Macrothink Institute™

Network Protocols and Algorithms
ISSN 1943-3581
2016, Vol. 8, No. 1

Security issues and threats that may affect the hybrid cloud of FINESCE

Julia Sánchez, Guiomar Corral, Ramon Martín de Pozuelo and Agustín Zaballos

Department of Engineering, La Salle-University Ramon Llull

Quatre Camins 30, 08022, Barcelona (Spain)

Tel: (+34) 932902400 E-mail: julias@salleurl.edu, guiomar@salleurl.edu, ramonmdp@salleurl.edu, zaballos@salleurl.edu

Received: December 14, 2015 Accepted: March 14, 2015 Published: March 31, 2016

DOI: 10.5296/npa.v8i1.8727 URL: http://dx.doi.org/10.5296/npa.v8i1.8727

Abstract

FINESCE is the *Smart Energy* use case project of the Future Internet Public Private Partnership Programme. It aims at defining an open infrastructure based on Information and Communications Technology (ICT) used to develop new solutions and applications in all fields of Future Internet related to the energy sector. To accomplish this goal a cloud-based environment is proposed, providing high scalability, fast provisioning, resilience and cost efficiency, while facilitating the deployment of applications and services for utilities.

The proposed solution for *Smart Energy* system encompasses *Cloud Computing* technologies taking advantage of the service delivery models that it provides (Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), Software-as-a-Service (SaaS)) over different cloud deployment solutions (Private, Public, Hybrid, Community). Therefore, it is necessary to study their implications, particularly with regard to security and data privacy, whether in transit or stored data, of the cloud solution chosen.

The present paper aims to gather basic security requirements in deploying a solution based on *Cloud Computing* highlighting issues in *hybrid clouds* because this is the deployment model used in *Smart Energy* use case. It also exposes attacks and vulnerabilities related to *Cloud Computing* to be considered for implementing a secure environment for FIDEV, the private platform implementation. Moreover, the security requirements for *Smart Energy* use case are defined. And, finally, the results of a security audit performed over the testbed platform that simulates a distributed storage solution for FINESCE project are presented.

Keywords: Cloud Computing, IaaS, SaaS, PaaS, cloud threats, cloud security, networking.

1. Introduction

Over the last decade, Smart Grids have led the revolution of the electrical grid transforming it in a set of automated and efficiently controlled processes by the incorporation of Information and Communications Technology (ICT). This fact has caused a great interest in continuing research on this topic. Some work done are focused on: provide a further insight qualitative analysis of the yet-unexplored communication requirements raised by the electrical distribution Smart Grid and propose solutions to allow faster and more reliable communication [1], apply unsupervised learning techniques to enhance the performance of data storage in Smart Grids [2], or provide solutions and results developed and tested in an IDEV-based environment, especially in the security domain and the security decisions about technologies used in the communications network [3]. These are some interesting lines of investigation about Smart Grids. However, the investigation line of this paper aims to provide a vision more focused in data security and privacy, whether in transit or stored data, when Cloud Computing is introduced as a key technology of the communications in the Smart Grid.

FINESCE: Future INtErnet Smart Utility ServiCEs [4] is the *Smart Energy* use case project of the second phase of Future Internet Public Private Partnership Programme (FI-PPP) [5] within European Commission's Seventh Framework Programme (FP7). Project's main objective is to contribute to the development of an open infrastructure based on ICT used to develop new solutions and applications in all fields of Future Internet in the energy sector. Through the collaboration of La Salle R&D, the FINESCE project will incorporate the concept of a virtual substation IEC61850 through FIDEV prototypes, which were initiated in the FP7 European project INTEGRIS: INTelligent Electrical GRId Sensor communications [6]. FIDEV is a platform built on commodity hardware, in which different software subsystems on top provide several communications and data concentrator functionalities. Among these new functionalities, a distributed storage system can be built over different interconnected FIDEVs, acting as a distributed Data Center (DC). These FIDEVs will be interconnected through the FIWARE Lab Cloud [7] using various Generic Enablers (GEs) [8], resulting in a distributed storage system based on a *hybrid cloud*. They will also provide seamless interaction between this FIDEVs-based private distributed storage system and the FIWARE Lab Cloud. In this sense, the system will be formed by a set of separated FIDEV's testbed devices (physical or virtualised) that will constitute a private cloud, plus public cloud storage capabilities by means of FIWARE Lab. Data can reside in any of the two clouds and be moved from one to another according to their owner's decision. Thus, this paper presents an alternative solution for electric companies with a more robust and scalable storage system thanks to the combination of cloud computing with their private DC infrastructure.

Several reasons could rise for the mobility of applications and data from the public cloud to local and viceversa. They range from application latency improvement to data confidentiality, including the low capacity of local resources. However, it will make DSO (Distribution System Operator) infrastructure ready to interact with the cloud in a very gradual incorporation of the novel functionalities while maintaining security policies in the public cloud environment. For this reason, Work Package 5 (WP5) Stream II inside the

FINESCE project presents a novel ICT infrastructure for *Smart Distribution Grids* that allows to flexible move Smart Grid data and applications from local systems to FIWARE Lab Cloud and protect them by the use of the security GEs developed in FIWARE.

Relying on the data network infrastructure of the utility, the FINESCE WP5 Stream II Trial interconnects different FIDEVs placed at different sites of ESB [9] in Ireland and FUNITEC – La Salle lab in Barcelona.

FIDEV is defined in FINESCE project as an upgrade of the communication part of IDEV devices defined in FP7 INTEGRIS integrating a set of GEs. These GEs will provide a secure interface with the distributed storage system and seamless interfaces to data management for the managers (in this case, a network manager from ESB). Among these new functionalities, it incorporates seamless interaction between FIDEVs private distributed storage system and the FIWARE Lab Cloud.

This paradigm of data being interchanged between public and private data clouds faces some security implications due to the lack of control about the infrastructure and applications that manage the information. It is a need to understand how cloud service delivery models (Infrastructure-as-a-Service, Platform-as-a-Service, Software-as-a-Service) are performed over hybrid clouds (deployment model used for *Smart Energy* use case) to draw a set of security requirements to design the solution. Moreover, it is also a need to thoroughly analyse some vulnerabilities, threats and problems that affect Cloud Computing technologies to minimize infrastructure risks since its conception. With this knowledge acquired it should be possible to design a proper infrastructure and perform a suitable trial to test if the design operates as expected.

The paper is organized as follows: Section 2 describes the *Smart Energy* use case infrastructure, Section 3 details the basic security requirements in Cloud Computing environments, Section 4 explains the security threads related to Cloud Computing, Section 5 illustrates the security issues in Cloud Computing, Section 6 defines the security requirements for the *Smart Energy* use case, Section 7 shows results of the security audit of the proposed system and, finally, Section 8 ends with the conclusions and further work.

2. Smart Energy Use Case Infrastructure Overview

FINESCE proposes to create an infrastructure based on a ***hybrid cloud*** environment to solve the needs of energy sector. The project solution requires processing and storing data of different sources like smart meters, electrical vehicles, client data, etc. This information will be stored and processed by FIDEVs located in public and private sites. The strengths of Cloud Computing are used to accomplish these needs and provide a robust system. Although there are different Cloud Computing deployment models (Public, Private, Community and Hybrid clouds) [10][11], a hybrid cloud deployment model is selected for the following reasons:

- The private cloud is used to ensure confidentiality of sensitive data stored like critical information about the electrical company.

- The public cloud is used to store non-sensitive data and historical measurement of smart meters, data of electric vehicle charging, etc. when FIDEVs are nearing its storage limit.
- There are applications susceptible to latency. The private cloud avoids vague and uncontrollable latency introduced by Internet.

"Fig. 1" shows the topology of the WP5 Stream II trial developed for *Smart Energy* use case. As it is shown, the trial interconnects different FIDEVs placed at different sites of ESB in Ireland and FUNITEC-La Salle lab in Barcelona. These two sites conform a private cloud each one, and the FIWARE Lab is considered the public cloud.

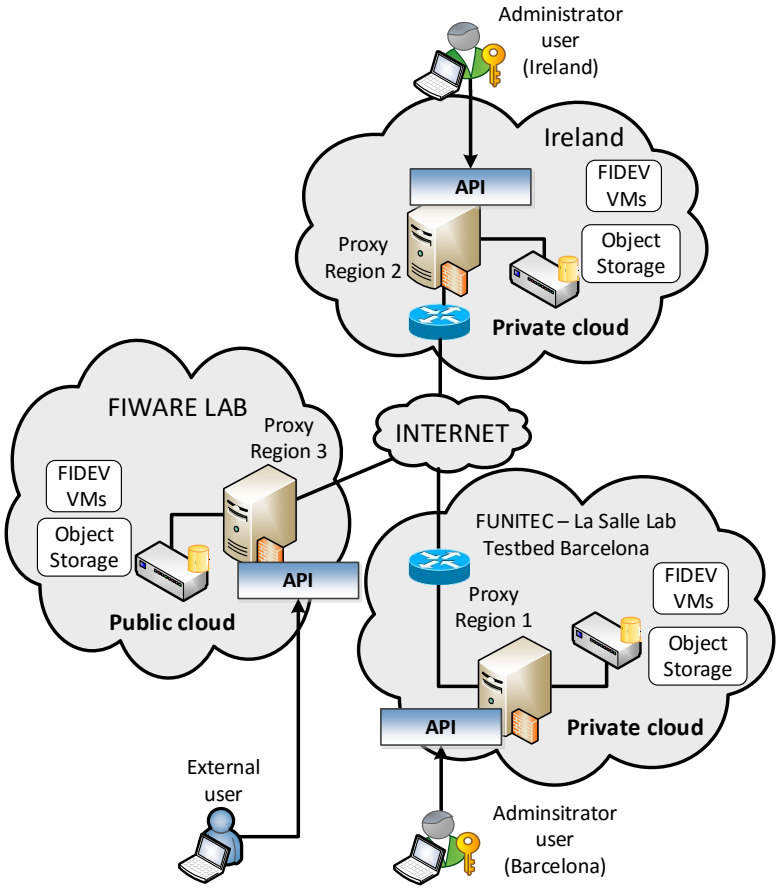

Figure 1. Generic overview of the WP5 Stream II Trial topology for Smart Energy use case.

Two GEs have been integrated into the FIDEV platforms, *Object Storage GE* and *Identity Management Keyrock GE*, contributing to the development of a distributed storage system and to provide security to hybrid cloud platform. The characteristics of the two selected GEs are the following:

- ***Object Storage GE*** [12]. Generic Enabler Implementation that provides robust, scalable object storage functionality based on *OpenStack Swift* [13]. The *OpenStack Swift API (Application Programming Interface)* provides a standardised mechanism to manipulate both the binary objects that are stored, and the hierarchy of containers in which they are organised. This RESTful API can be accessed from any client

technology that can communicate over HTTP.

- ***Identity Management – KeyRock*** [14]. Identity Management covers a number of aspects involving user's access to networks, services and applications, including secure and private authentication from users to devices, networks and services, authorization and trust management, user profile management, privacy-preserving disposition of personal data, Single Sign-On (SSO) to service domains and Identity Federation towards applications. Furthermore, Identity Management is used for authorising foreign services to access personal data stored in a secure environment.

Moreover, not only a deployment model is needed to deliver services or applications to process and store data in the Cloud Computing environment implemented with FIDEVs. It is also needed a delivery method (or a mix of them) implemented over the hybrid cloud. The service delivery models available are Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS) and Software-as-a-Service (SaaS) [10].

As NIST definition states [15], Cloud Computing is a model that enables on-demand access to a shared set of configurable computing resources that can be rapidly provisioned and released with minimal management effort or interaction by the Cloud Service Provider (CSP). This provisioning takes place via virtualization techniques in order to provide an efficient way to deliver the resources over the Internet. Cloud Computing solutions offer several benefits [16]. However, the fact that the management of some physical data and machines is implemented by CSPs (in the case of trial II by FIWARE Lab) keeping the customer (electric company) a minimum control over virtual machines, creates some concern and suspicion. How do the electric companies know their information in cloud is having no problem of availability and security? Is the information stored safely?

Cloud Computing solutions move some application software and databases of customers to large datacenters where the management and the services are not the same confidence when housed in an internal infrastructure. This paradigm poses security challenges that will be exposed in following sections.

3. Security Requirements

Before knowing security requirements related to the *Smart Energy* use case, the security requirements related to hybrid clouds and the service delivery models implemented over this deployment model have to be analyzed.

In a Cloud Computing environment there are many security risks depending on how CSPs deliver their services to customers. As shown in "Fig. 2", Cloud Computing should consider the risks associated with (1) the security in data storage, (2) the security in data transmission, (3) the application security and (4) the security related to third-party resources. Each delivery model of cloud services (IaaS, PaaS or SaaS) transparently provides a set of resources with the following characteristics [15]:

- **On-Demand Self Service**. Anyone can provision and consume cloud resources on their own.

- **Ubiquitous Network Access**. Access to cloud resources through public networks such as Internet.
- **Rapid Elasticity**. Ability to scale almost immediately if the need for resources increases.
- **Measured Service**. Monitoring of resource consumption in order to account for the costs of pay per use model.
- **Multi-tenancy**. A single instance of a software application serves multiple clients or tenants.

As presented in "Fig. 2", every service delivery model may be provided through different cloud deployment models (Private, Public, Hybrid and Community Clouds), which also have their own security risk by nature [15][16].

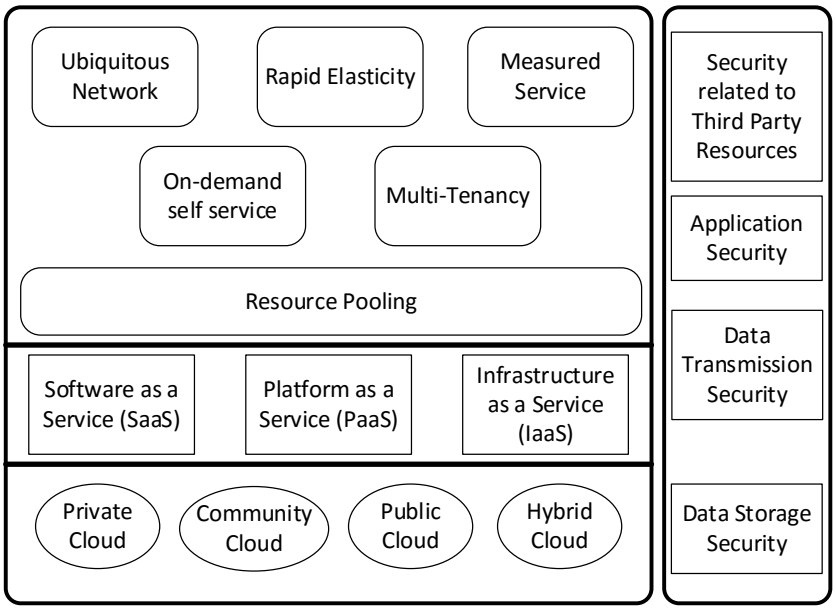

Figure 2. Security complexity in a cloud environment [16].

The characteristics of a hybrid cloud and its associated security risks are the following:

- It is managed by the organization or by third parties.
- Resources may be within or outside the customer premises.
- Access through Internet to multiple but limited distinct entities.
- It is more secure than public clouds where all depend on CSPs and SLAs (Service Level Agreements) have to be detailed and analysed consciously. But it is less secure than private clouds where customer data are inside the customer organization's own infrastructure (managed by the customer, security responsibilities easiest to identify).

Above this hybrid cloud, services could be provided through any of the service delivery models. The analysis of the characteristics and security issues of these models may help customer administrators to take decisions about the best model depending on the service:

- ***Infrastructure-as-a-Service*** completely abstracts the underlying hardware allocating physical resources on demand (typically in a virtualization environment), providing storage, networking and computing capabilities and allowing users to use infrastructure as a service. IaaS only provides basic security [10], including perimeter, such as firewalls, Intrusion Prevention Systems (IPS) and Intrusion Detection Systems (IDS). It also includes load balancing to provide more availability and VMM (Virtual Machine Monitors) to monitor the performance of virtual machines and provide isolation between them.
- ***Platform-as-a-Service*** allows customers to build their own applications by delivering a set of tools and development platforms that provide full life cycle without worrying about the underlying hardware and software. The cloud hosts SOA (Software Oriented Architecture) environments for hiding the underlying web elements. For this reason, and because the attackers are likely to attack visible code, a set of metrics are needed to measure quality and security of the encryption in the code written and to prevent the development of applications exposed to attacks [10].
- ***Software-as-a-Service*** delivers applications via Internet without installing software. This model improves operational efficiency and reduces costs for customers. Many security problems related to the basic components of SaaS [17] applications are known (shown in next sections). From the customer's perspective, it is difficult to understand if data are secure and if applications are available at all times due to the lack of visibility into how data is stored and applications are deployed. The SOA challenges [10] in this model are focused on how to preserve or enhance the security previously provided by traditional hosting systems.

There is a compromise between system control, data and cost efficiency as shown in “Fig. 3”. The less control by the customer, the lower the costs of implementing business applications. This implies a loss of confidence because security depends largely on the CSP. Therefore the customer is forced to rely on that security extends along the entire stack as it has no other alternative. To define the conditions of service delivery and enhance this confidence, SLAs are established between customers and their suppliers to ensure the quality, availability, reliability and performance of the resources provided [18].

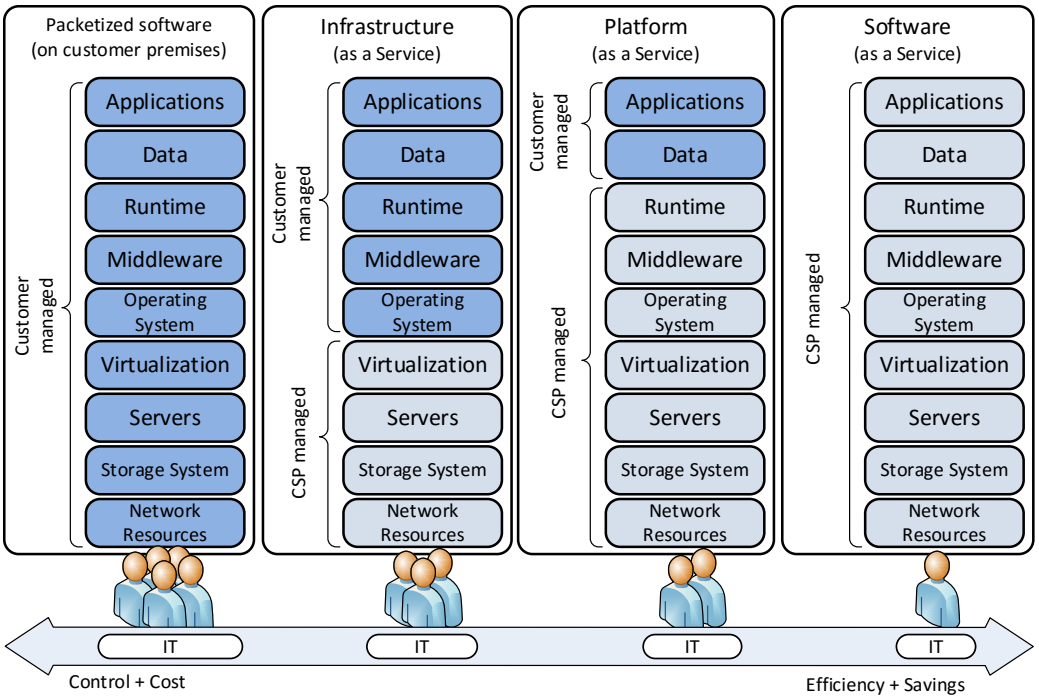

Figure 3. Cloud service delivery models.

Once analyzed each service delivery model and the underlying possible deployment models, a set of basic security requirements [10] that have to be accomplished can be defined:

- Identification and Authentication
- Authorization
- Confidentiality
- Integrity
- Non-repudiation
- Availability

"Table 1" summarizes these basic security requirements for each service delivery model depending on the underlying deployment model. In a hybrid cloud environment it is important maintaining data integrity in transit and data stored. Moreover, if the delivery is through a SaaS application there are more security requirements to accomplish.

Each deployment model has its own specific problems and security issues. CSPs, customers and organizations must consider several factors, including the available budget, the purpose of cloud and security requirements deciding on a specific model.

Table 1. Security requirements for service delivery models and different deployment models [10].

Security Requirements	Cloud Deployment Models								
	Public Cloud			*Private and Community Clouds*			*Hybrid Cloud*		
	IaaS	*PaaS*	*SaaS*	*IaaS*	*PaaS*	*SaaS*	*IaaS*	*PaaS*	*SaaS*
Identification and Authentication	**Yes**	No	**Yes**	**Yes**	No	**Yes**	No	No	**Yes**
Authorization	**Yes**	**Yes**	**Yes**	No	No	**Yes**	No	No	**Yes**
Confidentiality	No	No	**Yes**	No	**Yes**	**Yes**	No	No	**Yes**
Integrity	**Yes**	No	**Yes**	No	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**
Non-Repudiation	No	No	**Yes**	No	No	**Yes**	No	No	No
Availability	Yes	**Yes**	No	**Yes**	**Yes**	**Yes**	No	No	No

In the *Smart Energy* use case implemented over a hybrid cloud, it is relevant to consider the security requirements related to PaaS and SaaS because the applications defined for the trial II are implemented in these service delivery models. In fact, a SaaS application for data storage is performed in FIWARE Lab (CSP in a public cloud) and a PaaS application for data storage and processing information is implemented in private sites. Due to the dependency of the FIWARE Lab, some SLAs may need to be established with this CSP to ensure the availability and security of the services delivered. Also, for applications consumed as PaaS, some metrics must be defined or some tests have to be performed to ensure that the system is protected. In section 7, the tests performed to check out system robustness are shown.

4. Security Threats in Cloud Computing

The biggest problem that faces cloud computing is to ensure *confidentiality* and *integrity of data* and *availability of services*. A central component for managing the risks associated with this problem is to understand the nature of security threats in the cloud. With a comprehensive understanding of these threats the solution proposed for the *Smart Energy* use case may be more robust and secure. In [19] the CSA (Cloud Security Alliance) presented a report aimed at highlighting the hazards associated with the shared and low demand nature of Cloud Computing with a brief explanation of each one and the implications that have to be considered when a system is exposed to these threats.

Once threats are known, it is needed to seek countermeasures to minimize risks. “Table 2” describes possible solutions and the risks associated with each threat catalogued with risk models like CIANA (Confidentiality, Integrity, Availability, Non-repudiation, Authentication), because it conforms to the basic security requirements specified in the previous section and STRIDE (Spoofing identity, Tampering with data, Repudiation, Information disclosure, Denial of Service, Elevation of Privilege), because it is related to vulnerabilities affecting the cloud. In addition, the threats are related to the corresponding CSA domains described in "*Security Guidance for Critical Areas on focus in Cloud Computing*" [20] where the best practices to consider "security by design" are depicted.

Table 2. Notorious threats to cloud and its countermeasures.

Threat	Risk Analysis [19]	Countermeasures [11]	CSA [20] Domains
Data Breaches	• CIANA: Confidentiality • STRIDE: Information disclosure	• Isolation of virtual machines and stored data • Full erase data sessions before delivering data to new users to prevent data leakage • Backup data offline	5, 10, 12, 13
Data Loss	• CIANA: Availability, Non-repudiation • STRIDE: Repudiation, Denial of Service	• Use DLP tools (*Data Loss Prevention*)	5, 10, 12, 13
Account or Service Traffic Hijacking	• CIANA: Authentication, Integrity, Confidentiality, Non-repudiation, Availability • STRIDE: Tampering with data, Repudiation, Information disclosure, Elevation of Privilege, Spoofing identity	• Do not share account credentials among employees • Double authentication techniques • Good definition of SLAs	2, 5, 7, 9, 11, 12
Insecure Interfaces and APIs	• CIANA: Authentication, Integrity, Confidentiality • STRIDE: Tampering with data, Repudiation, Information disclosure, Elevation of Privilege	• Evaluate APIs before using it • CSP: Strong access controls, authentication and encrypted transmission	5, 6, 9, 10, 11, 12
Denial of Service (DoS)	• CIANA: Availability • STRIDE: Denial of Service	• Use intrusion detection and prevention systems (IDS, IPS)	8, 9, 10, 13, 14
Malicious Insiders	• STRIDE: Spoofing identity, Tampering with data, Information disclosure	• User access level controls	2, 5, 11, 12
Abuse of Cloud Services	• CIANA: N/A • STRIDE: N/A	• Use registration and validation processes before giving customers access to the cloud • Passive monitoring to ensure that a user does not affect others	2, 9
Insufficient Due Diligence	• STRIDE: All	• Security of data, combined with risk transfer in the form of insurance coverage and acceptance of risk taking by CSPs	2, 3, 8, 9
Shared Technology Vulnerabilities	• STRIDE: Information disclosure, Elevation of Privilege	• Strong compartmentalization between users • Strong authentication mechanisms • SLAs that include remedy	1, 5, 11, 12, 13

On the other hand, along with the knowledge of hazards associated with the cloud, it is also of vital interest to know the limitations of security problems to which a customer is exposed to minimize risks. Due to that purpose, Gartner [21] proposes seven specific areas on which customers should collect information before selecting a CSP: (1) What types of users have privileged access and how they are hired, (2) Regulatory compliance and Certifications needed, (3) Preserving privacy requirements regardless of the location of the data, (4) Securely data isolation between customers, (5) Availability of data recovery in case of disaster, (6) Support for research and extraction of evidence (due if a crime is incurred), (7) Long-term viability, availability of data regardless of whether the CSP breaks or another company takes over the CSP.

After detailing the most relevant threats and risks related with cloud, it is more feasible to design a system for the *Smart Energy* use case that includes security from conception. On the other hand, knowing the system depends on the public cloud of FIWARE, it is necessary to ensure the seven areas described above reaching some agreements represented through SLAs with this CSP.

5. Security Issues in Cloud Computing

The aim of this section is to highlight the problems directly associated with each service delivery model because is crucial knowing the issues related to IaaS, PaaS and SaaS to develop a secure and robust solution for *Smart Energy* use case depending on the services that FIDEVs deliver.

First, in the **IaaS model** it should be noted that there are no security breaches in the virtualization manager. The other important factor is the reliability of the data stored within the hardware vendor. Due to the increasing virtualization of "everything", it becomes an aspect of great interest how the data owner (customer) retains ultimate control over it regardless of location. IaaS is prone to varying degrees of security issues based on the deployment model and in hybrid clouds like the *Smart Energy* use case, it is important to take into account the following aspects to implement accurate solutions [10]:

- The management of the infrastructure is carried out by the own organization and also third-parties are involved.
- The infrastructure is on organization premises or is owned by third-parties.
- Infrastructure location is on organization premises or in third-party facilities.
- Access and use could be reliable and unreliable

If an IaaS application is developed in the future, the FIDEVs or other new nodes located in private cloud are more secured than storage nodes located in public cloud. The internal infrastructure in Barcelona and Ireland is controlled and managed by administrators directly related to the electric company but in the public cloud the performance of the solution depends on the management of FIWARE administrators.

Second, in **PaaS model**, the CSP can give some control to application developers on top of the platform. But any security is given below the application level, such as IPS at network and host levels. This can concern the CSP, who should pay special attention to offer strong guarantees that data will remain inaccessible between applications. For *Smart Energy* use case, PaaS is implemented for FIDEVs in private clouds located in Barcelona and Ireland. These locations must face security needs implementing IPS, DoS (Denial of Service) detection and virtual machines isolation.

Last but not least, in **SaaS model**, the customer depends on the provider who applies appropriate security measures. User data is stored in the SaaS provider datacenter, along with the data of other users. On the other hand, if the SaaS provider is using a computer service in the public cloud, user data can be stored along with the data of other SaaS applications

unrelated. The CSP may also replicate data across multiple locations through various countries in order to provide high availability. This whole operation of SaaS causes the CSP put all the effort into ensuring the user the privacy of its data about other users and to ensure that user that is being implemented appropriate security measures and that the application will be available when requested. “Table 3” [16] contains the security problems associated with the SaaS model showing a brief definition of the environment that affects each problem and possible solutions to be applied.

Table 3. Security Problems in SaaS environments.

Security Problem	Environment definition	Solutions and/or Recommendations
Data Security	• User data out customer premises • CSP: Additional control to ensure security and prevent breaches by application vulnerabilities or malicious employees	• Robust encryption and authorization techniques • Administrators without access to client instances nor OS (Operating System) • Routinely register and audit access
Network Security	• Sensitive user data processed by the application and stored in the CSP	• Ensure data flow through the network to prevent leakage of sensitive information • Attack protection against MITM (*Man-in-the-middle*), IP spoofing, port scanning, packet sniffing, etc. • Encryption techniques of network traffic, SSL and TLS
Data Location	• Uncertainty about location of user data stored by the customer • Law enforcement and privacy of data may vary between countries • Jurisdiction of the data when an investigation occurs	• The user must make sure how the laws apply
Data Integrity	• Typically, ACID transactions to ensure data integrity (Atomicity, Consistency, Isolation and Durability) • In distributed systems, maintaining proper data management and fail-safe • SOA Environments use SOAP and REST • HTTP does not allow guaranteed delivery	• Centralized management of transactions • Implement mechanisms for guaranteed delivery at API level • WS-Transaction and WS-Reliability for integrity
Data Segregation	• Multi-tenancy, data from several users at the same location • Intrusion problems between users	• CSP: Guarantee to maintain physical and application limits
Data Access	• Regarding security policies provided to users to access data	• Customer must set data access security policies • Ensure the compliance of these policies by the CSP to prevent unauthorized intrusion • CSP must ensure boundaries between tenants

Security Problem	Environment definition	Solutions and/or Recommendations
Authentication and Authorization	• LDAP servers commonly used to access corporations and Active Directory to access SMB • Software user management hosted out customer premises, user credentials stored in CSP databases	• Customer must remember delete/deactivate or create/enable accounts of employees who leave the company or new employees • If it is necessary for security, CSP may delegate to LDAP/AD authentication company
Data Confidentiality	• Exchange or storage of data on remote servers owned or operated by third parties and accessible via the Internet or other connections	• Establish security policy and SLAs with CSP adapted to the requirements of confidentiality and privacy of the user • Maintain knowledge of: (1) Rights applicable to privacy according to data submitted to the CSP, (2) Obligations of CSP regarding privacy and confidentiality as location data, (3) Legality associated with the data by location
Web Application Security	• Changing SaaS application software transparently to the user • If the software is not programmed correctly, the data behind the SaaS application and the application itself will be at risk	• Check that the SaaS application is not susceptible to the most relevant vulnerabilities identified in the OWASP Top 10 Project [22]
Data Breaches	• Sensitive customer data stored in cloud	• Prohibit direct access to CSP employees databases • Control and monitor access to any part of the cloud environment to prevent leaks of sensitive information
Virtualization	• It is assumed that different instances in the same virtual machine are isolated between them and from virtualization tasks	• Ensure isolation • Use VMMs at root level, without privileges that allow guests access to the host system
Availability	• SaaS application developed in multi-tier architecture with load balanced instances running on multiple servers is assumed	• SaaS application developed with resistance to HW/SW faults and DoS attacks • Have a plan for business continuity and disaster recovery
Backups	• It is assumed that the CSP conducts regular copies of sensitive customer data to facilitate rapid disaster recovery	• Use robust encryption schemes to protect backups and prevent information leaks
Identity Management (IdM)	• SaaS application system feature that controls access to resources by placing restrictions on established identities	• Maintain a robust identity management system • There are three perspectives to consider in implementing IdM: pure identity, user access (log-on) and service

Storage nodes located at FIWARE Lab are consumed as SaaS applications. So, it is necessary to define security requirements for *Smart Energy* use case that try to avoid the problems presented in “Table 3”.

6. Security requirements for Smart Energy Use Case

Once analyzed the basic security requirements in cloud environments and the threats and vulnerabilities that affect Cloud Computing, it is possible to define the security requirements for the *Smart Energy* use case hybrid cloud. Before defining the security requirements related to Smart Energy use case, it is necessary to show a bit more accurate description of the solution proposed for the project mentioned.

As it is mentioned in section 2 and shown in "Fig. 1", the *Smart Energy* use case topology is based in a ***hybrid cloud*** with different areas or regions interconnected via Internet. In local areas like Barcelona and Ireland, a system of private clouds is deployed while the FIWARE Lab, the testing platform of FIWARE project, represents the part of the public cloud. In each region, FIDEV devices are located and act as virtual substations collecting data of devices connected to the grid (smart metering, charging electric vehicle points, etc.). FIDEVs are based on *OpenStack Object Storage* functionality to provide data storage and the necessary APIs to interface with them. These APIs are based on FIWARE GE defined in FIWARE project and are used as follows:

- In the public cloud the ***Object Storage GE*** [12] is used for data storage and it is consumed as a SaaS application of FIWARE Lab platform. When FIDEVs are reaching its maximum storage capacity, non-sensitive data are uploaded to FIWARE Lab through *Object Storage GE CDMI* (Cloud Data Management Interface).
- In the private cloud, local instances of the *Object Storage GE* have been deployed with the specifications provided by FIWARE. In this way, the local result is an object storage system based on *Openstack Swift* module [13] and an identity management system based on *Openstack Keystone* module [23]. In this private cloud, resources are consumed in PaaS mode. *Object Storage* containers in proxies (locations that storage files) are synchronized between them with Rsync [24] to provide a distributed storage system.

As "Table 1" summarizes, the basic security requirements for a hybrid cloud must comply with the characteristics of *Identification, Authentication, Authorization, Confidentiality and Integrity* for SaaS environments and *Integrity* for PaaS environments. All these basic security features are met as follows:

- **Integrity**. Both, *Object Storage* and *Swift* are responsible for storing data with integrity.
- **Identification, Authentication and Authorization**. When a user wants to perform operations on data from the private cloud (upload, download, encrypt, decrypt) first authenticates against *Keystone* checking credentials and if that user is authorized in the storage application requested, the access is permitted.
- **Authorization**. The data transfer operations to the cloud of FIWARE are made from local FIDEVs, which imply that the user is previously authenticated for this operation.

- **Confidentiality**. The data is kept confidential by a user-defined key. A hash is generated using SHA-256 and the hash is used to encrypt data with AES-256.

However, although a secure system is designed and the result should be a robust system implemented, it is necessary to take into account that security issues can occur and affect to the proposed infrastructure for FINESCE project in WP5 Stream II.

In "Table 4" the most important security issues considered for this project are presented. The aim is to establish an order of implementation priorities regarding the security aspects.

Table 4. Security Requirements for Smart Energy Use Case.

Security Issue		Problem Description	Priority	Reason of priority value
Data Security	**Data Leakage**	Data is stolen and delivered without permission of the proprietary. It affects confidentiality.	5	If a malicious user can access the system, user stored data could be compromised. This fact could derive in legal problems.
	Data Forgery	Data is modified by a malicious user and not detected. It affects integrity and maybe confidentiality.	6	To erase or modify data it is first needed a granted access to the system. Once the access is accomplished, if notifications of changes are not considered, a malicious user could modify user stored data.
	Data Lost	Data is erased by a malicious user or a human error. It affects confidentiality and availability.	7	If a backup system is maintained, this could be an important but not critical problem since data could be restored.
Network Security	**Data Transaction**	Data is delivered through the network and could be visible to malicious users if it is not encrypted. It could also not be transmitted correctly due to DoS (Denial of Service) attacks. It depends on the sensibility of the data transmitted that this issue becomes more critical. It affects availability and confidentiality of the services.	1	Because it is not necessary to access the system to obtain data under these circumstances, it is considered that the most important aspect is that data transactions (data in transit) are encrypted.
	Commands execution	Many applications that can reside in FIDEVs could be sensitive to latency. A DoS attack to the network resources could affect its performance. It affects availability of the services.	8	Network resources have to be controlled because the access to data stored and applications in FIDEVs depends on them. It is considered that network will be designed to detect DoS attacks and avoid latency problems.
Authentication		Access to FIDEVs and data storage has to be controlled and tracked to avoid wrong usage. It affects confidentiality, integrity and availability if a malicious user gets a user with rights granted.	2	It is very important to maintain control over the users that access data stored in FIDEVs and track the actions this users perform to avoid problems with data stored and FIDEVs functionality. If wrong usage is detected and users are authenticated, the system can isolate the problematic user to avoid damage.

Security Issue	Problem Description	Priority	Reason of priority value
Authorization	Not all users have the same authorization policies to different zones, resources or data stored. Admin users, privileged users, guest users and third party users must be catalogued with different authorization rules. It affects confidentiality, integrity and availability if a good policy is not implemented.	3	It is important to maintain isolated rights to access resources in FIDEVs because the system could have third-party users, guests/clients, administrators… and not all should have complete access. The system could be modified by users without complete knowledge of what they are doing or by malicious users if a good authorization policy is not applied.
Identity Management	The way to maintain a good connection between users and authorization rules is implementing a robust IdM. If user policies are wrong assigned or not controlled, this issue can affect confidentiality, integrity and availability.	4	Necessary to map users with their respective authorization rules and to maintain control over granted access to the system.

7. Security Audit

The main objective of the security audit is to check the vulnerabilities that may have systems in FINESCE environment based on FIDEVs to identify potential threats and minimize the risk of exposure of data processed and the infrastructure itself. By means of the security audit performance it is possible to verify the minimum security requirements needed as “Table 4” establishes and the good operation of the code implemented in APIs.

The security analysis of the infrastructure is done from the point of view of *Ethical Hacking*. Specific operation tools are used to check the exposure of the system and detect all possible vulnerabilities before the system is compromised. These tools can be found in several Linux distributions and other specific systems for security analysis. In this project *Kali Linux by Offensive Security* [25] has been used.

Attacks on any system which is exposed may be contained both inside and outside the infrastructure to protect. The environment used to test the infrastructure proposed for *Smart Energy* use case is presented in “Fig. 4”.

The security audit performed is intended to recognize vulnerabilities of the system, maybe due to weaknesses in the components used or maybe due to poor implementation of code inside each component.

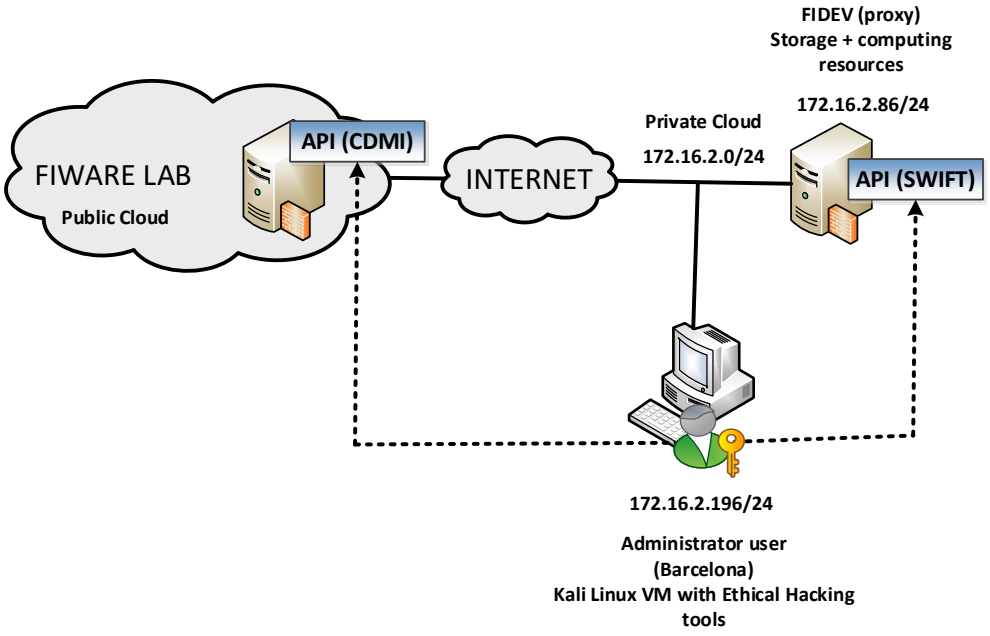

Figure 4. Testing environment for Smart Energy use case.

7.1 Technical Requirements and Processes performed by the solution

Before performing the security audit, it is necessary to collect some information about the technical requirements of the systems used and the processes established to authenticate users and storage information.

7.1.1 Technical Requirements

The testing environment has been developed over a virtualized platform using VMWare ESXi. It has been implemented an environment with one FIDEV or proxy node and two storage nodes.

The FIDEV node has the following technical characteristics:

- **Hardware**: 1 processor, 2 GB RAM, 5GB HHDD.
- **Operating System**: Ubuntu 14.04 LTS (64-bit version) with all packages uploaded.
- **Main components**: *Swift and Keystone Openstack* modules, MariaDB, RabbitMQ, *Network Transport Protocol* (NTP).

The storage nodes have the following technical characteristics:

- **Hardware**: 1 processor, 512MB RAM, 5GB HHDD.
- **Operating System**: Ubuntu 14.04 LTS (64-bit version) with all packages uploaded.
- **Main components**: *Swift Openstack* module, RabbitMQ, *Network Transport Protocol* (NTP).

With this information is possible to perform a search on CVE (Common Vulnerabilities and Exposures) databases [26][27][28] to check the vulnerabilities known associated to the components used. In section 7.2.2 is presented a sum of the vulnerabilities that may affect the system and the possible countermeasures are exposed.

7.1.2 Authentication Process

In order to ensure that the user can store data in the private cloud, the authentication proxy (FIDEV) will first validate that the user has an *Object Storage* application membership in FIWARE *IdM Keyrock*. Once this validation has succeeded, then it will authenticate against *Keystone*. The token used for validating the user will be keystone's one.

Different roles will be applied depending on the role on the *Keyrock*'s application role (reseller/purchaser).

In the authentication process, the user sends its username and password to the proxy. Then, the proxy sends these credentials to FIWARE Lab's *Keyrock* and gets a token. With this token the proxy can ask again to FIWARE Lab what applications the user has. If one of them is *Object Storage*, the proxy requests a token to *Keystone* and then obtains the final token that authenticates the user. This token will be bundled in the response to the user so that after this process the user knows the token. When the authentication proxy knows the token, it will store it in *memcache* so that for future requests (while the token is valid) it will optimize the process avoiding the validation process against FIWARE Lab that introduces lots of latency.

Some kind of federated authentication is implemented so that when someone wants to access a file that is stored in the private cloud, the same credentials and token can be reused with the public cloud one.

7.1.3 Storage Process

When a user wants to perform some action, this user must first authenticate. Once the credentials are validated, the user can interact with the proxy node via APIs to try to perform any action allowed in the distributed storage system. The actions allowed are the following:

- **List data objects of a user and container.** List all the public data objects stored by the utility in a specific directory of the hybrid cloud storage platform.
- **Upload data object.** Upload a data object to the public or private data storage. User should specify the name of the object and also in which container wants to be stored.
- **Download data object.** Download a data object from the public or private data storage. User should specify the name of the object and also in which container it is stored.
- **Create a new data container**. Create a data container in the public or private data storage. User should specify the name of the object and also in which container it is stored.

- **Delete data object**. Delete a data object or a container from the public or private data storage. User should specify the name of the object and also in which container it is stored.

It is important to know the actions allowed because their performance is evaluated in the security audit to check the APIs implementation.

Among FIDEVs (both public and private cloud) information can be exchanged using the APIs thus creating the distributed storage system. Moreover, when FIDEVs in private cloud are reaching their maximum storage capacity, non-sensitive data is uploaded to FIWARE Lab by *CDMI Object Storage GE*. NTP is used to synchronize proxy and storage nodes because is needed for the good implementation of the distributed storage system.

When a user uploads a file to the distributed storage system he/she has to decide if the file is uploaded encrypted or in plain text. Due to encryption can be used, the following main characteristics have to be taken into account to perform a good auditing:

- The user provides the key to encrypt data. The user is responsible of providing a strong key to encrypt the file that is uploading.
- The file is uniquely stored encrypted. If the user loses the key, there's no way to recover the original file.
- The algorithms used in order to encrypt are SHA-256 and AES-256. When the user sends the key, the key gets hashed using SHA-256 so that an input of any length provides a 256 bit-length encrypting key.
- The uploaded or downloaded data gets encrypted or decrypted using the hash of the provided key. If no password is provided, the file gets stored in plain text. If the provided key is incorrect, the system will try to decrypt the file and some output will be provided because the system can't know if the key is correct or not, so the user will download something that isn't what was uploaded.
- The only feasible way of encrypting data is using CDMI API.

7.2 Security Audit Performance

The process of performing a security audit by means of *Ethical Hacking* is based in performing some penetration tests (*pentesting*). To make a good *pentesting* process there are a few steps to do. These steps are the same that an outsider attacker would do. Every step provides detail to the step that follows. The five steps are:

- **Reconnaissance.** Identify and document as much information about the target as possible.
- **Scanning.** Scan the target network and information system. All these information will be used to exploit the target.
- **Exploitation.** Get into the system using system vulnerabilities and proven techniques.
- **Maintaining Access.** Once the system is exploited, backdoors and rootkits are left on the system to allow access in the future.

- **Reporting.** Detailed report to explain each step in the hacking process, vulnerabilities exploited, and systems that were actually compromised.

Taking into account these five steps, the security audit performed to the *Smart Energy* use case testbed follows the step by step process described in "Fig. 5". The process described is adapted to the needs of the test environment and focused in discovering vulnerabilities of the system audited. For this reason, the *Maintaining Access phase* is not performed and neither backdoors nor rootkits are left on the system to allow access in the future.

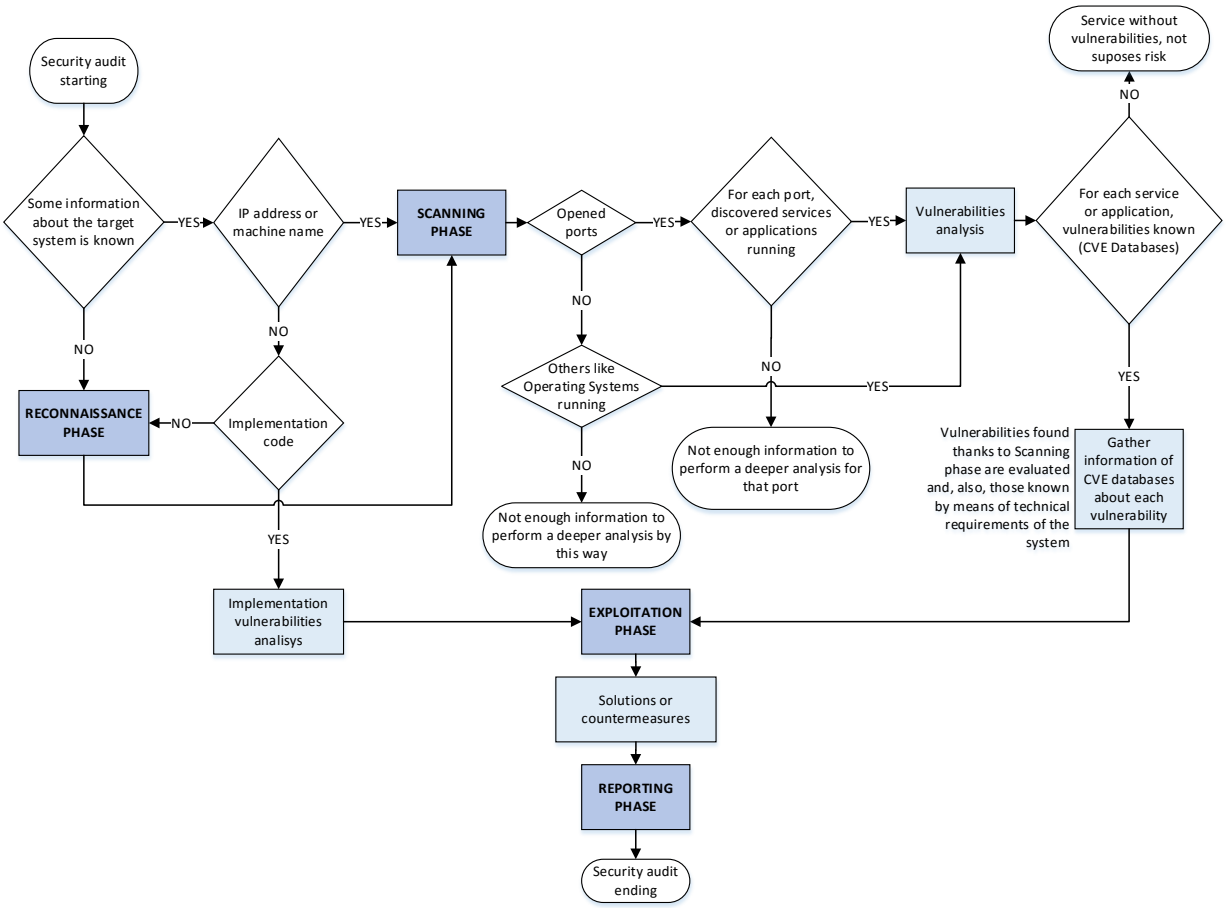

Figure 5. Security audit process.

The security analysis performed by means of the process described above is based on the following aspects:

- The type of *Ethical Hacking* analysis performed is a Gray Box testing because it is known some information about the components used like *Openstack Swift* and *Keystone*. Also, FIWARE *Generic Enablers* like *Keyrock* and *Object Storage* are used. Moreover, a static IP address is assigned to FIDEVs and IPs does not have to be discovered. *Reconnaissance phase* is not performed in this analysis because the information needed about the target system is known.
- First step done is the *Scanning phase*. Ports opened and services running are discovered.
- Next step performed is the searching of vulnerabilities associated to components

detailed in section 7.1.1 and services that are running in the system and discovered in the *Scanning phase*.

- The following step analyzes the code to find implementation vulnerabilities associated to authentication methods, file storage operation and creation of containers to storing data.
- Then, a set of tests are performed to check if the system is enough robust against some feasible attacks and vulnerabilities. This is the *Exploitation phase*.
- Finally, some solutions or countermeasures for security problems are provided and the processes, tests and results are reported to conclude the security audit (*Reporting phase*).

Moreover, when a problem has been reported there are different criteria to punctuate its severity:

- When the vulnerability is one of the CVE database their punctuation is used.
- In the case it is known that there is a vulnerability but the attack cannot be performed for any reason or simply it is an implementation vulnerability, a custom punctuation is calculated with NVD CVSS calculator v2 (National Vulnerability Database Common Vulnerability Scoring System) [29] establishing decisions about the impact of the vulnerability and valuating each metric that the calculator uses to obtain the CVSS base score [30].

7.2.1. Scanning Phase

First of all, the host where is implemented the FIDEV is scanned to see which ports are opened and which is its fingerprint to know more about the infrastructure. With **Nmap** tool [31], it has been discovered that the system uses **Python 2.7.6** scripts and it has ports **22, 443, 873, 6000, 6001 and 6002 opened**. The host uses **Ubuntu** and kernel **Linux 3.X**. “Table 5” sums the results of the scanning.

Table 5. Results of the scanning with Nmap tool.

Port	Status	Service	Version
22/tcp	Open	ssh	Version 2.0
443/tcp	Open	ssl/http	Apache httpd 2.4.7
873/tcp	Open	Rsync	Versión 3.1
6000/tcp	Open	¿?	¿?
6001/tcp	Open	¿?	¿?
6002/tcp	Open	¿?	¿?

Knowing that it has port 443 opened it is possible to discover the cipher and openssl features that the system uses. To do this type of scans some tools can be used, like **SSLScan** or **SSLyze**. But it is used **TLSSLed** (from https://github.com/drwetter/testssl.sh), because it gives a large amount of information about SSL. The relevant results about the execution of the script are presented in below.

- OpenSSL version: 1.0.1e
- SSL versions supported: SSLv3, TLSv1.0, TLSv1.1, TLSv1.2
- Cyphers: Several robust methods to cypher based on AES.

These results will be detailed in next sections, but coming up next it is possible to see if an outsider attacker can know which principal software are in the host a part of Rsync, SSH and SSL. With an easy search in Google using "*python 2.7.6 port 22 443 6000 6001 6002*" the findings show *Openstack* and their modules: *Keystone* and *Swift*. As a Gray box audit is performed it is already known that this system uses *Openstack* and *Swift* module. *Swift* is the service that is running under 6000, 6001 and 6002 ports. Port 22 is used to access remotely between devices, port 443 is used in secure transmissions between FIDEVs located in private sites and port 873 is used to synchronize storage nodes with Rsync.

Moreover, Nessus has been used to analyze the host but the results obtained only have given information extra that have already been known in the scans with Nmap.

7.2.2 Known Vulnerabilities

With the information gathered about the system in section 7.1.1 and the results of the scans performed, it is possible to search in CVE databases the vulnerabilities those components or services running have. "Table 6" sums the most critical vulnerabilities found with the *type of vulnerability, CVSS base score* obtained and its related *CVSS vector.* Only components or services with vulnerabilities are reported.

Table 6. Known vulnerabilities of the system.

Component or Service	**Vulnerability**	***Type***	***CVSS Vector***	***CVSS Base Score***
Swift v1.0	CVE-2012-4406	Code Execution	(AV:N/AC:L/Au:N/C:P/I:P/A:P)	7.5
	CVE-2013-4155	Denial of Service, Overflow	(AV:N/AC:L/Au:S/C:N/I:N/A:P)	4.0
	CVE-2013-6396	Obtain information	(AV:N/AC:M/Au:N/C:P/I:P/A:N)	5.8
Keystone API v2.0	CVE-2014-0204	Obtain Priviledges	(AV:N/AC:L/Au:S/C:P/I:P/A:P)	6.5
	CVE-2013-4222	Not defined	(AV:N/AC:L/Au:S/C:P/I:P/A:P)	6.5
	CVE-2014-3520	Not defined	(AV:N/AC:M/Au:S/C:P/I:P/A:P)	6.0
Python 2.7.6	CVE-2014-1912	Code Execution and Overflow	(AV:N/AC:L/Au:N/C:P/I:P/A:P)	7.5
	CVE-2014-7185	Obtain information and Overflow	(AV:N/AC:L/Au:N/C:P/I:N/A:P)	6.4
SSH v2.0	CVE-1999-1029	Not defined	(AV:N/AC:L/Au:N/C:P/I:P/A:P)	7.5
	CVE-2002-1715	Bypass of a restriction or similar	(AV:L/AC:L/Au:N/C:C/I:C/A:C)	7.2
Apache 2.4.7	CVE-2013-6438	Denial of Service	.(AV:N/AC:L/Au:N/C:N/I:N/A:P)	5.0
	CVE-2014-0098	Denial of Service	(AV:N/AC:L/Au:N/C:N/I:N/A:P)	5.0
	CVE-2014-0226	Denial of Service, Code Execution, Overflow, obtain information	(AV:N/AC:M/Au:N/C:P/I:P/A:P)	6.8
	CVE-2014-0231	Denial of Service	(AV:N/AC:L/Au:N/C:N/I:N/A:P)	5.0
OpenSSL 1.0.1e	CVE-2015-0292	Denial of Service, Memory Corruption, Overflow	(AV:N/AC:L/Au:N/C:P/I:P/A:P)	7.5

Component or Service	Vulnerability	*Type*	*CVSS Vector*	*CVSS Base Score*
	CVE-2014-8176	Denial of Service, Memory Corruption	(AV:N/AC:L/Au:N/C:P/I:P/A:P)	7.5
	CVE-2014-3567	Denial of Service	(AV:N/AC:M/Au:N/C:N/I:N/A:C)	7.1
	CVE-2014-3513	Denial of Service	(AV:N/AC:M/Au:N/C:N/I:N/A:C)	7.1
	CVE-2014-3512	Denial of Service, Overflow	(AV:N/AC:L/Au:N/C:P/I:P/A:P)	7.5

Analysing all vulnerabilities, the most critical shown in the table above and other not shown, their impact is as follows:

- **Swift**. All vulnerabilities could be exploited remotely and the impact on the basic characteristics (*Confidentiality, Integrity and Availability*) is usually partial (the full system is not exposed, full information is not damaged or tampered and the system not remains unavailable for other users but with low performance).
- **Keystone.** All vulnerabilities can be exploited remotely and it is required to log into the system once. All vulnerabilities found agree to disclose only part of the system information, leave the system partially restricted (with some interoperability but low performance) and modify certain information attacking the integrity of the data but not all.
- **Python.** All vulnerabilities are remotely exploitable and most of them partially compromise system availability and confidentiality. All of them can be exploited without authentication.
- **SSH.** All vulnerabilities can be exploited without authentication against the system. All vulnerabilities somehow compromise confidentiality but CVE-2002-1715 allows for full disclosure of the system information and also completely compromises the integrity and availability.
- **Apache.** All vulnerabilities can be exploited remotely, no authentication is required and it basically can partially affect service availability (slow performance).
- **OpenSSL.** All vulnerabilities are remotely exploitable, not requires authentication and most partially compromise the confidentiality, integrity and availability of the system. There are only two vulnerabilities found that leave the system completely unavailable (CVE-2014-3567 and CVE-2014-3513).

The impact of these vulnerabilities can be avoided uploading all the components and services to the last update. Only has to be considered that Apache could have interoperability issues with Ubuntu 14.04 LTS Operating System depending on the version used.

7.2.3 Implementation Vulnerabilities

This subsection presents the vulnerabilities due to poor code writing or related to poor application design. Some tests have been performed against the actions allowed in the API to check its vulnerabilities and the results are as follows:

- **Brute force attack to CDMI authentication.** There is no limit to authenticate against CDMI. It has been proved to authenticate a lot of times with the same user and different password, and there have not appeared any error about the fact of exceeding any limit of times to authenticate.
- **Overwrite container that already exists**. If it is tried to create a container with the same name of another that already exists, in the same location, the HTTP request is accepted, but the data is not overwritten. This action not supposes a critical problem or vulnerability. It only highlights that notifications of such actions have to be incorporated in the system.
- **Overwrite file content.** If it is tried to upload a file that already exists, it is overwritten. It supposes a problem because a malicious user with guaranteed access to the system can overwrite files of another user with its own content.
- **Upload file without parameter @myobject.** @myobject is the parameter used to define the name of the file that is going to be uploaded. If a file is uploaded without @myobject parameter defined, a file without content is uploaded. This action not supposes a critical problem or vulnerability. It only highlights the need to change the API implementation to check that all parameters are defined before a file uploading.
- **Download file without the filename parameter.** If you do not put the name of the file that you want to download, the content of the file downloaded will be information about container. This information is not sensitive data at all, but delivers information about characteristics of the container.
- **Download file with a name that does not exist.** It is not allowed to download a file that does not exist. It will give you an error output.
- **Limitation in number of containers.** It is possible to edit a parameter named *max_container_per_account* to limit the maximum number of containers per user. But it seems that need a plugin to work because more containers are created than the number configured, and there were not any error. It is not a big problem in terms of space, but it can be used to do a DDoS (Distributed DoS) attack, saturating the server, because it is permitted to launch infinite requests.
- **Maximum file size.** In <swift.conf> file it can be edited the maximum size of files and it is respected in the operation of storing data.
- **Maximum container size.** There is no limitation on space per user. One user can use all the available space. It has been tried to upload files indefinitely. An error has shown when all the storage space has been filled.

The actions that suppose risks to the system have been valuated with CVSS criteria and using the NVD CVSS calculator v2. The results are shown in “Table 7”.

Table 7. Implementation vulnerabilities evaluation.

Operations	CVSS			
	Vector	*Base Score*	*Impact Subscore*	*Exploitability Subscore*
Brute force attack to CDMI authentication (tries not controlled)	(AV:N/AC:M/Au:N/C:C/I:C/A:N/CDP:ND/TD:ND/CR:H/IR:H/AR:ND)	8.8	**10**	**8.6**
Overwrite file content (actions not controlled and not notified)	(AV:N/AC:M/Au:S/C:N/I:C/A:N/CDP:ND/TD:ND/CR:ND/IR:H/AR:ND)	6.3	**10**	**6.8**
Limitation in number of containers	(AV:N/AC:M/Au:S/C:N/I:N/A:C/CDP:ND/TD:ND/CR:ND/IR:ND/AR:H)	6.3	**10**	**6.8**
Maximum container size	(AV:N/AC:M/Au:S/C:N/I:N/A:C/CDP:ND/TD:ND/CR:ND/IR:ND/AR:H)	6.3	**10**	**6.8**

Impact Subscore is adapted with environmental metrics depending on the specific impact in confidentiality, integrity, and availability requirements for the system designed.

7.2.4 Feasible Attacks

Different types of attacks have been evaluated. They could be classified in three categories:

1. **Injection attacks** [32]. The performance of **SQLi** (SQL injection), **LDAPi** (LDAP injection), **CSRF** (Cross-Site Request Forgery) and **XSS** (Cross-Site Scripting) have been analyzed and the results are as follows:
    - It could be tried to perform an SQL and LDAP injection because *Openstack* can use both systems. SQL injection is difficult to perform because there is no web based user interface or similar implemented and the knowledge required to attack the system is really high or it is required a granted access to directly see databases. LDAP is not installed in the system.
    - The probability to be infected by CSRF or XSS attacks is very low, because there is no interface which user can interact; all is used by command line, and only few commands are allowed. There is a risk because it depends of the user that manages the system to only download from legitimate links, and be careful what terminal entries use. In the case the system would be infected, the risk of data integrity and confidentiality or system availability would be higher or lower depending on the quality of the script used to infect the system.
2. **SSL related attacks**. It has been tried to capture traffic sniffing the network to see if data transfers are secure. Moreover, **SSLStrip tool** [33] has been used to capture HTTPS traffic and try to decrypt them.

- **Sniffing**. There is no encryption in the transactions done between FIDEVs in the private cloud and FIDEVs in the FIWARE Lab. If an attacker does a Man-in-the-Middle and then sniff the network, he/she can read all the traffic traveling between these two points. An HTTP Request to FIWARE Lab has been performed to verify credentials sent. Due to all traffic that travels through the network uses a non-encrypting protocol, it can be seen the token, host, email, password, etc. The best way to prevent these type of attacks, which are so easy to perform, is using an encrypted protocol like HTTPS. With all the information captured about a user by sniffing the communication, an attacker can access with guaranteed credentials. All the integrity of the information and confidentiality of the transactions are involved.
- **SSLStrip**. *Openstack* is secure against SSLStrip attacks in private clouds because of it uses ONLY HTTPS to communicate between FIDEV nodes (this is because the test is performed simulating an environment where only HTTPS in activated). Whether if it would use both protocols (HTTP and HTPPS), and a web environment, the probability to suffer this kind of attack would be much higher. For these reasons the risk is almost null in the test environment.
- **Other Attacks.** There are some attacks that can be feasible caused by the low encryption used in the system. In last years have been found some critical vulnerabilities in SSL protocol. Most of them related with SSL's first versions and weak cipher, like RC4. The execution of testssl.sh shows that these protocols and ciphers are allowed. Several of the most important attacks related with SSL protocols are: **CRIME** [34], **BREACH** [35], **POODLE** [36], **HEARTBLEED** [37]. To prevent these types of attacks is better to not use SSL v.3 or RC4 cipher, and only allow TLS 1.1 or above versions, and disable all weak cipher, because an attacker can cause a downgrade of the encryption and perform the attack. And finally to be sure that the system is not weak, it would be better to disable all 128 bits encryption mechanisms, because nowadays it can be broken so much easily. In general terms, these kind of attacks are not critical if they are performed by a beginner or with a non-powerful computer. In the other hand, if the attacker is a specialized one, or has a powerful computer, he/she can perform the attack so much easier (although it's not an easy attack), and all the integrity and confidentiality of the information are in risk.

3. **DoS Attacks.** The computer where are installed our virtual machines is not powerful. So a simple DDoS attack can be performed without many problems. First it has been tried to make a simple script that does not stop to authenticate, and after a couple of minutes, the uploading of a file has been tried. And as it was expected, the file could not be uploaded because the system had been collapsed. After that, different test against system bandwidth has been performed. DoS attacks have been proved with three different tools: **LOIC [**38**]**, **BoNeSi** [39] and **Slowloris** [40]. Just one desktop computer (CPU: i7-4720HQ, RAM: 16GB, bandwidth: 50Mbps) has been used to perform these attacks. The results are as follows:

- **LOIC**. To perform a successful attack with this tool it is needed a botnet, or something similar. That is the reason because the test failed (not availability of a botnet), and the system performance remains as usually.
- **BoNeSi**. It has been proved BoNeSi to simulate a botnet. And the results were the same as LOIC, the system responds perfectly.
- **Slowloris**. The attack was successful and the authentication could not be performed.

As it has been seen, the risk to be affected by these types of attacks depends on the dimensions of the botnet and the powerful of its machines. If there is not a system to mitigate these kind of attacks, like a firewall or a system that ban an IP if it tries to interact with the system more than the normal use in a short time, the impact could be dangerous.

"Table 8" shows the severity punctuation obtained by the weaknesses of the system and exposure to be damaged depending on the type of attack. These has been valuated with CVSS criteria and using the NVD CVSS calculator v2.

Table 8. Weaknesses related to feasible attacks.

Attack type		Risk			
		CVSS			
		Vector	*Base Score*	*Impact Subscore*	*Exploitability Subscore*
Injection attacks		(AV:N/AC:H/Au:N/C:P/I:P/A:N/CDP:ND/TD:ND/CR:M/IR:M/AR:ND)	4	**4.9**	**4.9**
SSLrelated attacks	Sniffing	(AV:N/AC:L/Au:N/C:C/I:N/A:N/CDP:ND/TD:ND/CR:H/IR:ND/AR:ND)	7.8	**10**	**10**
	SSLStrip	(AV:N/AC:L/Au:N/C:N/I:N/A:N/CDP:ND/TD:ND/CR:H/IR:ND/AR:ND)	0	**0**	**10**
	Cypher and Encryption vulnerabilities	(AV:N/AC:L/Au:N/C:C/I:P/A:N/CDP:ND/TD:ND/CR:H/IR:H/AR:ND)	8.5	**10**	**10**
DoS attacks		(AV:N/AC:L/Au:N/C:N/I:N/A:C/CDP:ND/TD:ND/CR:ND/IR:ND/AR:H)	7.8	**10**	**10**

Impact Subscore is adapted with environmental metrics depending on the specific impact in confidentiality, integrity, and availability requirements for the system designed.

7.2.5 Security Audit Results

Security issues, threats and vulnerabilities in Cloud Computing have been studied and the key to deliver secure services through the cloud resides in perfectly knowing all these problems associated and try to apply "security by design". The most important security issues related to the system developed for the Smart Energy use case are the following:

- All interactions that take place with the CDMI (FIWARE Lab) are using HTTP making possible that a malicious user could obtain user credentials and the key with which the information is encrypted.
- Files can only be interpreted if the decrypt key is available or captured because files are stored encrypted.

- A malicious user without credentials can use brute force or dictionary attacks to access the system. The *Keystone API* does not protect the continued attempts.
- *Object Storage API* has the following operational problems regarding containers that cause unavailability of the storage service:
    - A user can create an unlimited amount of containers, which could disable the system for other users.
    - The size of the containers is not limited. It is possible for a single user uploading files to hold the total storage space causing other users inability to upload files.
- *Object Storage API* does not notify when a user tries to overwrite a file. Therefore, a malicious user who has gained access to the system or even a user with guarantees but erroneously performing operations, could introduce files masquerading the old ones that were already stored. If the content of the files is not checked, changes in storage are undetectable.

Once the system is analyzed, and the vulnerabilities are known, solutions can be taken to solve security issues presented in previous sections. “Table 9” shows some solutions to each security issue discovered for *Smart Energy* use case.

Table 9. Solutions to security issues in Smart Energy use case.

Security Issue		Solution
Data Security	***Data Leakage***	Simplest scenario: An attacker has to be authenticated to steal data. Keystone controls user access. Files protected. However, if the attacker sniffs traffic sent to the public cloud and captures valid credentials, data leakage shall not be avoided. Moreover, a file transferred to the public cloud could be intercepted without need to steal credentials. By the moment there is any solution to this problem. Anyway, at least files are always encrypted from the source.
	Data Forgery	It is needed a mechanism to notify changes in data stored in the distributed system. By the moment the system has not notification tools. Moreover, if it is taken into account data sent or received from the public cloud, strong hashes used avoid tampering of data.
	Data Lost	The system is a distributed system storage that maintains multiple copies of each file using Rsync. Data lost could be restored.
Network Security	***Data Transaction***	Data exchanges between FIDEVs inside the private cloud will be encrypted by activating HTTPS and using SSL. However, data exchanged with FIWARE Lab (public cloud) through CDMI only can be transmitted with HTTP (insecure). Multiple requests have been launched to FIWARE Lab administrators to activate HTTPS. The infrastructure could be protected against DoS attacks using level 7 firewalls and IPS technology.
	Commands execution	Both in the public cloud as in the private cloud, a protection system against DoS attacks has to be implemented. It could be ensured that this premise is true in the private cloud as it depends on the organization but in the public cloud certain SLAs will be established with the CSP.
Authentication		Keystone manages the authentication process but brute force attacks are not controlled.
Authorization		Keystone manages authorization rules.
Identity Management		Keystone manages user accounts.

8. Conclusions

Cloud Computing solutions provide great flexibility for existing business given the wide range of solutions (IaaS, PaaS, SaaS) that can be implemented internally or externally in organizations. Given its cost efficiency by implementing models like SaaS, it has become a growing trend but the fact that more and more users rely on a CSP puts these suppliers in the crosshairs of malicious users. When an organization thinks of moving information or applications to the cloud, it must carefully analyze the threats and risks to which it is exposed as the business model it needs. Above all, it is very important to ensure that the security policy established in the organization extends to the cloud and perform this will require a level of service by setting a SLA between the CSP and the organization. In this way the organization, operating as a client of cloud services may have cataloged the risks to which it is exposed.

For the *Smart Energy* use case, it has been proved the need to apply to the chosen *cloud* solution various analysis and audits to detect possible threats and risks. Hence, it gives the option to take countermeasures before finally putting the system into production. In addition, the reliance on public cloud of FIWARE demonstrates the importance of knowing how are implemented cloud solutions of the CSP provider to face problems related to *Data Location, Data Segregation, Data Violation, Data Availability, Data Access between tenants, Virtualization and Web Applications Security*, which are more dependent on agreements with the CSP held by the SLAs and the implementation of physical infrastructure than the proposed system.

Next steps of this project may include the monitoring of the real production environment of the electric company to assure that security requirements defined are implemented and, the risks in the data transactions and data storage are minimized with the implementation of the solutions provided.

Acknowledgement

This work is being carried out thanks to FINESCE funding, a project developed in the Seventh Framework Program (FP7) of the European Commission under the FI.ICT-2011 call, Grant Number 604677. The authors also thank to *Agència de Gestió d'Ajuts Universitaris i de Recerca* for its support to the GRITS Research group (*Grupo de Investigación en Internet Technologies & Storage (2014-SGR-589)*) and to La Salle University – Ramon Llull.

References

[1] Selga J.M., Zaballos A., Navarro J., *"Solutions to the Computer Networking Challenges of the Distribution Smart Grid"*, IEEE Communications Letters, Vol. 17, Issue 3, pp.588-591, March 2013. http://dx.doi.org/10.1109/LCOMM.2013.020413.122896

[2] Sancho A., Navarro J., Arrieta I., Armendáriz J.E., Jiménez V., Zaballos A., Golobardes E., *"Improving Data Partition Schemes in Smart Grids Via Clustering Data Streams"*,

Expert Systems with Applications, Vol. 41, Issue 13, pp.5832-5842, October 2014

[3] Selga J.M., Corral G., Zaballos A., Martín de Pozuelo R., *"Smart Grid ICT Research Lines out of the European Project INTEGRIS"*, Network Protocols and Algorithms, Vol. 6, Issue 2, pp.93-122, June 2014. http://dx.doi.org/10.5296/npa.v6i2.5439

[4] *"FINESCE Project"*. Link: http://www.finesce.eu/ (last accessed November 12, 2015)

[5] *"Future Internet PPP European Programme"*. Link: https://www.fi-ppp.eu/ (last accessed November 12, 2015)

[6] *"INTEGRIS Project"*. Link: http://fp7integris.eu/index.php (last accessed November 01, 2015)

[7] *"FIWARE platform"*. Link: https://www.fiware.org/ (last accessed October 03, 2015)

[8] *"FIWARE catalog of Generic enablers"*. Link: http://catalogue.fiware.org/ (last accessed November 21, 2015)

[9] *"ESB (Electricity Supply Board) - Ireland's premier electricity utility"*. Link: https://www.esb.ie/ (last accessed September 26, 2015)

[10] Fernandes D.A.B., Soares L.F.B., Gomes J.V., Freire M.M., Inácio P.R.M., *"Security issues in Cloud Environment: A survey"*, International Journal of Information Security, Vol. 13, Issue 2, pp.113-170, April 2014. http://dx.doi.org/10.1007/s10207-013-0208-7

[11] Venkata S., Padmapriya S., *"A Survey on Cloud Computing Security Threats and Vulnerabilities"*, International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering, Vol. 2, Issue 1, pp.622-625, January 2014. Available at: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.428.7796

[12] FIWARE Catalogue, *"Object Storage GE"*. Link: http://catalogue.fiware.org/enablers/object-storage-ge-fiware-implementation (last accessed November 21, 2015)

[13] OpenStack, *"SWIFT"*. Link: https://wiki.openstack.org/wiki/Swift (last accessed November 21, 2015)

[14] FIWARE Catalogue, *"Identity Management - KeyRock"*. Link: http://catalogue.fiware.org/enablers/object-storage-ge-fiware-implementation (last accessed November 21, 2015)

[15] Mell P., Grance T., *"The NIST Definition of Cloud Computing"*, National Institute of Standards and Technology, Special Publication 800-145, September 2011. http://dx.doi.org/10.6028/NIST.SP.800-145

[16] Subashini S., Kavitha V., *"A survey on security issues in service delivery models of cloud computing"*, Journal of Network and Computer Applications, Vol. 34, Issue 1, pp.1-11, January 2011. http://dx.doi.org/10.1016/j.jnca.2010.07.006

[17] Soofi A.A., Khan M.I., Talib R., Sarwar U., *"Security Issues in SaaS Delivery Model of Cloud Computing"*, International Journal of Computer Science and Mobile Computing, Vol. 3, Issue 3, pp.15-21, March 2014. Available at: http://www.ijcsmc.com/docs/papers/March2014/V3I3201401.pdf

[18] Dillon T., Wu C., Chang E., *"Cloud Computing: Issues and Challenges"*, 24th IEEE International Conference on Advanced Information Networking and Applications, ISBN 978-1-4244-6695-5, pp.27-33, April 2010. http://dx.doi.org/10.1109/AINA.2010.187

[19] Cloud Security Alliance (2013) "*Notorious Nine: Cloud Computing Top Threats*". Link: https://cloudsecurityalliance.org/download/the-notorious-nine-cloud-computing-top-threats-in-2013/

[20] Cloud Security Alliance (2011) "*Security guidance for critical areas of focus in Cloud Computing V3.0*". Link: https://cloudsecurityalliance.org/guidance/csaguide.v3.0.pdf

[21] Gartner (2008) "*Assessing the Security Risks of Cloud Computing*". Link: https://www.gartner.com/doc/685308

[22] OWASP (2013) "*Top Ten Project*". Link: https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project#tab=OWASP_Top_10_for_2013 (last accessed October 26, 2015)

[23] OpenStack, "*Keystone*". Link: https://wiki.openstack.org/wiki/Keystone (last accessed November 21, 2015)

[24] Rsync. Link: https://rsync.samba.org/ (last accessed November 28, 2015)

[25] Kali by Offensive Security, "*Kali Linux VM*". Link: https://www.kali.org/ (last accessed November 28, 2015)

[26] Mitre.org, "*CVE Common Vulnerabilities and Exposures*". Link: https://cve.mitre.org/ (last accessed December 01, 2015)

[27] "*National Vulnerability Database*". Link: https://nvd.nist.gov/home.cfm (last accessed December 01, 2015)

[28] "*CVE details*". Link: http://www.cvedetails.com/ (last accessed December 01, 2015)

[29] NVD, "*CVSS calculator v2*". Link: https://nvd.nist.gov/CVSS-v2-Calculator (last accessed December 01, 2015)

[30] FIRST.org, "*Common Vulnerability Scoring System*". Link: https://www.first.org/cvss (last accessed December 01, 2015)

[31] Nmap, "*Free Security Scanner For Network Exploration*". Link: https://nmap.org/ (last accessed November 18, 2015)

[32] Issac B., Israr N., "*Case Studies in Secure Computing: Achievements and Trends*", CRC Press, August 2014, ISBN 978-1-4822-0706-4, Chapter 17, pp. 343-372. http://dx.doi.org/10.1201/b17352

[33] "*SSLStrip tool*", Link: https://github.com/moxie0/sslstrip (last accessed November 30, 2015)

[34] "*CRIME Attack*". Link: http://www.pcworld.com/article/262307/crime_attack_abuses_ssltls_data_compression_feature_to_hijack_https_sessions.html (last accessed November 30, 2015)

[35] "*BREACH Attack*". Link: http://breachattack.com/ (last accessed November 30, 2015)

[36] "*POODLE Attack*". Link: https://poodle.io/ (last accessed November 30, 2015)

[37] "*HEARTBLEED Attack*". Link: https://www.owasp.org/index.php/Heartbleed_Bug (last accessed November 30, 2015)

[38] "*LOIC tool*". Link: http://sourceforge.net/projects/loic/ (last accessed November 30, 2015)

[39] "*BoNeSi tool*". Link: https://github.com/markus-go/bonesi (last accessed November 30, 2015)

[40] "*Slowloris tool*". Link: https://github.com/llaera/slowloris.pl (last accessed November 30, 2015)