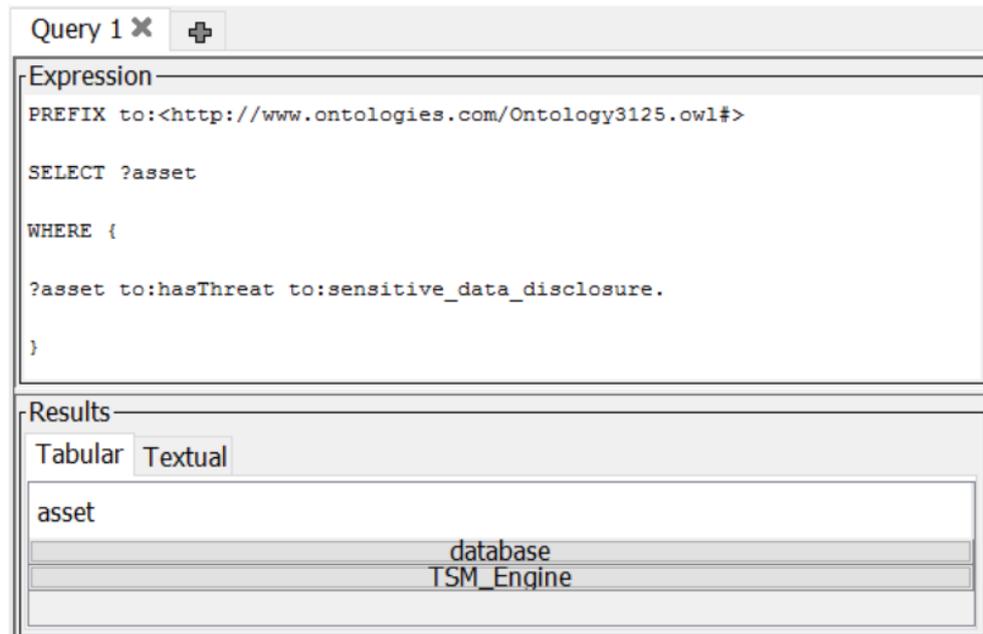


Figure 26. TSM Application model (showing instances of the CSP class)

Figure 26 shows the defined instances of the CSP class of the TSM application model. They represent the declarations or examples of CSPs e.g., Google cloud. In addition to the different aspects of the TSM application model that has been presented, a typical query for retrieving information from the application model is shown in Figure 27, where the response is also captured.



The screenshot shows a query editor window titled "Query 1". The "Expression" section contains the following SPARQL query:

```
PREFIX to:<http://www.ontologies.com/Ontology3125.owl#>

SELECT ?asset

WHERE {

  ?asset to:hasThreat to:sensitive_data_disclosure.

}
```

The "Results" section is set to "Tabular" view and displays the following results:

asset
database
TSM_Engine

Figure 27. A sample query/response from the TSM application model

In Figure 27, a query is sent to retrieve all components that are affected by 'sensitive data disclosure' threat. The response to the query reveals that the Database (DB) and TSM engine (TSM_e) are the components affected by the threat.

6. DISCUSSIONS

The heterogeneous nature of multi-cloud environments introduces dynamism within the environment particularly regarding the health and security status of the components and applications hosted in the environment and thereby requires adequate security awareness and transparency within the environment. Security awareness and transparency will ensure that the application/component security status as well as internal events and interactions within the environment can be determined in real time. To achieve this, a security evaluation framework was proposed in this thesis. This is to enable the security assessment of the multi-cloud environment, thereby revealing the security status of the environment and its components at any time required.

The proposed security evaluation framework consists of five engines;

- Request handling engine
- Security policy engine
- Metrics monitoring engine
- Security measurement engine
- Decision & analytics engine

They all work collaboratively to bring about transparency and security awareness. The procedures for evaluating security in multi-clouds have been outlined in this thesis through the different operations provided by the different engines. To demonstrate these procedures, the TSM application was used as a case study with focus on threat identification, risk analysis, security metrics, security controls and application modelling, which is provided by the security policy engine (SPE). In this thesis, the application modelling operation was explored, as all other engines rely on the application model for their operations.

In identifying the threats, the peculiar nature and operation of the TSM application components were studied and likely threats were selected using relevant threat catalogues. The threat catalogues provided an opportunity to easily identify and select the threats that could affect the TSM application components. In particular, threats relating to storage and web applications were selected owing to the nature of the TSM application components. As regards risk analysis, the threats were categorized based on STRIDE, thus making it possible to recognize and understand the nature of each threat before calculating the risks according to the threat categories. The risk calculations showed that the database and TSM engine are the most critical components, clearly pointing out that these components are the most likely to be attacked.

In determining the security metrics, the selection of security metrics leveraged on several metrics defined and proposed by reputable standards organization such as NIST and relevant EU funded research projects such as A4Cloud⁶, SPECS⁷ and MUSA⁸. In particular, the choice of selecting and using metrics defined by these projects was based on the fact that they were cloud and multi-cloud related projects. For the security controls, the security evaluation framework relied on NVD database (i.e., NIST security controls) and CCM for the selection of security controls. These controls have been known to be adequate to provide necessary countermeasure to mitigate threats as they have been extensively referenced in many literatures and research projects.

The operations mentioned in the aforementioned paragraphs culminated in the development of the TSM application model, which represents the foundation of the security evaluation framework. The TSM application model was developed using ontology. Ontology is machine-readable, inductive, and reusable and brings about the development of domain knowledge, hence its choice to development of the TSM application model. Specifically by using ontology, the security domain knowledge of the TSM application has been developed and it provides an opportunity to update and extend the KB in order to accommodate the complexity of the multi-cloud environment. The ability to capture the complexities make ontology suitable for application modelling.

The TSM application model (security ontology) is robust as it consists of 24 classes, 21 properties and 200 instances. This is as a result of the extensive description of the TSM application to accommodate the appropriate application and security requirements, together with their relationships. The model can also be extended to encapsulate the dynamics within the multi-cloud environment. The TSM application model serves as a base for the operations of the other engines in the framework. Through these engines, the framework will ensure security monitoring of all application components and the entire multi-cloud environment, detection of threats and unusual behaviors, mitigation of threats, reporting of security status and restoration of normal operating conditions within the multi-cloud environment.

The realization of the aforementioned framework operations will lead to security awareness and transparency in the multi-cloud environment as users and application owners become more knowledgeable about the operation, execution and most importantly the security status of their applications. The main benefits attached to this are that multi-cloud users will have control over their assets, security management becomes seamless for CSPs and cloud customers' trust and confidence in using multiple cloud resources increases.

⁶ <http://cloudaccountability.eu>

⁷ <http://www.specs-project.eu/>

⁸ <https://www.musa-project.eu/>

7. CONCLUSION

Multi-cloud adoption continues to grow owing to its benefits. On the contrary, its level of complexity continues to increase owing to its heterogeneous and open nature, which increases its attack surface and introduces various threats and vulnerabilities. This has clearly made security a major issue to ponder about deeply and has affected the trust and confidence of cloud customers. It has led to the difficulty of establishing the security status of application components hosted in multiple clouds and has given rise to the need for transparency and security awareness in multi-cloud environments. Therefore, it is required to design and implement an approach for managing security management in multi-cloud environments. It is in view of this that this thesis has been conducted.

Transparency and security awareness are required in multi-cloud environments to enable application owners determine the security status of their application as well as the entire multi-cloud environment, and to verify that CSPs are granting the SLA. In this thesis, a security evaluation framework has been proposed. In addition, the procedures for actualizing transparency and security awareness through security evaluation have been clearly explained. The framework has brought about the identification of threats, estimation of risk, identification of the most critical application components, development of application model, selection of security metrics and security controls. Although a major issue might be the ability of the desired CSP to grant the required security control as it is possible that not all CSPs may grant the security controls.

The framework involves application modelling, application security monitoring, security measurement, decision making and security status visualization. In this thesis, major emphasis has been placed on application modelling. The application modelling exercise gave rise to the application model, which is an organic aspect of the framework. Ontology played a major role in the model development as different security concepts, relationships and mechanisms were defined and specified in the ontology, to adequately represent the application and security requirements. With this, the security ontology of the TSM application was actualized. While the Security Policy Engine (SPE) offers the application modelling functionality, the operations of all the other engines as well as the procedures for realizing them have been explained in this thesis.

The proposed security evaluation framework addresses the problem statements of this thesis, which relates to security evaluation, transparency and security awareness in multi-cloud environments as the framework methodology (Chapter 3) clearly explains how this may be achieved. The framework has been partially implemented as mentioned earlier, with efforts concentrated on the application modelling aspect, the most significant aspect of the framework. However, it is expected that the implementation of the entire security

evaluation framework will bring forth the realization of transparency and security awareness in the multi-cloud environment. Specifically, the proposed framework will offer;

- Evaluation of multi-cloud environment security and the presentation of evaluation results to the end users.
- Multi-cloud environment transparency and security awareness of multi-cloud application components.

The major area of improvement in the framework is the complete implementation of all the engines i.e., security policy, metrics monitoring, security measurement, and decision & analytics engines. This will ensure that security in multi-cloud environments can be promptly analyzed and assessed, and transparency and security awareness can be achieved through adequate security evaluation.

REFERENCES

- [1] R. P. V. chander, S. Elias, S. Shivashankar, and M. P, “A REST based design for Web of Things in smart environments,” in *2012 2nd IEEE International Conference on Parallel, Distributed and Grid Computing*, 2012, pp. 337–342.
- [2] D. Cook and S. Das, *Smart Environments: Technology, Protocols and Applications (Wiley Series on Parallel and Distributed Computing)*. New York, NY, USA: Wiley-Interscience, 2004.
- [3] D. J. Cook and S. K. Das, “How smart are our environments? An updated look at the state of the art,” *Pervasive Mob. Comput.*, vol. 3, no. 2, pp. 53–73, Mar. 2007.
- [4] G. M. Youngblood, E. O. Heierman, L. B. Holder, and D. J. Cook, “Automation Intelligence for the Smart Environment,” in *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, San Francisco, CA, USA, 2005, pp. 1513–1514.
- [5] H. Zhu *et al.*, “Review of state-of-the-art wireless technologies and applications in smart cities,” in *IECON 2017 - 43rd Annual Conference of the IEEE Industrial Electronics Society*, 2017, pp. 6187–6192.
- [6] D. Minoli, “IPv6 Over Low Power WPAN (6LoWPAN),” in *Building the Internet of Things with IPv6 and MIPv6: The Evolving World of M2M Communications*, Wiley Telecom, 2013, p. 392-.
- [7] M. Schappacher, E. Schmitt, A. Sikora, P. Weber, and A. Yushev, “A flexible, modular, open-source implementation of 6LoWPAN,” in *2015 IEEE 8th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*, 2015, vol. 2, pp. 838–844.
- [8] J. Gibson, R. Rondeau, D. Eveleigh, and Q. Tan, “Benefits and challenges of three cloud computing service models,” in *2012 Fourth International Conference on Computational Aspects of Social Networks (CASoN)*, 2012, pp. 198–205.
- [9] Y. Jain, “Top 5 Cloud Platforms and Solutions to Choose From.” [Online]. Available: <https://www.newgenapps.com/blog/top-5-cloud-platforms-and-solutions-to-choose-from>. [Accessed: 15-May-2018].
- [10] P. Mell and T. Grance, “SP 800-145, The NIST Definition of Cloud Computing,” National Institute of Standards & Technology, Sep. 2011.

- [11] “AWS vs Azure vs Google Cloud Market Share 2017 - Free Report,” *Skyhigh*, 13-Jul-2017. [Online]. Available: <https://www.skyhighnetworks.com/cloud-security-blog/microsoft-azure-closes-iaas-adoption-gap-with-amazon-aws/>. [Accessed: 15-May-2018].
- [12] “RightScale 2018 State of the Cloud Report.” [Online]. Available: <https://www.rightscale.com/lp/state-of-the-cloud>. [Accessed: 15-May-2018].
- [13] H. A. Latchman and A. V. Mundi, “Power Line Communication Technologies,” in *Smart Environments: Technologies, Protocols, and Applications*, Wiley-Blackwell, 2005, pp. 47–62.
- [14] D. R. Andersson *et al.*, “Smart access to small lot manufacturing for systems integration,” in *2018 Pan Pacific Microelectronics Symposium (Pan Pacific)*, 2018, pp. 1–9.
- [15] L. Wang and X. V. Wang, “Latest Advancement in CPS and IoT Applications,” in *Cloud-Based Cyber-Physical Systems in Manufacturing*, Springer, Cham, 2018, pp. 33–61.
- [16] P. A. Nixon, W. Wagealla, C. English, and S. Terzis, “Security, Privacy and Trust Issues in Smart Environments,” in *Smart Environments: Technologies, Protocols, and Applications*, Wiley-Blackwell, 2005, pp. 249–270.
- [17] G. Mantas, D. Lymberopoulos, and N. Komninos, “Security in Smart Home Environment,” *Wirel. Technol. Ambient Assist. Living Healthc. Syst. Appl.*, pp. 170–191, 2011.
- [18] N. A. Malik, M. Y. Javed, and U. Mahmud, “Threat Modeling in Pervasive Computing Paradigm,” in *2008 New Technologies, Mobility and Security*, 2008, pp. 1–5.
- [19] G. Martins, S. Bhatia, X. Koutsoukos, K. Stouffer, C. Tang, and R. Candell, “Towards a systematic threat modeling approach for cyber-physical systems,” in *2015 Resilience Week (RWS)*, 2015, pp. 1–6.
- [20] Microsoft, “The STRIDE Threat Model.” [Online]. Available: [https://docs.microsoft.com/en-us/previous-versions/commerce-server/ee823878\(v%3dcs.20\)](https://docs.microsoft.com/en-us/previous-versions/commerce-server/ee823878(v%3dcs.20)). [Accessed: 15-May-2018].
- [21] K. Stouffer, S. Lightman, V. Pillitteri, M. Abrams, and A. Hahn, “SP 800-82 Rev. 2, Guide to Industrial Control Systems (ICS) Security,” National Institute of Standards & Technology, 2015.

- [22] P. Wang, A. Ali, and W. Kelly, "Data security and threat modeling for smart city infrastructure," in *2015 International Conference on Cyber Security of Smart Cities, Industrial Control System and Communications (SSIC)*, 2015, pp. 1–6.
- [23] K. Beckers, D. Hatebur, and M. Heisel, "A Problem-Based Threat Analysis in Compliance with Common Criteria," in *2013 International Conference on Availability, Reliability and Security*, 2013, pp. 111–120.
- [24] K. Beckers, S. Faßbender, M. Heisel, and S. Suppan, "A Threat Analysis Methodology for Smart Home Scenarios," in *Smart Grid Security*, 2014, pp. 94–124.
- [25] D. Ghosh, R. Sharman, H. Raghav Rao, and S. Upadhyaya, "Self-healing systems — survey and synthesis," *Decis. Support Syst.*, vol. 42, no. 4, pp. 2164–2185, Jan. 2007.
- [26] H. Psaiar and S. Dustdar, "A survey on self-healing systems: approaches and systems," *Computing*, vol. 91, no. 1, pp. 43–73, Jan. 2011.
- [27] IBM, "An Architectural Blueprint for Autonomic Computing," Jun. 2006.
- [28] E. Grishikashvili Pereira, R. Pereira, and A. Taleb-Bendiab, "Performance evaluation for self-healing distributed services and fault detection mechanisms," *J. Comput. Syst. Sci.*, vol. 72, no. 7, pp. 1172–1182, Nov. 2006.
- [29] M. Sharmin, S. Ahmed, and S. I. Ahamed, "MARKS (Middleware Adaptability for Resource Discovery, Knowledge Usability and Self-healing) for Mobile Devices of Pervasive Computing Environments," in *Third International Conference on Information Technology: New Generations (ITNG'06)*, 2006, pp. 306–313.
- [30] L. McAvoy, L. Chen, and M. Donnelly, "An Ontology Based Context Management System for Smart Environments," in *Proceedings of the International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies (UBICOMM 2012)*, Barcelona, Spain, 2012, vol. 12, pp. 18–23.
- [31] A. Evesti and E. Ovaska, "Ontology-Based Security Adaptation at Run-Time," in *2010 Fourth IEEE International Conference on Self-Adaptive and Self-Organizing Systems*, 2010, pp. 204–212.
- [32] K. E. Kushida, J. Murray, and J. Zysman, "Cloud Computing: From Scarcity to Abundance," *J. Ind. Compet. Trade*, vol. 15, no. 1, pp. 5–19, Mar. 2015.
- [33] M. Badger, T. Grance, R. Patt-Corner, and J. Voas, "SP 800-146, Cloud Computing Synopsis and Recommendations," National Institute of Standards & Technology, May 2012.

- [34] International Organization for Standardization, “ISO/IEC 17826:2016 - Information technology -- Cloud Data Management Interface (CDMI).” [Online]. Available: <https://www.iso.org/standard/70226.html>. [Accessed: 15-May-2018].
- [35] E. Schouten, “Cloud computing defined: Characteristics & service levels,” *Cloud computing news*, 31-Jan-2014. [Online]. Available: <https://www.ibm.com/blogs/cloud-computing/2014/01/31/cloud-computing-defined-characteristics-service-levels/>. [Accessed: 19-May-2018].
- [36] The Open Web Application Security Project (OWASP), “Application Threat Modeling - OWASP.” [Online]. Available: https://www.owasp.org/index.php/Application_Threat_Modeling. [Accessed: 16-May-2018].
- [37] A. Bouayad, A. Blilat, N. E. H. Mejhed, and M. E. Ghazi, “Cloud computing: Security challenges,” in *2012 Colloquium in Information Science and Technology*, 2012, pp. 26–31.
- [38] R. E. Johnson, “Cloud computing security challenges and methods to remotely augment a cloud’s security posture,” in *2010 International Conference on Information Society*, 2010, pp. 179–181.
- [39] A. Behl, “Emerging security challenges in cloud computing: An insight to cloud security challenges and their mitigation,” in *2011 World Congress on Information and Communication Technologies*, 2011, pp. 217–222.
- [40] D. L. Quoc, L. Yazdanov, and C. Fetzer, “DoLen: User-Side Multi-cloud Application Monitoring,” in *2014 International Conference on Future Internet of Things and Cloud*, 2014, pp. 76–81.
- [41] V. Casola *et al.*, “MUSA Deployer: Deployment of Multi-cloud Applications,” in *2017 IEEE 26th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE)*, 2017, pp. 107–112.
- [42] N. Hochgeschwender, G. Biggs, and H. Voos, “A Reference Architecture for Deploying Component-Based Robot Software and Comparison with Existing Tools,” in *2018 Second IEEE International Conference on Robotic Computing (IRC)*, 2018, pp. 121–128.
- [43] C. Ebert, G. Gallardo, J. Hernantes, and N. Serrano, “DevOps,” *IEEE Softw.*, vol. 33, no. 3, pp. 94–100, May 2016.
- [44] A. Taha, S. Manzoor, and N. Suri, “SLA-Based Service Selection for Multi-Cloud Environments,” in *2017 IEEE International Conference on Edge Computing (EDGE)*, 2017, pp. 65–72.

- [45] M. M. Alshammari, A. A. Alwan, A. Nordin, and I. F. Al-Shaikhli, “Disaster recovery in single-cloud and multi-cloud environments: Issues and challenges,” in *2017 4th IEEE International Conference on Engineering Technologies and Applied Sciences (ICETAS)*, 2017, pp. 1–7.
- [46] J. Kosinska, J. Kosinski, and K. Zielinski, “The Concept of Application Clustering in Cloud Computing Environments: The Need for Extending the Capabilities of Virtual Networks,” in *2010 Fifth International Multi-conference on Computing in the Global Information Technology*, 2010, pp. 139–145.
- [47] C. Zhuo and Y. Xiaohu, “High available software architecture based on cluster technology,” in *TENCON '02. Proceedings. 2002 IEEE Region 10 Conference on Computers, Communications, Control and Power Engineering*, 2002, vol. 1, pp. 327–330 vol.1.
- [48] A. M. Ortiz, E. Rios, W. Mallouli, E. Iturbe, and E. M. de Oca, “Self-protecting multi-cloud applications,” in *2015 IEEE Conference on Communications and Network Security (CNS)*, 2015, pp. 643–647.
- [49] M. Singhal *et al.*, “Collaboration in multicloud computing environments: Framework and security issues,” *Computer*, vol. 46, no. 2, pp. 76–84, Feb. 2013.
- [50] K. Kritikos, T. Kirkham, B. Kryza, and P. Massonet, “Security Enforcement for Multi-Cloud Platforms – The Case of PaaS,” *Procedia Comput. Sci.*, vol. 68, pp. 103–115, Jan. 2015.
- [51] S. Hussain, A. Kamal, S. Ahmad, G. Rasool, and S. Iqbal, “THREAT MODELING METHODOLOGIES: A SURVEY,” *Sci IntLahore*, vol. 26, no. 4, pp. 1607–1609, 2014.
- [52] S. O. Afolaranmi, L. E. G. Moctezuma, M. Rak, V. Casola, E. Rios, and J. L. M. Lastra, “Methodology to Obtain the Security Controls in Multi-cloud Applications,” presented at the 6th International Conference on Cloud Computing and Services Science, 2016, vol. 1, pp. 327–332.
- [53] V. Casola, A. D. Benedictis, M. Rak, and U. Villano, “A Security Metric Catalogue for Cloud Applications,” in *Complex, Intelligent, and Software Intensive Systems*, 2017, pp. 854–863.
- [54] E. Chew, M. Swanson, K. M. Stine, N. Bartol, A. Brown, and W. Robinson, “SP 800-55 Rev. 1. Performance Measurement Guide for Information Security,” National Institute of Standards & Technology, Gaithersburg, MD, United States, 2008.
- [55] The Center for Internet Security, “CIS Security Metrics - Quick Start Guide v1.0.0.” 01-Nov-2010.

- [56] Joint Task Force Transformation Initiative, “SP 800-53 Rev. 4, Security and Privacy Controls for Federal Information Systems and Organizations,” National Institute of Standards & Technology, Apr. 2013.
- [57] Cloud Security Alliance, “Cloud Security Alliance: Cloud Controls Matrix v3.0.1 (9-1-17 Update),” 01-Sep-2017. [Online]. Available: <https://cloudsecurityalliance.org/download/cloud-controls-matrix-v3-0-1/>. [Accessed: 16-May-2018].
- [58] N. Ferry, H. Song, A. Rossini, F. Chauvel, and A. Solberg, “CloudMF: Applying MDE to Tame the Complexity of Managing Multi-cloud Applications,” in *2014 IEEE/ACM 7th International Conference on Utility and Cloud Computing*, 2014, pp. 269–277.
- [59] S. Bechhofer *et al.*, “OWL Web Ontology Language Reference,” Feb-2004. [Online]. Available: <https://www.w3.org/TR/owl-ref/>. [Accessed: 16-May-2018].
- [60] J. Jürjens, “UMLsec: Extending UML for Secure Systems Development,” in *«UML» 2002 — The Unified Modeling Language*, 2002, pp. 412–425.
- [61] R. J. Rodríguez, J. Merseguer, and S. Bernardi, “Modelling Security of Critical Infrastructures: A Survivability Assessment,” *Comput. J.*, vol. 58, no. 10, pp. 2313–2327, Oct. 2015.
- [62] Y. I. Khan and M. U. Ndubuaku, “Ontology-based automation of security guidelines for smart homes,” in *2018 IEEE 4th World Forum on Internet of Things (WF-IoT)*, 2018, pp. 35–40.
- [63] G. Denker, L. Kagal, T. Finin, M. Paolucci, and K. Sycara, “Security for DAML Web Services: Annotation and Matchmaking,” in *The Semantic Web - ISWC 2003*, 2003, pp. 335–350.
- [64] A. Kim, J. Luo, and M. Kang, “Security Ontology for Annotating Resources,” in *On the Move to Meaningful Internet Systems 2005: CoopIS, DOA, and ODBASE*, 2005, pp. 1483–1499.
- [65] J. Undercoffer, A. Joshi, and J. Pinkston, “Modeling Computer Attacks: An Ontology for Intrusion Detection,” in *Recent Advances in Intrusion Detection*, 2003, pp. 113–135.
- [66] A. Herzog, N. Shahmehri, and C. Duma, “An Ontology of Information Security,” *Tech. Appl. Adv. Inf. Priv. Secur. Emerg. Organ. Ethical Hum. Issues*, pp. 278–301, 2009.

- [67] S. Fenz and A. Ekelhart, "Formalizing Information Security Knowledge," in *Proceedings of the 4th International Symposium on Information, Computer, and Communications Security*, New York, NY, USA, 2009, pp. 183–194.
- [68] H. Xu, D. Xiao, and Z. Wu, "Application of Security Ontology to Context-Aware Alert Analysis," in *2009 Eighth IEEE/ACIS International Conference on Computer and Information Science*, 2009, pp. 171–176.
- [69] T. Takahashi, Y. Kadobayashi, and H. Fujiwara, "Ontological Approach Toward Cybersecurity in Cloud Computing," in *Proceedings of the 3rd International Conference on Security of Information and Networks*, New York, NY, USA, 2010, pp. 100–109.
- [70] L. Youseff, M. Butrico, and D. D. Silva, "Toward a Unified Ontology of Cloud Computing," in *2008 Grid Computing Environments Workshop*, 2008, pp. 1–10.
- [71] K. Bernsmed, A. Undheim, P. H. Meland, and M. G. Jaatun, "Towards an Ontology for Cloud Security Obligations," in *2013 International Conference on Availability, Reliability and Security*, 2013, pp. 577–581.
- [72] T. Salman, D. Bhamare, A. Erbad, R. Jain, and M. Samaka, "Machine Learning for Anomaly Detection and Categorization in Multi-Cloud Environments," in *2017 IEEE 4th International Conference on Cyber Security and Cloud Computing (CSCloud)*, 2017, pp. 97–103.
- [73] P. Carvallo, A. R. Cavalli, W. Mallouli, and E. Rios, "Multi-cloud Applications Security Monitoring," in *Green, Pervasive, and Cloud Computing*, 2017, pp. 748–758.
- [74] A. Evesti, "Adaptive security in smart spaces," Doctoral Thesis, University of Oulu, Finland, 2013.
- [75] A. Al-Ajlan, "The Comparison between Forward and Backward Chaining," *Int. J. Mach. Learn. Comput.*, vol. Vol. 5, no. No. 2, pp. 106–113, Apr. 2015.
- [76] Z. ChuangLu, "Research on the Semantic Web Reasoning Technology," *AASRI Procedia*, vol. 1, pp. 87–91, Jan. 2012.
- [77] Cloud Security Alliance, "Cloud Security Alliance: The Treacherous 12 Cloud Computing Top Threats in 2016." Feb-2016.
- [78] The Open Web Application Security Project (OWASP), "OWASP Top 10 2013." [Online]. Available: https://www.owasp.org/index.php/Top_10_2013-Top_10. [Accessed: 18-May-2018].

- [79] The Open Web Application Security Project (OWASP), “OWASP Top 10 Application Security Risks - 2017.” [Online]. Available: https://www.owasp.org/index.php/Top_10-2017_Top_10. [Accessed: 18-May-2018].
- [80] The Mitre Corporation, “Common Weakness Enumeration - CWE List Version 3.1.” [Online]. Available: <https://cwe.mitre.org/data/index.html>. [Accessed: 18-May-2018].
- [81] R. Hasan, S. Myagmar, A. J. Lee, and W. Yurcik, “Toward a Threat Model for Storage Systems,” in *Proceedings of the 2005 ACM Workshop on Storage Security and Survivability*, New York, NY, USA, 2005, pp. 94–102.
- [82] Cloud Accountability Project, “D:C-5.2 Validation of the accountability metrics.” 06-Oct-2014.
- [83] MUSA Project, “D4.1 Initial security assurance mechanisms and tools,” Nov. 2016.
- [84] M. Conti, N. Dragoni, and V. Lesyk, “A Survey of Man In The Middle Attacks,” *IEEE Commun. Surv. Tutor.*, vol. 18, no. 3, pp. 2027–2051, thirdquarter 2016.
- [85] T. Lodderstedt, M. McGloin, and P. Hunt, “RFC 6819 - OAuth 2.0 Threat Model and Security Considerations.” 2013.
- [86] OWASP, “Broken Access Control - OWASP.” [Online]. Available: https://www.owasp.org/index.php/Broken_Access_Control. [Accessed: 20-May-2018].
- [87] M. Gogan, “The threat of privileged user access - monitoring and controlling privilege users,” *SC Media UK*, 10-Nov-2016. [Online]. Available: <https://www.scmagazineuk.com/opinion/the-threat-of-privileged-user-access--monitoring-and-controlling-privilege-users/article/568624/>. [Accessed: 20-May-2018].
- [88] SPECS Project, “D2.2.2 Report on conceptual framework for SLA negotiation - Final,” Oct. 2015.

APPENDIX A: DESCRIPTION OF IDENTIFIED THREATS

This appendix presents a detailed description of the identified threats that are likely to affect the TSM application as presented in Table 2.

S/N	Threat Category	Threats	Description
1	Spoofing	Broken authentication [79]	A situation where authentication is not properly implemented, thus giving attackers the opportunity to compromise passwords, keys, tokens etc. to assume valid users' status
		Sensitive data disclosure [79]	The careless exposure of sensitive information at rest or in transit, usually due to the data not being encrypted
		Man-in-the-middle [84]	This involve intercepting the communication between two endpoints and altering the messages being sent
2	Tampering	Injection flaws [79]	This flaw involves additional and untrusted data is inserted and sent as part of a query. The intention is to modify the request
		Cross site scripting (XSS) [79]	A flaw that occurs when untrusted data is sent to a browser without validation. The intention is to hijack sessions and perform malicious activities.
		Modifying metadata [81]	This involves changing metadata in order to disrupt a storage system
3	Repudiation	Overly permissive cross-domain whitelist [80]	This occurs when a software component makes of a cross-domain policy that contains untrusted domains
4	Information Disclosure	Access token leaks [85]	This occurs when tokens are eavesdropped by an attacker during transmission. It occurs when the communication is not secure.
		Insecure direct object reference [78]	This occurs when a reference to an internal implementation object e.g., database key is exposed.
		Obtain access tokens [85]	This occurs when tokens are stolen from a client and used by an attacker for malicious actions
		Weak identity, credential & access management [77]	The use of poor authentication systems or weak passwords, which leads to information disclosure
		Data breaches [77]	A situation where sensitive data or information is viewed or processed without authorization
		Sniffing storage traffic [81]	This involves monitoring storage service traffic in order to steal data
5	Denial of Service	Denial of Service (DoS) [77]	This is aimed at making a resource unavailable for valid users. It may carried through disruption or resource overloading
		Distributed Denial of Service (DDoS) [77]	This is aimed at making a resource unavailable for valid users. It may carried through disruption or resource overloading
		Deletion of data [81]	This involves deliberate erasure of data in order to make data unavailable to the users

6	Elevation of Privileges	Unauthorized access to admin interface [86]	This involves gaining access to the admin interface without authorization
		Over privileged applications and accounts [87]	The use of a privileged program to obtain access to an application or account without authorization
		Account hijacking [77]	This involves stealing or taking over a user's account details for malicious intentions and actions
		Resource owner impersonation [85]	This involves illegally obtaining a user's credentials and gains authorization without the consent of the user

APPENDIX B: DESCRIPTION OF SELECTED SECURITY METRICS

This appendix presents a comprehensive description of the security metrics as proposed in [82], [83], [88]. The metrics have been selected to monitor the identified threats in the TSM application. In addition, the range over which the metrics is specified is listed.

S/N	Selected Security metrics	Description	Range
1	Access control and enforcement	This metric represents the number of valid access attempts, failed access attempts, access retries and also frequency of password change attempts	Integer > 0
2	Data encryption	This metrics checks if the data being transmitted/stored in cloud storage is encrypted and not in plain text	Yes/No
3	HSTS (HTTP Strict Transport Security)	This metric requires that the resource be transported or make available over a secure HTTP connection	Yes/No
4	HTTP to HTTPS re-directs	This metric requires that clients use only secure HTTP protocol for service delivery	Yes/No
5	Identity assurance	<p>This metrics specifies the quality of the authentication mechanisms</p> <p>Level 0: No authentication mechanisms are in place</p> <p>Level 1: Simple challenge response mechanisms are allowed and no identity proofing is required</p> <p>Level 2: Single factor remote network authentication is required; in this case, authentication is successful if the claimant proves control of the authentication token through a secure authentication protocol</p> <p>Level 3: Multifactor authentication mechanisms are in place. Proofs of control of the authentication token are done through a cryptographic protocol</p> <p>Level 4: Multifactor authentication with a hardware cryptographic token is required. Strong cryptographic mechanisms are required along physical tokens with a FIPS 140-2 level greater than 2, and identity proofing is done in person</p>	$0 \leq \text{integer} \leq 4$
6	Level of confidentiality	<p>This metric specifies the confidentiality level of a system. It involves 4 levels;</p> <p>Level 0: Data may be accessible by the cloud provider personnel for regular operational purposes, under the control of an authentication, authorization and accounting (AAA) mechanism</p> <p>Level 1: Data is accessible via AAA mechanism</p> <p>Level 2: Technical and organizational measures are in place so that data may only be accessible to privileged CSP personnel (administrators) for debugging or maintenance purposes, under the control of an AAA mechanism</p> <p>Level 3: Technical and organizational measures are in place so that data is only accessible to privileged CSP per-</p>	$0 \leq \text{integer} \leq 4$

		<p>sonnel to respond to law enforcement or extraordinary requests made by the client, under the control of an AAA mechanism.</p> <p>Level 4: Data is encrypted by the client with cryptographic keys that cannot be ascertained by the provider.</p>	
7	Level of redundancy	This metric specifies the number of replicas of a software component set up during system operation	Integer > 0
8	Log unalterability	<p>This metric specifies the protection level of the log management system against tampering.</p> <p>Level 0: No integrity mechanisms are in place</p> <p>Level 1: Log integrity is protected only by access control measures</p> <p>Level 2: Cryptographic mechanisms are in place for guaranteeing log unalterability or WORM (Write Once Read Many) devices are used.</p>	$0 \leq \text{integer} \leq 2$
9	Personnel security screening measure	This metrics measures the percentage of individuals screened before being granted access to organizational information and information systems	$0 \leq \text{integer} \leq 100$
10	Service availability	The percentage amount of time that the service is available to users	$0 \leq \text{integer} \leq 100$
11	SQL Injection	This metric monitors the queries to identify any SQL injection attempts	Yes/No
12	TLS cryptographic strength	<p>This metric measures the strength of the cryptosystem. The values (level 1-8) are based on ECRYPT recommendation 2012: https://www.keylength.com/en/3/. The default level is 7</p> <p>Level 1: Attacks in "real-time" by individuals. Only acceptable for authentication tag size.</p> <p>Level 2: Very short-term protection against small organizations. Should not be used for confidentiality in new systems.</p> <p>Level 3: Short-term protection against medium organizations, medium-term protection against small organizations</p> <p>Level 4: Very short-term protection against agencies, long-term protection against small organizations</p> <p>Level 5: Legacy standard level</p> <p>Level 6: Medium-term protection</p> <p>Level 7: Long-term protection</p> <p>Level 8: "Foreseeable future"</p>	$1 \leq \text{integer} \leq 8$
13	Vulnerability measure	<p>It measures the efficiency in percentage (%) of high vulnerabilities mitigated within organizationally defined time periods after discovery</p> <p>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.</p>	$0 \leq \text{integer} \leq 100$

APPENDIX C: DESCRIPTION OF SECURITY CONTROLS DOMAIN

This appendix explains the different security controls domain as provided in the CSA cloud control matrix (CCM) [57]. The security controls domain helps to classify the selected security controls.

S/N	Security control domain	Security domain ID	Description
1	Application & Interface Security (Data Integrity)	AIS-03	Data input and output integrity routines (i.e., reconciliation and edit checks) shall be implemented for application interfaces and databases to prevent manual or systematic processing errors, corruption of data, or misuse.
2	Application & Interface Security (Data Security/Integrity)	AIS-04	Policies and procedures shall be established and maintained in support of data security to include (confidentiality, integrity, and availability) across multiple system interfaces, jurisdictions, and business functions to prevent improper disclosure, alteration, or destruction.
3	Encryption & Key Management (Sensitive Data Protection)	EKM-03	Policies and procedures shall be established, and supporting business processes and technical measures implemented, for the use of encryption protocols for protection of sensitive data in storage (e.g., file servers, databases, and end-user workstations), data in use (memory), and data in transmission (e.g., system interfaces, over public networks, and electronic messaging) as per applicable legal, statutory, and regulatory compliance obligations.
4	Encryption & Key Management (Storage and Access)	EKM-04	Platform and data-appropriate encryption (e.g., AES-256) in open/validated formats and standard algorithms shall be required. Keys shall not be stored in the cloud (i.e., at the cloud provider in question), but maintained by the cloud consumer or trusted key management provider. Key management and key usage shall be separated duties.
5	Identity & Access Management (Audit Tools Access)	IAM-01	Access to, and use of, audit tools that interact with the organization's information systems shall be appropriately segregated and access restricted to prevent inappropriate disclosure and tampering of log data.
6	Identity & Access Management (Segregation of Duties)	IAM-05	User access policies and procedures shall be established, and supporting business processes and technical measures implemented, for restricting user access as per defined segregation of duties to address business risks associated with a user-role conflict of interest.
7	Identity & Access Management (User Access Authorization)	IAM-09	Provisioning user access (e.g., employees, contractors, customers (tenants), business partners, and/or supplier relationships) to data and organizationally-owned or managed (physical and virtual) applications, infrastructure systems, and network components shall be authorized by the organization's management prior to access being granted and appropriately restricted as per established policies and procedures. Upon request, provider shall inform customer (tenant) of this user access, especially if customer (tenant) data is used as part the service and/or customer (tenant) has some shared responsibility over implementation of control.

8	Identity & Access Management (User ID Credentials)	IAM-12	<p>Internal corporate or customer (tenant) user account credentials shall be restricted as per the following, ensuring appropriate identity, entitlement, and access management and in accordance with established policies and procedures:</p> <ul style="list-style-type: none"> • Identity trust verification and service-to-service application (API) and information processing interoperability (e.g., SSO and Federation) • Account credential lifecycle management from instantiation through revocation • Account credential and/or identity store minimization or reuse when feasible • Adherence to industry acceptable and/or regulatory compliant authentication, authorization, and accounting (AAA) rules (e.g., strong/multi-factor, expireable, non-shared authentication secrets)
9	Infrastructure & Virtualization Security Network Architecture	IVS-13	<p>Network architecture diagrams shall clearly identify high-risk environments and data flows that may have legal compliance impacts. Technical measures shall be implemented and shall apply defense-in-depth techniques (e.g., deep packet analysis, traffic throttling, and black-holing) for detection and timely response to network-based attacks associated with anomalous ingress or egress traffic patterns (e.g., MAC spoofing and ARP poisoning attacks) and/or distributed denial-of-service (DDoS) attacks.</p>

APPENDIX D: DESCRIPTION OF SELECTED SECURITY CONTROLS

This appendix describes the different security controls selected to mitigate the identified threats in the TSM application. The security controls descriptions are as provided by NIST [56] in the NVD database.

S/N	Security Control Family	Security Control ID	Security Control Name	Security Control Description
1	Access Control	AC-2	Account Management	See https://nvd.nist.gov/800-53/Rev4/control/AC-2
		AC-3	Access Enforcement	The information system enforces approved authorizations for logical access to information and system resources in accordance with applicable control policies
		AC-5	Separation of Duties	See https://nvd.nist.gov/800-53/Rev4/control/AC-5
		AC-6	Least Privilege	The organization employs the principle of least privilege, allowing only authorized accesses for users (or processes acting on behalf of users) which are necessary to accomplish assigned tasks in accordance with organizational missions and business functions
2	Contingency Planning	CP-9(6)	Information System Backup Redundant Secondary System	The organization accomplishes information system backup by maintaining a redundant secondary system that is not collocated with the primary system and that can be activated without loss of information or disruption to operations
3	Identification and Authentication	IA-2(13)	Identification and Authentication (Organizational users) Out-of-band authentication	The information system implements a defined out-of-band authentication under organization-defined conditions
		IA-5	Authenticator Management	See https://nvd.nist.gov/800-53/Rev4/control/IA-5
4	System and Services Acquisition	SA-13	Trustworthiness	See https://nvd.nist.gov/800-53/Rev4/control/SA-13
		SA-18	Tamper Resistance and Detection	The organization implements a tamper protection program for the information system, system component, or information system service
5	System and Communications Protection	SC-5	Denial of Service Protection	The information system protects against or limits the effects of several types of denial of service attacks by employing organization-defined safeguards
		SC-5(3)	Denial of Service Protection	See https://nvd.nist.gov/800-53/Rev4/control/SC-5

			Detection/Monitoring	
		SC-8	Transmission Confidentiality and Integrity	The information system protects the confidentiality and integrity of transmitted information
		SC-8(1)	Transmission Confidentiality and Integrity Cryptographic or Alternate Physical Protection	The information system implements cryptographic mechanism to prevent unauthorized disclosure of information, detect changes to information during transmission unless otherwise protected by organization-defined alternative physical safeguards
		SC-12	Cryptographic Key Establishment and Management	The organization establishes and manages cryptographic keys for required cryptography employed within the information system in accordance with organization-defined requirements for key generation, distribution, storage, access and destruction
		SC-13	Cryptographic Protection	The organization system implements organization-defined cryptographic uses and types of cryptography required for each use in accordance with applicable federal laws, executive orders, directives, policies, regulations, and standards
		SC-16	Transmission of Security Attributes	The information system associates organization-defined security attributes with information exchanged between information systems and between system components
		SC-23	Session Authenticity	The information system protects the authenticity of communications sessions
6	System and Information Integrity	SI-10	Information Input validation	The information system checks the validity of organization-defined information inputs

APPENDIX E: A SNIPPET OF THE TSM APPLICATION ONTOLOGY

This appendix presents a snippet of the TSM application ontology.

```
<?xml version="1.0"?>
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:owl="http://www.w3.org/2002/07/owl#"
  xmlns:to="http://www.ontologies.com/Ontology3125.owl#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema#">
  <owl:Ontology rdf:about="http://www.ontologies.com/Ontology3125.owl"/>
  <owl:Class rdf:about="http://www.ontologies.com/Ontology3125.owl#Security_Controls_Domain">
    <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
  </owl:Class>
  <owl:Class rdf:about="http://www.ontologies.com/Ontology3125.owl#TSMe_probe">
    <rdfs:subClassOf>
      <owl:Class rdf:about="http://www.ontologies.com/Ontology3125.owl#Sec_Monitoring_Probe"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:about="http://www.ontologies.com/Ontology3125.owl#Security_Metrics">
    <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
  </owl:Class>
  <owl:Class rdf:about="http://www.ontologies.com/Ontology3125.owl#Spoofing">
    <rdfs:subClassOf>
      <owl:Class rdf:about="http://www.ontologies.com/Ontology3125.owl#Threat"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:about="http://www.ontologies.com/Ontology3125.owl#Risk_Level">
    <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
  </owl:Class>
  <owl:Class rdf:about="http://www.ontologies.com/Ontology3125.owl#CSP">
    <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
  </owl:Class>
  <owl:Class rdf:about="http://www.ontologies.com/Ontology3125.owl#confidentiality">
    <rdfs:subClassOf>
      <owl:Class rdf:about="http://www.ontologies.com/Ontology3125.owl#Security_Objectives"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:about="http://www.ontologies.com/Ontology3125.owl#Threshold">
    <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
  </owl:Class>
</rdf:RDF>
```

```

</owl:Class>
<owl:Class rdf:about="http://www.ontologies.com/Ontology3125.owl#Access_control">
  <rdfs:subClassOf>
    <owl:Class rdf:about="http://www.ontologies.com/Ontology3125.owl#Security_Control"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:about="http://www.ontologies.com/Ontology3125.owl#MJP_probe">
  <rdfs:subClassOf>
    <owl:Class rdf:about="http://www.ontologies.com/Ontology3125.owl#Sec_Monitoring_Probe"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:about="http://www.ontologies.com/Ontology3125.owl#Hardware_Requirement">
  <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
</owl:Class>
<owl:Class rdf:about="http://www.ontologies.com/Ontology3125.owl#DB_probe">
  <rdfs:subClassOf>
    <owl:Class rdf:about="http://www.ontologies.com/Ontology3125.owl#Sec_Monitoring_Probe"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:about="http://www.ontologies.com/Ontology3125.owl#Elevation_of_Privileges">
  <rdfs:subClassOf>
    <owl:Class rdf:about="http://www.ontologies.com/Ontology3125.owl#Threat"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:about="http://www.ontologies.com/Ontology3125.owl#Tampering">
  <rdfs:subClassOf>
    <owl:Class rdf:about="http://www.ontologies.com/Ontology3125.owl#Threat"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:about="http://www.ontologies.com/Ontology3125.owl#Storage_asset">
  <rdfs:subClassOf>
    <owl:Class rdf:about="http://www.ontologies.com/Ontology3125.owl#Asset"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:about="http://www.ontologies.com/Ontology3125.owl#Infrastructure_and_virtualization_security">
  <rdfs:subClassOf rdf:resource="http://www.ontologies.com/Ontology3125.owl#Security_Controls_Domain"/>
</owl:Class>
<owl:Class rdf:about="http://www.ontologies.com/Ontology3125.owl#CEC_probe">
  <rdfs:subClassOf>
    <owl:Class rdf:about="http://www.ontologies.com/Ontology3125.owl#Sec_Monitoring_Probe"/>
  </rdfs:subClassOf>

```