D3.3.4

Final Report on Evaluation Activities

Project number:	257243
Project acronym:	TClouds
Project title:	Trustworthy Clouds - Privacy and Resilience for Internet-scale Critical Infrastructure
Start date of the project:	1st October, 2010
Duration:	36 months
Programme:	FP7 IP

Deliverable type:	Report
Deliverable reference number:	ICT-257243 / D3.3.4 / 1.0
Activity and Work package contributing to the deliverable:	Activity 3 / WP 3.3
Due date:	30th September 2013 – M36
Actual submission date:	4th October 2013

Responsible organisation:	FCSR
Editor:	Marco Abitabile
Dissemination level:	Public
Revision:	1.0

Abstract:	D3.3.4 describes the execution of the validation activities as described in D3.3.3. Moreover it gives an overview of the other sub-components and their validation.
Keywords:	Validation execution, subcomponents, healthcare, Smart Light System, A2 infrastructure

Editor

Marco Abitabile (FCSR)

Paulo Jorge Santos (EFA)

Contributors

Martin Deutschmann, Sebastian Ressi (TEC)

Sören Bleikertz (IBM)

Norbert Schirmer (SRX)

Mihai Bucicoiu (TUDA)

Alysson Bessani, Marcel Santos (FFCUL)

Paolo Smiraglia, Roberto Sassu (POLITO)

Johannes Behl, Klaus Stengel (TUBS)

Nuno Emanuel Pereira, Miguel Areias (EDP)

Disclaimer

This work was partially supported by the European Commission through the FP7-ICT program under project TClouds, number 257243.

Executive Summary

The following document describes all the work performed following the D3.3.3 strategy.

The aim of Task 3.3.3 is to validate and evaluate the TClouds platform. As part of a proper research and development cycle as well as quality control, an evaluation and validation component is necessary. It ensures that the requirements specified are met and that problems, defects and malfunctions are prevented. Although much of this should already occur in an iterative fashion throughout implementation of the requirements, a more formal point in time allows for careful planning and targeted efforts.

A3 within TClouds has as its focus the evaluation of the TClouds platform, as well as the development of applications to run on this platform. Two scenarios have been selected for this purpose. The former is a Home Healthcare case while the latter is a Smart Lighting System use case. A3 links with A1 as it uses the requirements generated there as general guidelines to adhere to. It further links to A2 as it ensures alignment of the applications to the general objectives of TClouds.

This document reports on the execution of the validation activities as defined in D3.3.4 and describes the validation activity from the point of view of the A3 scenario. The validation includes also those components that have not been directly used by the Healthcare and Smart Light System scenario. The overall idea behind this is that TClouds infrastructure resulted in a comprehensive tool able to host different customer needs. The Healthcare and Smart Light System scenario represent two particular realities that need specific cloud features. Nonetheless, TClouds encompass other subcomponents that might be useful for other needs. Most of these components are high level components that take advantage from the SaaS paradigm and can be useful to all those companies that do not have to setup complex platform or systems, but just need cloud features for internal activities as well as to externalize the IT infrastructures.

This document contains an extensive description of all validation processes performed for each component of TClouds Infrastructure.

We would suggest that this document be read not in a traditional sequence, but starting from the end: the conclusions of each chapters (surveys in Chapter 2 and validation activities in Chapter 3) and the final thoughts in Chapter 4 provide the overall view of the work that has been done.

We would then suggest that the bulk of Chapter 3 be read to have a deeper understanding of how the validation activities have been performed and their outcomes.

Chapter 3 shows the validation activity execution of the Healthcare and Smart Light System scenario plus the validation activities of the components not directly used by the two use cases.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Chapter Authors:
Marco Abitabile (FCSR)

1.1 Work Package 3.3 – Validation and Evaluation of the TClouds platform

WP3.3 aims at defining the validation and evaluation of the TCLOUDS Platform. It makes use of the results produced by WP3.1 and WP3.2 to benchmark and quantify the innovations provided by the technical work-packages of A2. To evaluate the project results, WP3.3 will first of all define the main project dimensions that need to be evaluated and the specific strategies and activities for the validation of these dimensions. After this phase it will define qualitative and, when possible, quantitative metrics and indicators, organizing the activities needed to compute these metrics. Finally it will implement the validation activities and draw conclusions on the TCLOUDS results.

1.2 Deliverable 3.3.4 – Final report on Evaluation Activities

1.2.1 Overview

The aim of Task 3.3.3 is to validate and evaluate the TClouds platform. As part of a proper research and development cycle as well as quality control, an evaluation and validation component is necessary. It ensures that the requirements specified are met and that problems, defects and malfunctions are prevented. Although much of this should already occur in an iterative fashion throughout implementation of the requirements, a more formal point in time allows for careful planning and targeted efforts.

A3 within TClouds has as its focus the evaluation of the TClouds platform, as well as the development of applications to run on this platform. Two scenarios have been selectedfor this purpose. The former is a Home Healthcare case while the latter is a Smart Lighting System use case. A3 links with A1 as it uses the requirements generated there as general guidelines to adhere to. It further links to A2 as it ensures alignment of the applications to the general objectives of TClouds.

This document reports on the execution of the validation activities as defined in D3.3.4 and describes the validation activity from the point of view of the A3 scenario. The validation includes also those components that have not been directly used by the Healthcare and Smart Light System scenario. The overall idea behind this is that TClouds infrastructure resulted in a comprehensive tool able to host different customer needs. The Healthcare and Smart Light System scenario represent two particular realities that need specific cloud features. Nonetheless TClouds encompass other subcomponents that might be useful for other needs. Most of these components are high level components that take advantage from the SaaS paradigm and can be useful to all those companies that do not have to setup complex platform or systems, but just need cloud features for internal activities as well as to externalize the IT infrastructures.

1.2.2 Structure

This document contains an extensive description of all the validation processes performed for each component of TClouds Infrastructure.

We would suggest that this document be read not in a traditional sequence, but starting from the end: the conclusions of each chapters (surveys in Chapter 2 and validation activities in Chapter 3) and the final thoughts in Chapter 4 provide the overall view of the work that has been done.

We would then suggest that the bulk of Chapter 3 be read for a deeper understanding of how the validation activities have been performed and their outcomes.

Chapter 3 shows the validation activity execution of the Healthcare and Smart Light System scenario plus the validation activities of the components not directly used by the two use cases.

1.2.3 Target Audience

The target audience of this deliverable includes all TClouds partners, especially partners from A2, who wish to properly evaluate and consider the validation activities outcome, in order to improve TClouds security solutions.

1.2.4 Relation to Other Deliverables

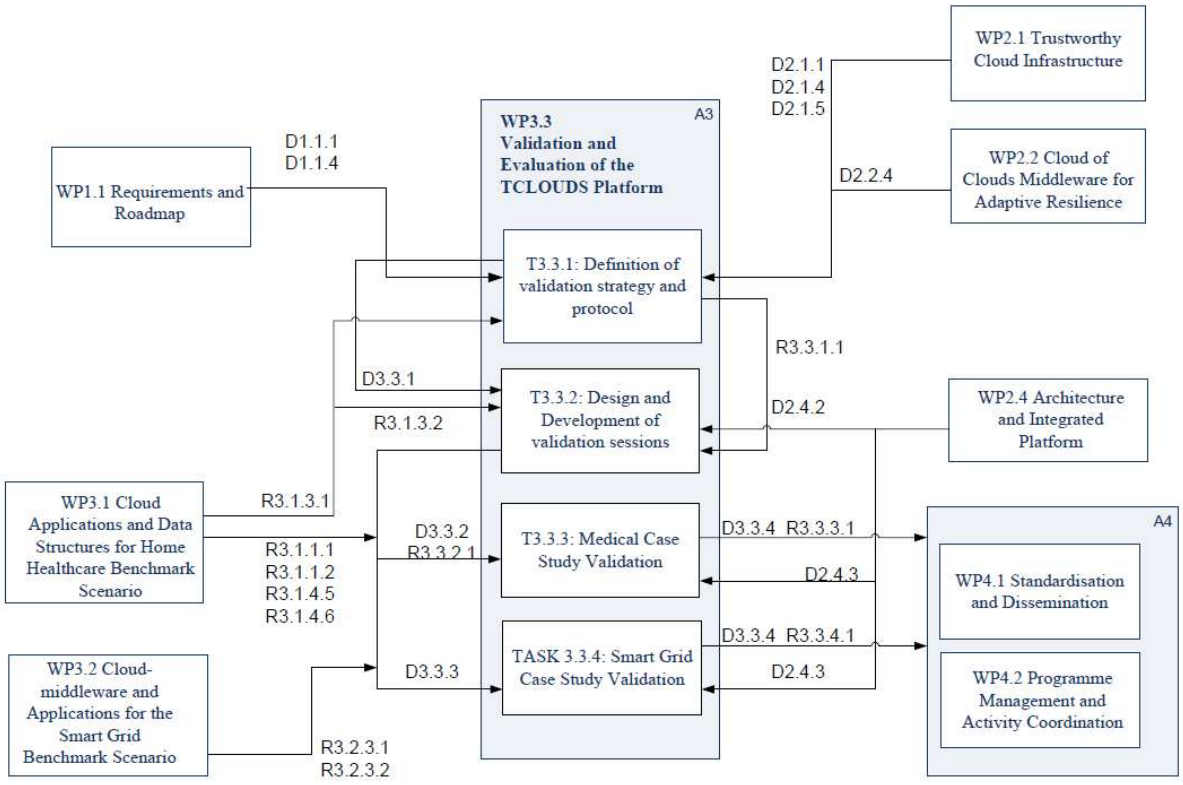


Figure 1 - Interdependency chart for WP3.3

1.3 Requirements

For the sake of simplicity this chapter lists the main A3 high-level requirements defined in D2.4.2

1.3.1 Legal requirements

LREQ1 - Confidentiality of personal data: The Cloud Provider must prevent the breach of users' personal data by securing the infrastructure (including the internal network) and ensuring the isolation among different tenants. Further, he must avoid accesses on data by unauthorized entities through accesses management or, at least, must record relevant events through an auditable logging mechanism (that also logs actions performed by Cloud provider's employees). Confidentiality can be achieved also by encrypting data in a way that decryption would be possible only for customers.

LREQ2 - Availability and Integrity of personal data: The Cloud Provider must prevent the loss or manipulation of users' personal data through Duplication and Distribution (this poses some new risks, please refer to D1.2.3).

LREQ3 - Control of location (country wise) and responsible provider (cloud subcontractor): The Cloud Provider must guarantee the applicability of law for processing personal data through location audit trails for the customer and safeguards that prevent data transfer to Cloud premises in other locations than those explicitly agreed with the customer.

LREQ4 - Unlinkability and Intervenability: The Cloud Provider must prevent unauthorized pooling, combining and merging of data through anonymization, pseudonymisation and splitting of data, through encryption of personal data (decryption only by customer) or isolation of tenants. The Cloud Provider must prevent the loss of control of data due to unauthorized copies through the encryption of data (with decryption by customers) or the effective and complete deletion. He must also provide customers with extensive control functions to avoid the risk of hindrance of the data subject's rights of access, rectification, erasure or blocking of data.

LREQ5 - Transparency for the customer: The Cloud Provider must inform his customers about the security measures adopted to protect their personal data against loss of control due to unauthorized copies, manipulation, unauthorized pooling, combining and merging. The Cloud Provider must also prove that he did not circumvent the security measures chosen by providing customers with an auditable logging of accesses made by himself and his employees.

1.3.2 Healthcare requirements:

AHSECREQ1 - Confidentiality of stored and transmitted data:

Prevents an attacker from retrieving and disclosing data from the patient data repository or information transmitted through the communication channel between the personal front end and the management application.

AHSECREQ2 - Integrity of stored and transmitted data:

Detects corruption done by an attacker of data stored in the patient data repository or exchanged through the communication channel between the personal front end and the management application.

AHSECREQ3 - Integrity of the application:

Detects corruption of the management application done by an attacker to modify its functionality.

AHSECREQ4 - Availability of stored and transmitted data:

Prevents Denial-of-Service attacks to the patient data repository or to the communication channel between the personal front end and the management application.

AHSECREQ5 - Availability of the application:

Prevents Denial-of-Service attacks to the management application.

AHSECREQ6 - Non repudiation:

Prevents an attacker from denying the fact that he/she has ever performed a specific action (e.g. he/she made the data available to unauthorized parties).

AHSECREQ7 - Accountability:

Detects actions done by an attacker to provide him/her with privileges for the patient that should not be assigned to him/her.

AHSECREQ8 - Data source authentication:

The attacker must not be able to run a process that appears as the legitimate management application.

AHPRIVREQ1 – Un-linkability and Anonymization of data flow:

Uses data anonymization/pseudonymization techniques to anonymize/pseudonymize the documents stored in the data store and enforces process confidentiality (e.g. the state, the memory and administrative interfaces of the process) by means of strong/secure access control.

1.3.3 Smart Lighting System Requirement

ASSECREQ1 – Trustworthy Audit: Smart Lighting actions (application access, create, update, and delete data) must be fully audited, and accessible only to privileged users

ASSECREQ2 - **Trustworthy Infrastructure:** The hosting infrastructure must prevent intrusions.

ASSECREQ3 - **Trustworthy Persistence Engine:** The persistence engine must prevent intrusions and ensure confidentiality, integrity and availability.

ASSECREQ4 - Resilient: The Smart Lighting System must be fault-tolerant at infrastructure and at persistence level.

ASSECREQ5 - **Trustworthy communications:** Communications between a client and the Smart Lighting System must prevent data from being altered by using adequate security mechanisms.

ASSECREQ6 - **High performance & Scalable:** The Smart Lighting System must have near real-time performance, and be able to scale on increased load.

Chapter 2

Surveys to stakeholders and results

Chapter Authors:
Marco Abitabile (FCSR), Norbert Schirmer (SRX), Ninja Marnau (ULD), Nuno Emanuel Pereira and Miguel Areias (EDP)

2.1 Introduction

This chapter describes the results of the surveys conducted during Y2 and Y3 aiming at understanding how stakeholders comprehend basic principles of the TClouds Infrastructure.

Section 2.2 shows a field study performed by A2 partners in order to judge the easiness of use of TClouds Trusted Infrastructure, while Sections 2.3 and 2.4 describes A3 survey conducted to the respective stakeholders of Healthcare and Smart Light System scenario.

2.2 TClouds Infrastructure End User Field Study

For the field study we want to evaluate the concept of “Trusted Virtual Domains” (TVD) as we employ it in the Trusted Infrastructures for Cloud Computing that we develop within the TClouds project. A TVD is a virtual infrastructure (computing, networking and storage) with trust, security and isolation guarantees, and it is put on top of the shared physical resources, in this case the TClouds infrastructure. Different TVDs are isolated by definition, by means of virtualisation, storage and network encryption (i.e. VPN technology). These mechanisms are built on into a “secure kernel” enforcing the isolation and the security policies. Such kernel is in execution on each computing node which is defined TrustedServer. Each node is managed by a central management component defined TrustedObjects Manager (TOM), which communicates via a TrustedChannel with the TrustedServers. The physical resources, as well as the security policies and the TVDs are internally managed by TOM. The TrustedChannel provides encryption, mutual authentication and integrity checks employing Trusted Computing Technology (such as the Trusted Platform Module) on both the TrustedServer and the TOM.

The target of evaluation in the field study is the Trusted Infrastructure consisting of the three components: TOM, TrustedServer and the TrustedChannel (behind the scenes). There are two groups of users that interact with the system. The former includes the administrators, setting up the servers, security policies and starting services. These interact mainly with the management component TOM. While the latter includes the end-users who using a service. To have end-to-end security the end-users that use a service within a TVD also have to use a trusted device like a TrustedDesktop, which is part of the Trusted Infrastructure. Hence we extend the components to evaluate with the TrustedDesktop as an example for a trusted endpoint to access the cloud services. On a TrustedDesktop the concept of TVD is exposed to the user. On a TrustedDesktop a user can simultaneously work with multiple compartments (each implemented as a virtual machine), where each compartment belongs to a distinct TVD (configured via the TOM). The TVD is graphically illustrated by a distinct colour associated with the TVD. The user can only access a service within a compartment that belongs to the same TVD as the service.

On this background the field study consists of two parts focusing on different aspects and user groups:

- End-users interacting with the Trusted Infrastructure via a TrustedDesktop. Here the focus of the field study is to evaluate how the concept of TVDs is accepted and understood by the end-users. Here we plan to employ students of the Ruhr-University Bochum (RUB), with which SRX collaborates on the topic of TrustedDesktop.
- Administrators which use the TOM and the TrustedServer. As these components are currently under development within the TClouds project and have not yet reached the same level of maturity as the TrustedDesktop, we don't plan to evaluate these components on a broad user basis. We plan to select early adopters from TClouds project partners, in particular Technikon.

This report elaborates on the end-user field study, performed with a group of students, from our on-going collaboration with the university. These students were equipped with a TrustedDesktop and they have been briefly informed about the concept of TVDs at the beginning of the study.

2.2.1 Questionnaire details

The main goal of the questionnaire is to find out if the participants:

- have understood the concept of TVDs and information flow control:
- have understood that data is encrypted when leaving a TVD and can only be decrypted in the very same TVD:
- have understood the concept of the Trustbar and the information displayed their (about compartments and TVDs).

The survey form is comprised of two essential parts: demographic data and questions regarding the project.

The first part contains questions about the participant, i.e. gender, subject of study and usage of the TrustedDesktop. The second part is comprised of questions regarding the project. Such section contains questions on the goals and concept of the project, its security architecture as well as exchange and encryption of data.

2.2.2 Survey results

On December 5th 2011 the participants were given an introductionary lesson on the TrustedDesktop system and the scenario of the TrustedInfrastructure. At the end of the session, 120 out of 130 participants received their laptop devices preinstalled with an installation of the TrustedDesktop system.

This survey's goal is to measure the participant's knowledge regarding the TrustedDesktop system, the TrustedInfrastructure and the TVDs and to find out which information was still present among participants. In addition, we wanted to know the participant's experience with the system.

104 members of the field study had participated in this survey. Our results show that 59 (56.73%) participants were able to remember the goals of the study and 35 members of said group did understand the goals of the TrustedInfrastructure, which is around a third of the participants of the survey. This order of magnitude is also confirmed by the detailed technical questions of the survey regarding the security benefits of the TrustedInfrastructure. We are glad about this ratio as one has to consider that the participants were confronted with novel and innovative security concepts which have influence on their everyday interaction and habits with their laptop. The TrustedInfrastructure is non-less but a paradigm shift in security, away from discretionary access control to pervasive information flow control. For example, users are unfamiliar in working simultaneously with various compartments within different security domains. Some of the participants of the field study will voluntarily continue using

the TrustedDesktop. Hence we judge the overall results of the survey as positive, but derive the need to further improve and polish the user experience and to provide more education on security concepts like TVDs.

In the following will be described questions and responses regarding the demographic part of the survey. Questions and responses to questions on the introduction session are described and discussed in the second section of this chapter.

2.2.2.1 Demographic Survey

104 out of 125 study members have participated in this survey, of whom 49 (47.12%) are female and 55 (52.88%) are male.

Subject of study: The majority of participants (51 out of 104) study a subject in the area of humanities or arts. 22 participants are students in the field of science, 19 in the area of engineering and 12 in the field of medicine.

Usage habits: Participants were asked how often they use their TrustedDesktop laptops. Figure 2 illustrates the frequency of usage among the participants.

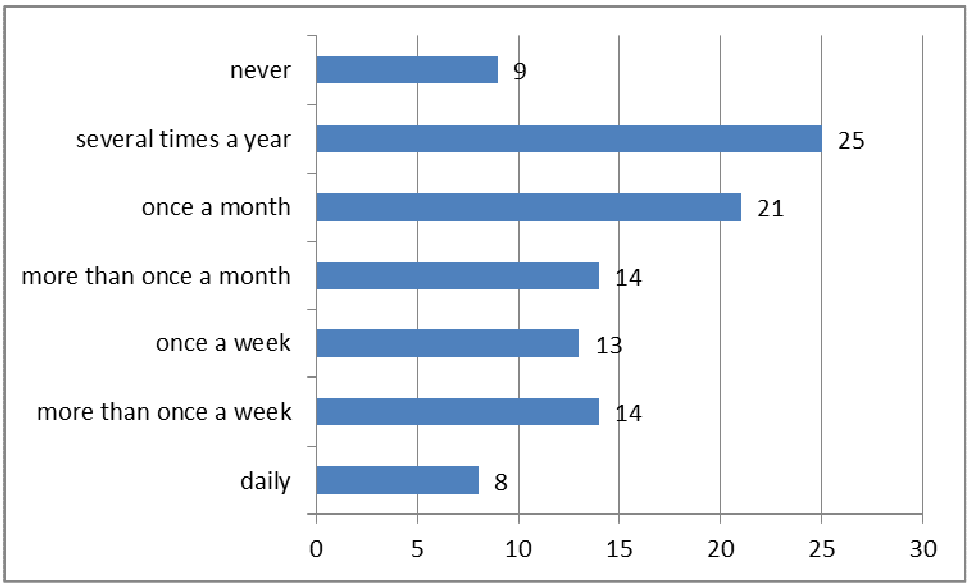


Figure 2 - Usage of the TrustedDesktop Laptop

The results indicate that 25 participants rarely and 9 never use their TrustedDesktop systems. A median usage of “once per month” shows that the majority of participants do not use their TrustedDesktop system on a daily basis.

Questioned about the reason for rare or sporadic usage of their TrustedDesktop systems, participants responded as follows:

- missing functionality, i.e. support for USB devices, non-working wireless LAN, lack of system performance and reduced battery runtime (39 participants);
- possession of another computer with full and working support for all devices (7 participants);
- No immediate necessity for the use of the TrustedDesktop system (7 participants).
- Unable to login (4 participants).
- The system is overwhelmingly complex (3 participants).

Experience with operating systems: the participants were questioned which operating system they normally prefer when not using the TrustedDesktop system. 83 participants use Windows, 6 use Linux, 9 use MacOS and 3 participants use both, Windows and Linux as their preferred operating system.

Email reading: the participants were asked which compartment they use to read their Emails. The responses to this question are depicted in Figure 3.

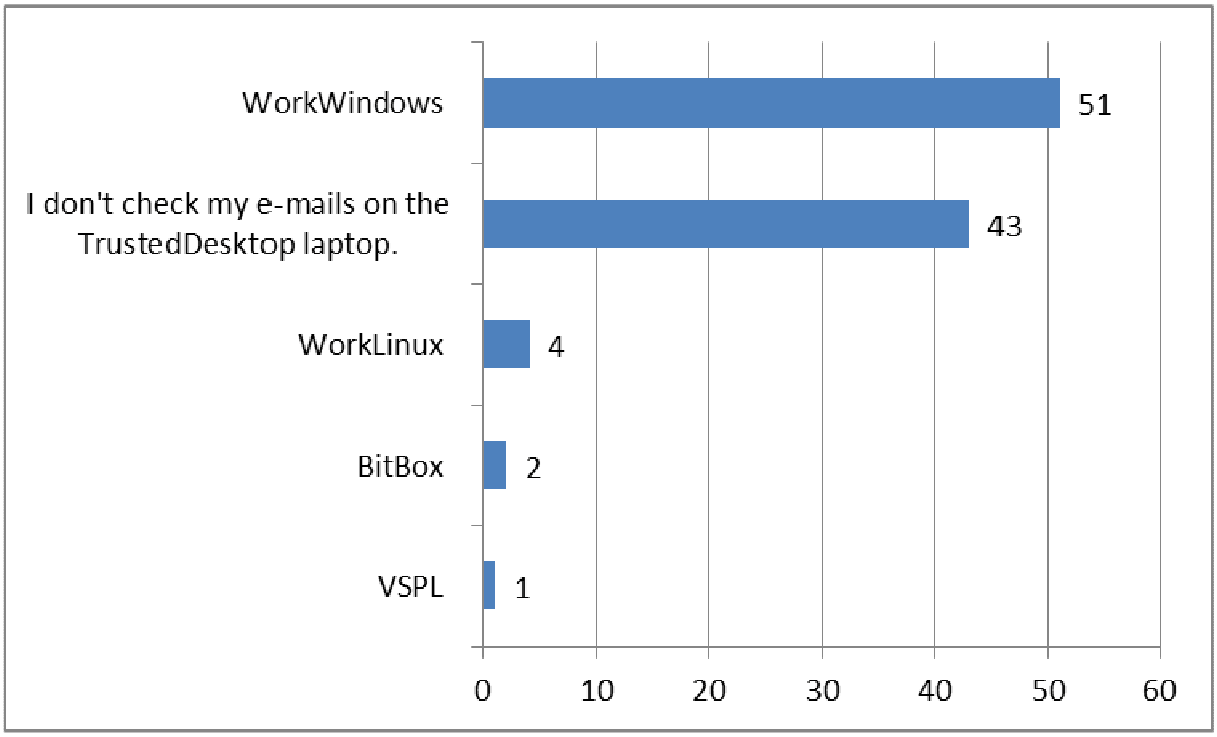


Figure 3 - Email reading

One participant had responded to read her/his Emails from within the VSPL compartment, which is, due to system policy, only feasible on servers belonging to the university network.

Positioning of the taskbar: The participants were asked where they prefer to place their taskbar when working with or without the TrustedDesktop system. Figure 4 depicts the distribution of responses to the question and indicates that the vast majority of participants prefers to place their taskbar on the bottom border of the screen.

2.2.2.2 Questions regarding the introductory course

2.2.2.2.1 Goals of the project

In the first question within the second section of the survey, participants were asked whether they could remember the goals of the project. If that was the case, they were asked to cite the project goals which they had memorized.

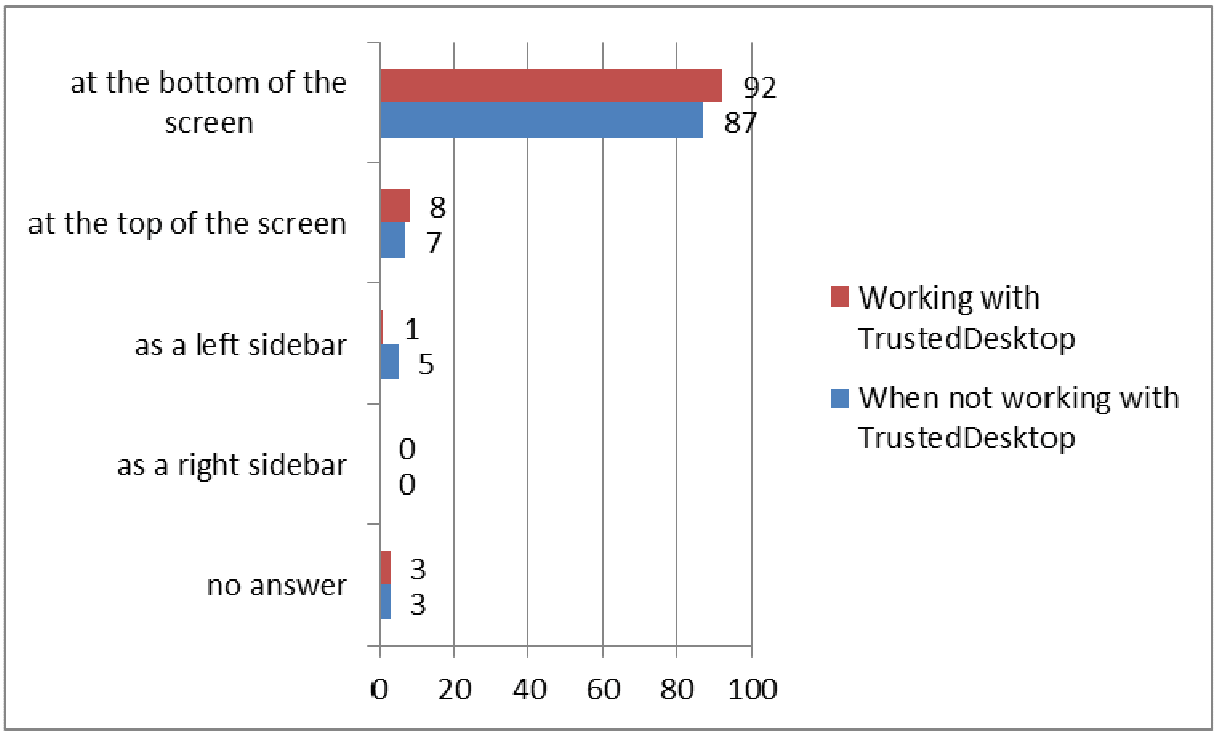


Figure 4 - Preferred position of the taskbar on the screen.

42 out of 104 participants have responded that they couldn't remember the project's goals and an additional 3 study members have not answered this question at all. The remaining 59 participants re-stated the project goals as follows:

- To test and improve the usability of the TrustedInfrastructure system (23 participants)
- Development of a secure operating system. Security is achieved through the isolation of compartments, thus creating a secure environment for sensitive data (secure usage of the VSPL client software) (21 participants)
- Protection of sensitive data against third parties/attackers, especially for the use of the VSPL client software (9 participants).
- To develop a secure operating system that is to be used on all computers on the campus and to present a safe environment for the use of the VSPL software. (5 participants).
- Secured access and storage of sensitive data with the help of isolated compartments. The compartments reside in a constrained environment, are unable to exchange data between one another and have only limited access to resources such as the internet connection, etc. (1 participant).

The responses to this question lead to the assumption that 23 participants had confused the goals of the project with the aims of the Field study. 35 participants responded that the project goals are the protection of personal and sensitive data.

One participant had responded that it is impossible to exchange data between compartments. This statement is only true for the BitBox compartment. Data exchange between compartments belonging to same TVDs is possible, while data exchange in between compartments of different TVDs is possible with certain restrictions applied.

To sum up, one can conclude that 59 (56.73%) participants were able to remember the project goals. Furthermore, it can be asserted that 35 out of this group (33.65% of all participants) have comprehended the project's goals.

2.2.2.2.2 Concept of the project

Questions #9, #13, #16, #20, #23 were created to find out whether participants were aware that the VSPL compartment's access was solely granted to servers in the university network within the same TVD. This restriction in the system was created to ensure that only trustworthy clients are able to access the VSPL services.

The following list summarizes the list of questions posed along with the answers given by the participants:

- **Usage of websites from within the VSPL compartment (Question #9)**: Participants were asked to choose one out of 3 options: (a) The VSPL compartment does not permit access to websites outside the scope of the university network, (b) the VSPL compartment grants access to any website and (c) I don't know.

  53% of the participants had ticked the correct answer (a) (see Figure 5). If we consider our frequent and infrequent groups, the results show that 32,7% of the infrequent and 45,7% of the frequent participants have no idea. That shows that there is no correlation between the "infrequent users" and having „no idea" about the question (see Figure 6). If we exclude the infrequent participants, Figure 7 shows the results of the participants who provided an answer. Here the correct answer clearly dominates.

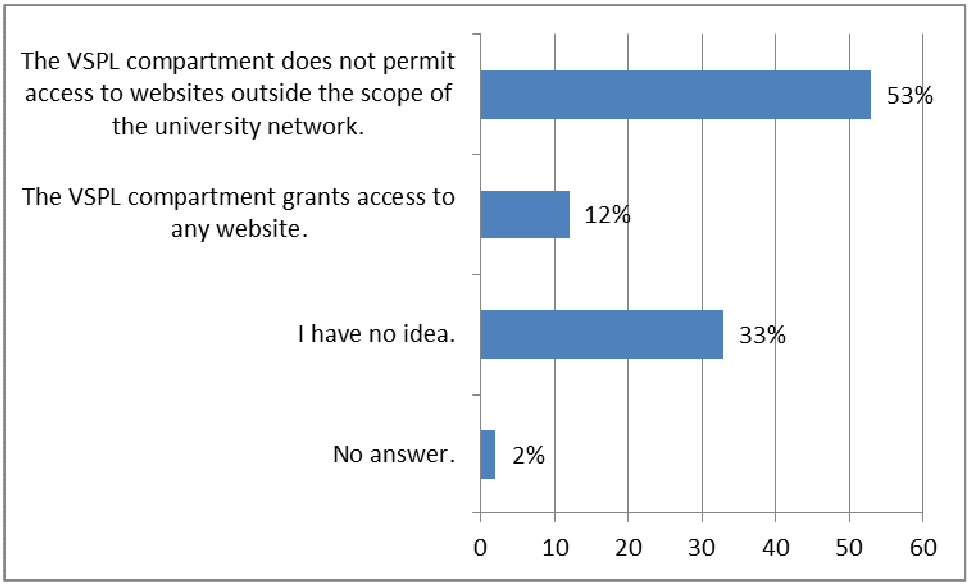


Figure 5 - Question 9: Which of the following statements is correct?

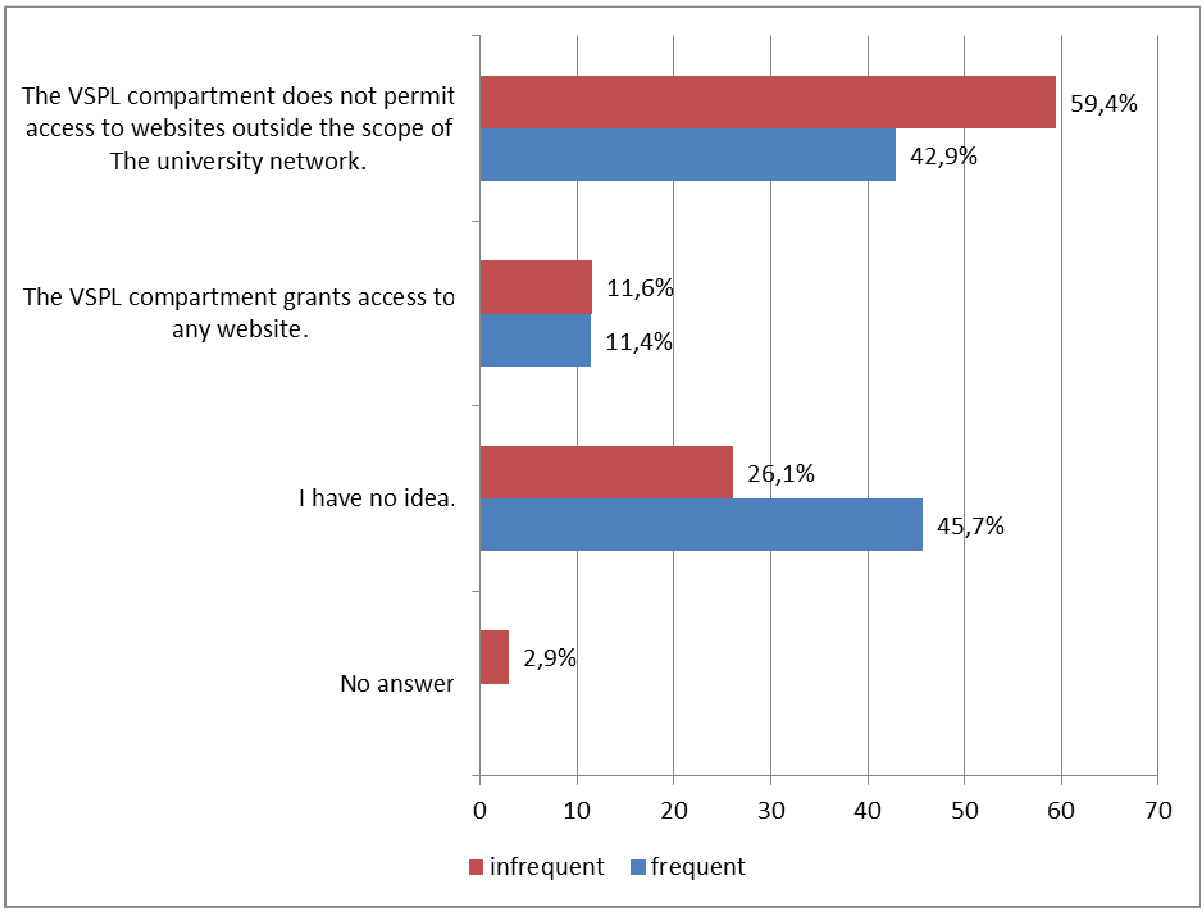


Figure 6 - Which of the following statements is correct (frequent vs. infrequent users)

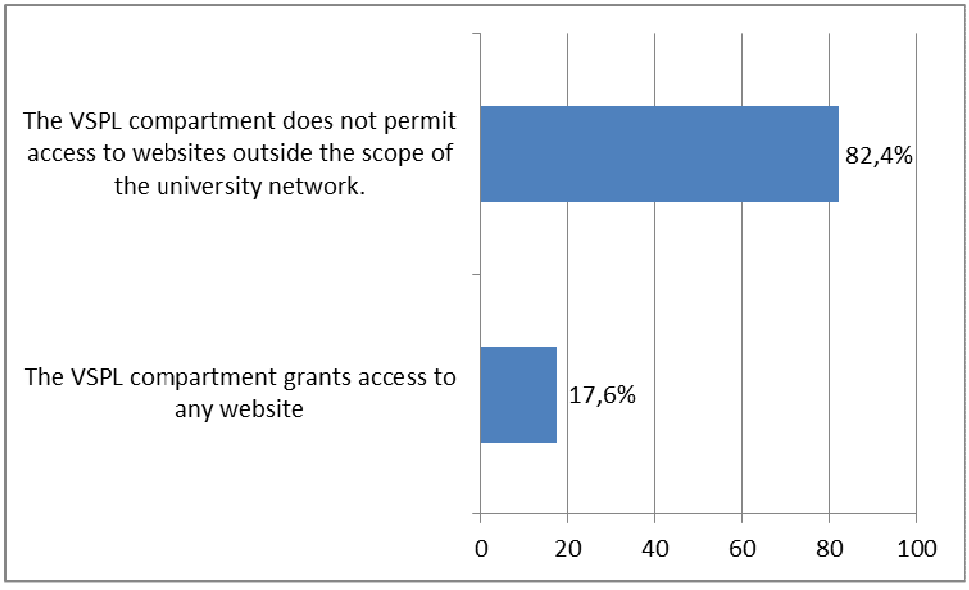


Figure 7 - Which of the following statements is correct (excluding clueless users)

- ***Usage of Email services from within the VSPL compartment (Question #13)***: Participants were questioned whether it is possible to receive or send emails from or to non-university accounts (i.e. john.doe@gmail.com).

30% of all participants provided the correct answer, namely that it is not possible (see Figure 8). 4% of all participants provided a false answer to this question and had responded that it is possible to receive or send emails from/to non-university accounts, of which one had commented his/her answer to being speculative and another participant reasoned his reply to this question with the fact that it is possible to redirect emails (which is true in this case). The remaining 62% participants had replied that they had “no idea”.

Again, if we consider our two groups, the results show that 62,3% of the infrequent and 60% of the frequent participants have no idea (see Figure 9). If we exclude these people from the overall group, we can see that the great majority of the people who provided an answer have stated the correct one (see Figure 10)

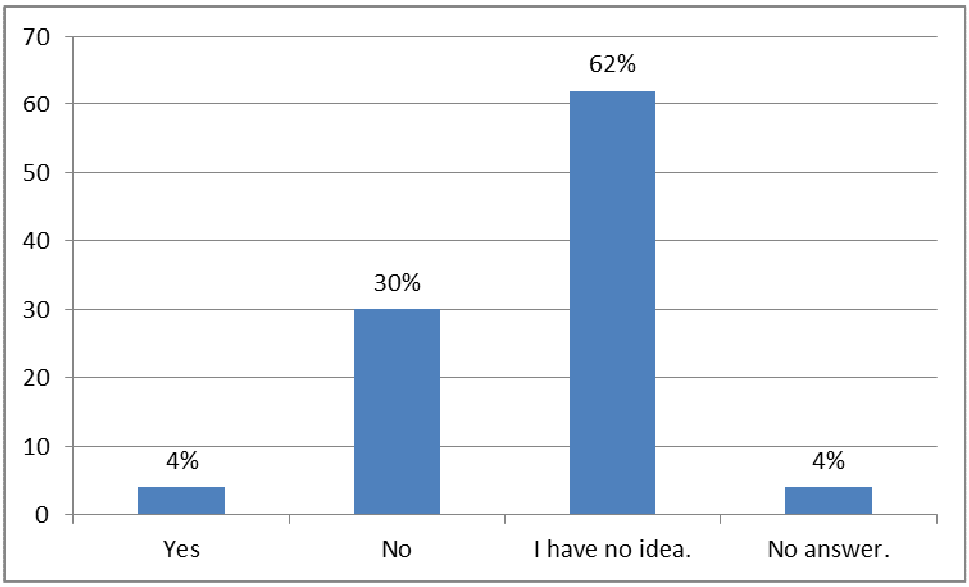


Figure 8 - Question 13: Is it possible to send or receive emails from or to non-university accounts from within the VSPL compartment?

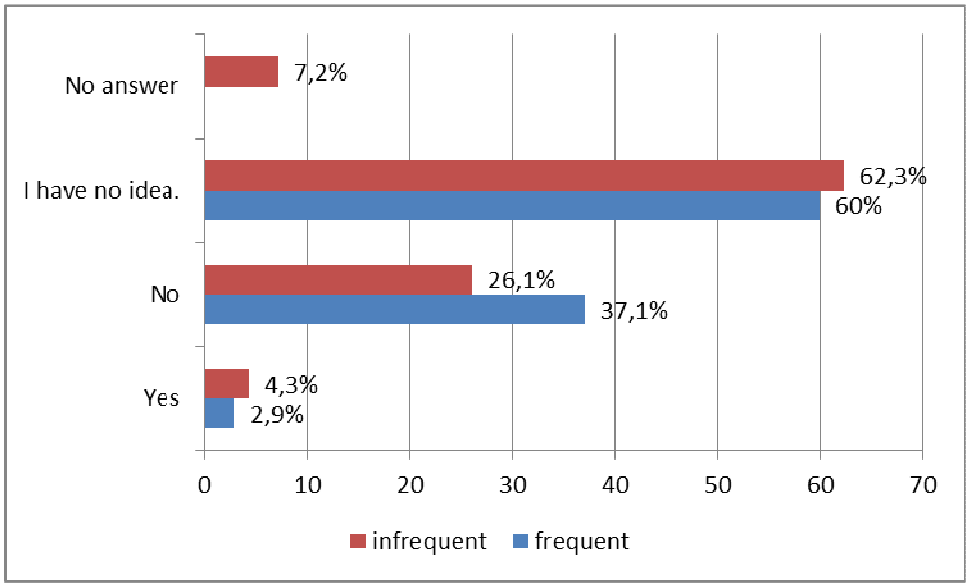


Figure 9 - Question 13: Is it possible to send or receive emails from or to non-university accounts from within the VSPL compartment?

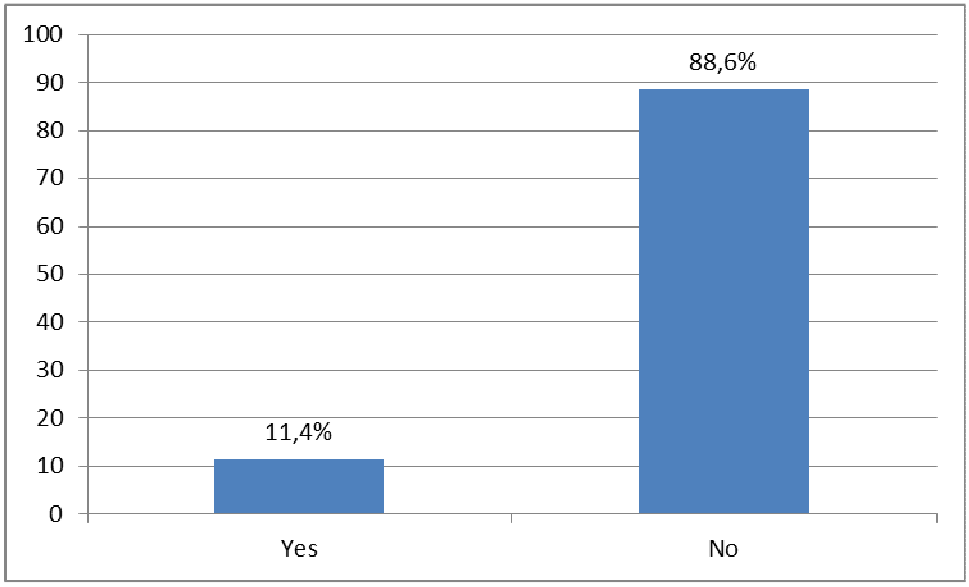


Figure 10 - Question 13: Is it possible to send or receive emails from or to non-university accounts from within the VSPL compartment?

- ***Internet attacks targeting the VSPL compartment (Question #16)****:* In question #16 participants were asked whether the VSPL compartment is protected against attacks from the Internet. 10 participants had provided the answer "*no*", of which 5 have reasoned their answer with "*as long as an internet connection persists, the compartment is not protected*". The other 5 members of this group have reasoned that "*there is no 100% guarantee against errors, i.e. implementation errors*". 44% of all participants have answered in a correct manner and stated that the VSPL compartment is secured against attacks from the internet (see Figure 11).

  In this question, again, a lot of the participants have stated that they have no idea about the answer. The results show that 48,6% of the frequent and 37,7% of the infrequent don't have an idea. Figure 12 shows the comparison. The results show entirely different statistics if we exclude the participants who answered with "I have no idea". Figure 13 shows that 82,1% of the participants who answered the question have checked the correct answer.

- ***Phishing attacks (Question #20)****:* Question #20 asked which compartment is protected against phishing attacks. All compartments were presented as valid options for this (multiple-choice) question including the options "*none of the compartments*" as well as "*I don't know*".

  The correct answer ("*only the VSPL compartment*") was given by 30 participants (29%). 18 participants had responded with "*I don't know*", 11 participants (11%) responded with "*none*", while all other participants provided false answers by selecting multiple compartments in miscellaneous combinations (see Figure 14).

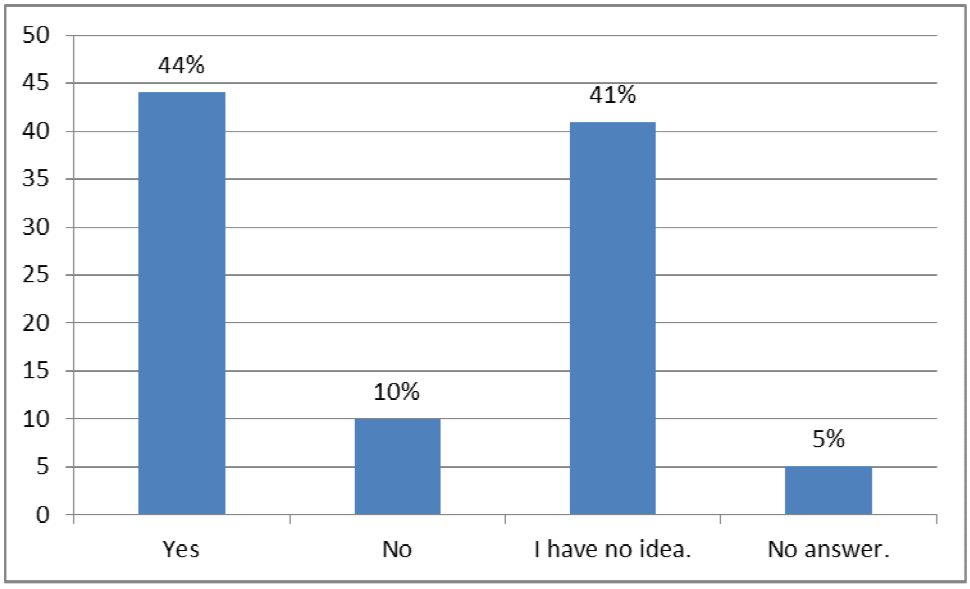


Figure 11 - Is the VSPL compartment protected against attacks from the Internet?

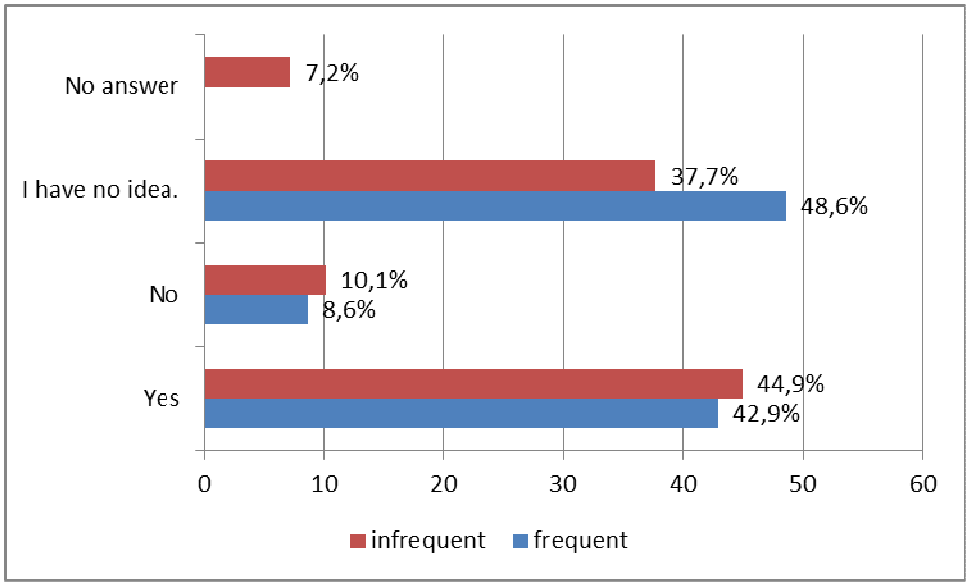


Figure 12 - Is the VSPL compartment protected against attacks from the Internet (frequent vs. infrequent users)?

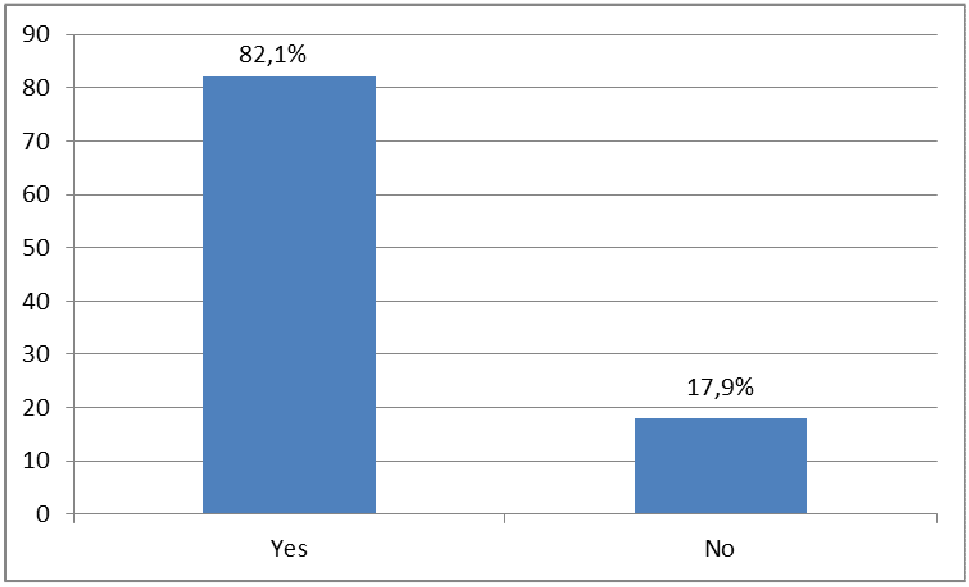


Figure 13 - Is the VSPL compartment protected against attacks from the Internet? (excluding clueless users)

- ***Using the Email account provided by the university from within the VSPL compartment (Question #23)****:* In this question, the participants were asked to choose in which compartments it is possible to access university-provided email account to send or receive emails. The participants were presented options to select one or more of all possible compartments as well as the option "*none*".

  25 (24%) of all participants have provided a correct answer to this question, namely *all compartments*. Two participants have selected the option *none*, while all remaining participants provided different combinations of (false) answers.

The answers presented above lead to the conclusion that good portion of the participants had problems remembering the concepts of the project. The percentage of correctly answered questions varies from 24% to 53% per question. Averaging all percentage values of correct answers leads to the assertion that an approximate one third (36%) of all participants have comprehended the concepts applied in the project. A plausible explanation for such high variance across different questions can be explained by the different nature of the questions posed as well as the variance in difficulty among questions. Another third of all participants (38.25%) have responded that they are not familiar with the concepts or "don't know" the correct answers to the questions presented in the survey. Excluding those "clueless answers" gives a positive result, as then the vast majority of the answers are the correct ones (see Table 1).

Question	Correct answers (percent)	Correct answers excluding „clueless"
Usage of websites from within the VSPL compartment	53%	82,4%
Usage of Email services from within the VSPL compartment	30%	88,6%
Attacks from the internet targeted at the VSPL compartment	44%	82,1%
Phishing attacks	29%	36,6%

Question	Correct answers (percent)	Correct answers excluding „clueless“
Using the Email account provided by the university from within the VSPL compartment	24%	24%
Average	36%	62,74%

Table 1 - Summary of correctly answered questions

So we learn two things from these results.

1. There is more education needed to teach the novel security concepts.
2. Once the concepts are understood, the consequences of the security concepts are clear to the users.

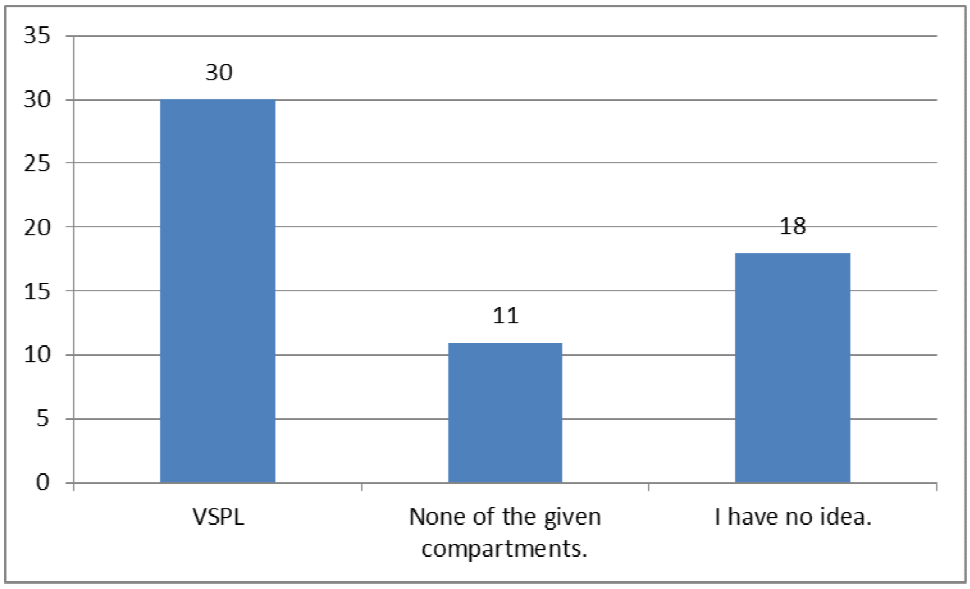


Figure 14 - Question 20: Which compartment(s) is/are protected against phishing attacks?

2.2.2.2.3 Questions regarding the security architecture, compartments and domains

Questions #11, #12, #15, #18 and #21 were created to test the participant's knowledge with focus on isolation and domains.

- Question #11 was formulated as follows: Provided that there's a virus in your WorkWindows compartment, what effect does the virus have on data in other compartments?

  90 out of 104 participants have responded to this question in a correct manner (see Figure 15). As the compartments are isolated from one another, the virus cannot infect data residing in other compartments.

- In question #12, participants were asked to state the definition of a Trusted Virtual Domain (TVD). This question was created as an open question, answers therefore varied largely across the 104 participants. Below is a list of (summarized) answers that were given:

  - *I don't know.* (50 participants)
  - *A protected area within the system.* (13 participants)

- *A TVD provides multiple security concepts and dependant of its use provides data and system security.* (6 participants)
- *Isolation of data across domains.* (5 participants)
- *A secure environment which provides data protection. Data cannot leave the TVD.* (5 participants)
- *Restricted area on the system with limited/restricted functionality and defined data entry points and exits.* (4 participants)
- *Complete isolation of distinct compartments.* (2 participants)
- *Secure internet connection.* (1 participant)
- *Only a TVD grants access to secure/private data.* (1 participant)
- *The security domain evaluates the system's security its development and design.* (1 participant)
- *A TVD is a form of access control to data within a network.* (1 participant)
- *A TVD is a compartment which provides remote access and methods for organization of distinct sub-compartments (i.e. Linux/Windows).* (1 participant)

- In question #15, participants were asked to name the compartments belonging to similar security domains. Each compartment was listed as an answer, together with "*None - every compartment has its own security domain.*". The participants could select multiple answers. 28 participants (28%) did choose the correct combination (WorkWindows and WorkLinux), while another 56 participants (54%) selected the last option – no two compartments are in the same security domain. The remaining participants have selected various incorrect compartment combinations.
- In question #18 the participants were asked whether it is possible to start and use more than one compartment simultaneously. The results (Figure 16) indicate that 91% of all participants provided the correct answer. Others selected "*no*" (2%) or "*I don't know*" (2%) or did provide no answer (5%).
- In question #21 we wanted to find out whether the participants have comprehended the implications of compartment colouring. The options presented were (a) "*To distinguish the security domains*" (b) "*To distinguish the compartments*" (c) "*For better presentation*" (d) "*I don't know*".

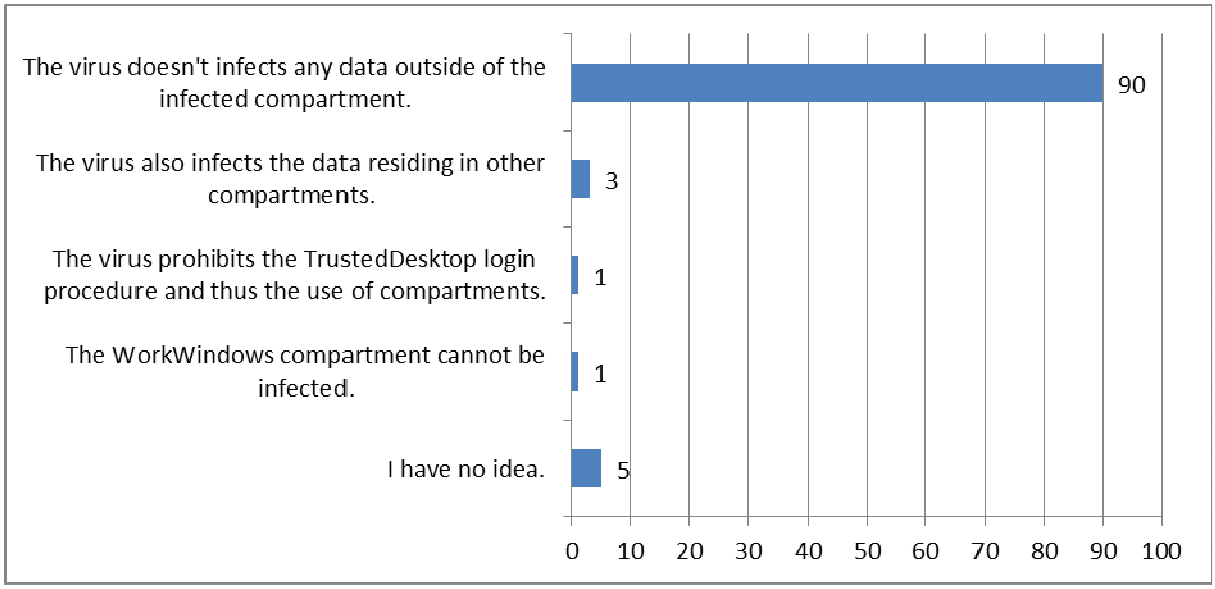


Figure 15 - Provided that there's a virus in your WorkWindows compartment, what effect does the virus have on data in other compartments?

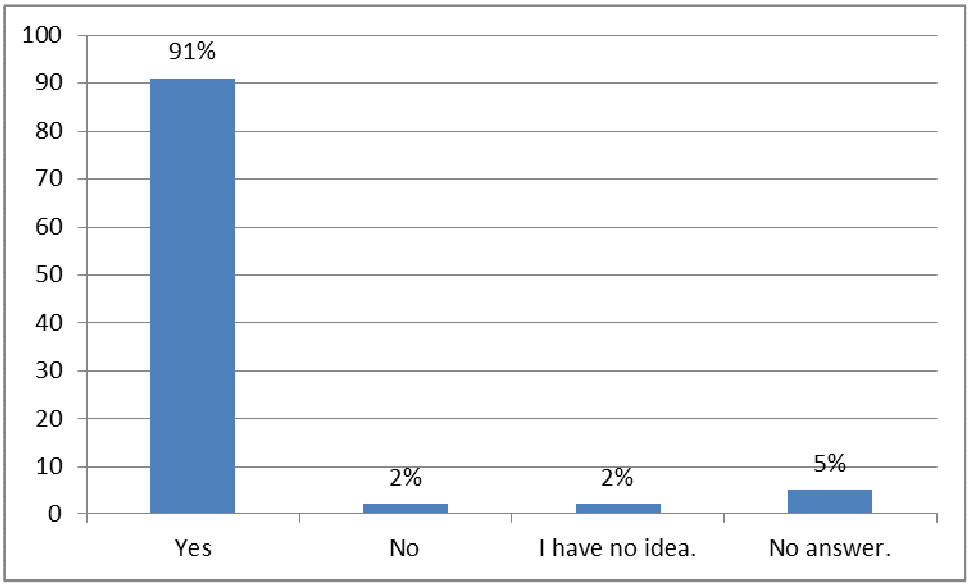


Figure 16 - Question 18: Is it possible to run and use more than one compartment at a time?

As the results show (see Figure 17), 49% of all participants have selected the correct answer.

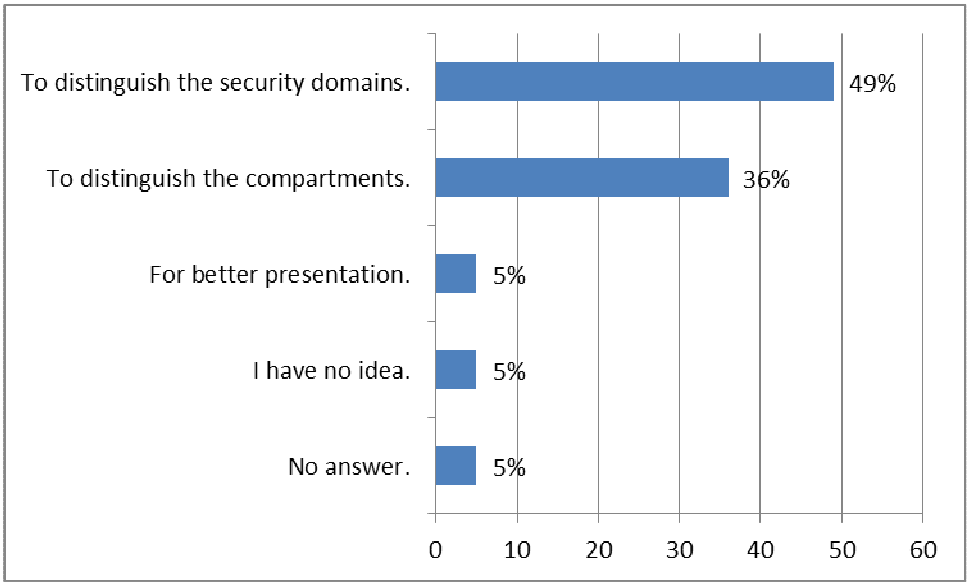


Figure 17 - Question 21: Why are the WorkWindows and the VSPL colored differently?

Table 2 displays the distribution of correctly answered questions. Similar to the first group of questions, there exists a notable fluctuation among the numbers which may be reasoned by the different nature of the questions. Another striking aspect to note is the fact that questions regarding the compartment colouring (#15 & #21) were answered in a correct manner by 28% and 49% respectively, which may lead to the assumption that an approximate two thirds of all participants are not familiar with the concept of compartment colouring.

	Question 11	Question 15	Question 18	Question 21	Average
Correct answers	86,5%	28%	91%	49%	63,6%

Table 2 - Summary of correctly answered questions

2.2.2.2.4 Data encryption and exchange

Questions #14, #17, #19 and #22 were created to test the user's knowledge regarding data exchange and encryption.

- The question #14 posed was as follows: You've found a cool video on YouTube. Select (one or more) valid options to share the link of the YouTube video with your friends.

  Selectable responses to this question were: (a) "*I would copy the address and send it to my friends via email from the BitBox compartment. They may then receive the email within their BitBox or Windows compartment and watch the video.*" (47 participants) (b) "*I would copy the address, switch to the Windows compartment and then send it via email. My friends receive the email within their Windows compartment and may watch the video therein.*" (36 participants) (c) "*Sharing the link with my friends is impossible.*" (13 participants) (d) "*I would copy the link to the Windows compartment and use that compartment to send it out. My friends will then again use their Windows compartment to read the email, copy the link to their BitBox compartment and watch the video therein.*" (11 participants)

The results to this question again were of mixed nature (Figure 18). One participant (1%) has selected the *two* correct answers ((a) and (d)). Another 47 participants have selected the first valid answer (a) and 11 participants selected the second right option ((d)).

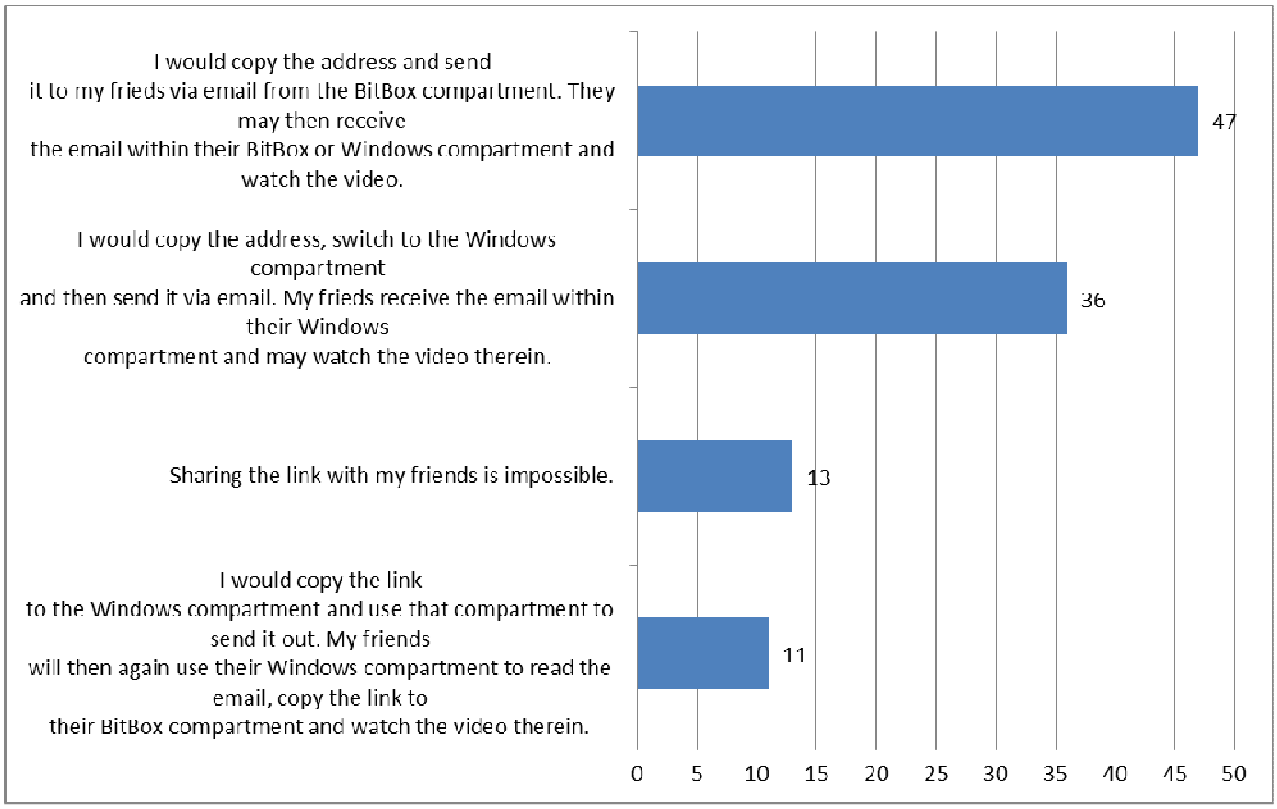


Figure 18 - You've found a cool video on YouTube. Select (one or more) valid options to share the link to the YouTube video with your friends.

- In question #17, participants were given the following scenario: Provided you have created a text file within the VSPL compartment and you want to send this file to one of your friends. Tick the compartment in which your friend will be able to read the text file.

  All compartment names were provided as selectable answers along with the option "*it is not possible at all*". Participants were asked to select one or more choices.

  The only correct option, namely *VPSL* was selected by 43 participants (41%). Approximately half out of this group (21.15% of all members) have selected *only* this option and have therefore provided the correct answer (Figure 19).

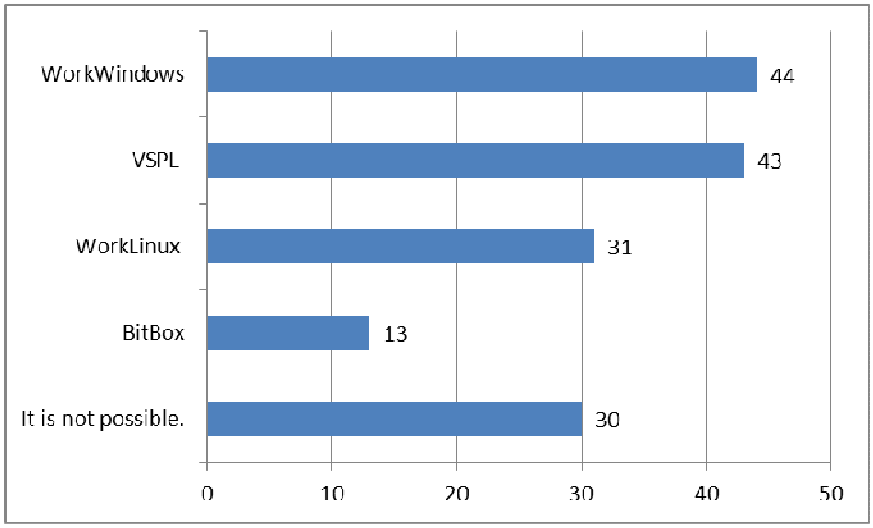


Figure 19 - Provided you have created a text file within the VSPL compartment and you want to send this file to one of your friends. Tick the compartment in which your friend will be able to read the text file.

- In question #19 we asked whether it is possible to exchange files between the compartments WorkLinux and WorkWindows. The correct answer (yes) was selected by 32% of all participants (see Figure 20). More than half of the participants doesn't have an idea about the possibility of file exchange – 51,4% of the frequent and 50,7% of the infrequent participants have selected the "I have no idea" option (see Figure 21). Excluding the clueless participants give us another results – 71,7% of the remaining people have indicated the correct answer (see Figure 22)

  51% of the participants have selected *I don't know* and 12% of the participants have selected *no*. 13 out of this group have reasoned their answer as follows:

  - *The two compartments are distinct virtual machines that have no knowledge about one another. (4 participants)*
  - *Windows and Linux are different operating systems. (2 participants)*
  - *There are no commonly shared folders among the two compartments. (7 participants)*

- In question #22 we provided a scenario in which the participants had to select exactly one option for their answer. The question was given as follows: You want to copy sensitive data from the VSPL compartment and send it to your professor in a trustworthy manner. Select one of the following answers.

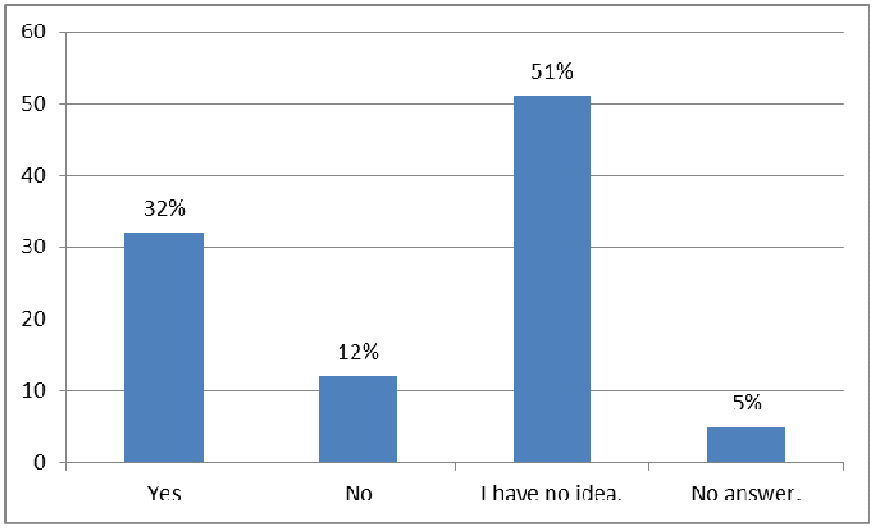


Figure 20 - Is it possible to exchange files between the two compartments WorkLinux and WorkWindows?

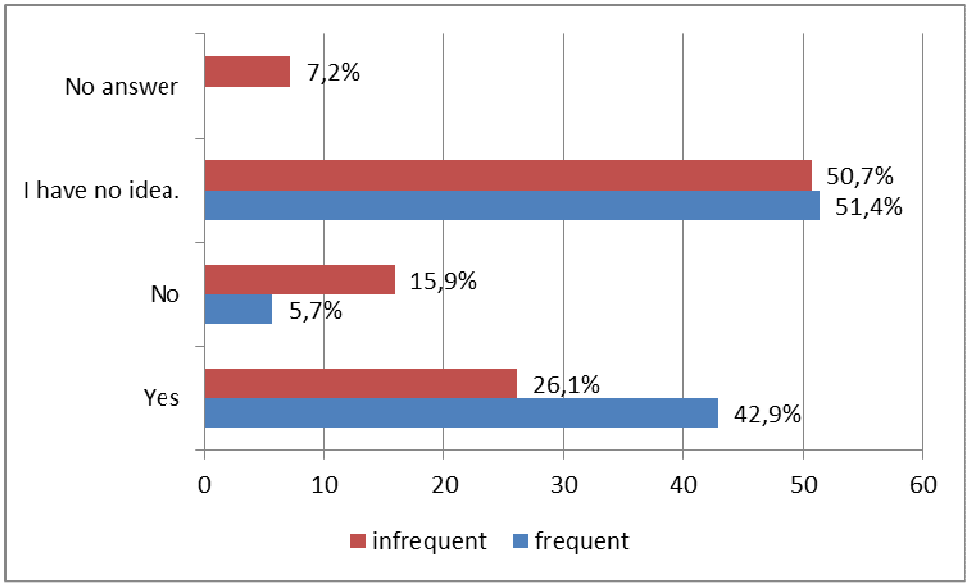


Figure 21 - Is it possible to exchange files between the two compartments WorkLinux and WorkWindows? (frequent vs. infrequent users)

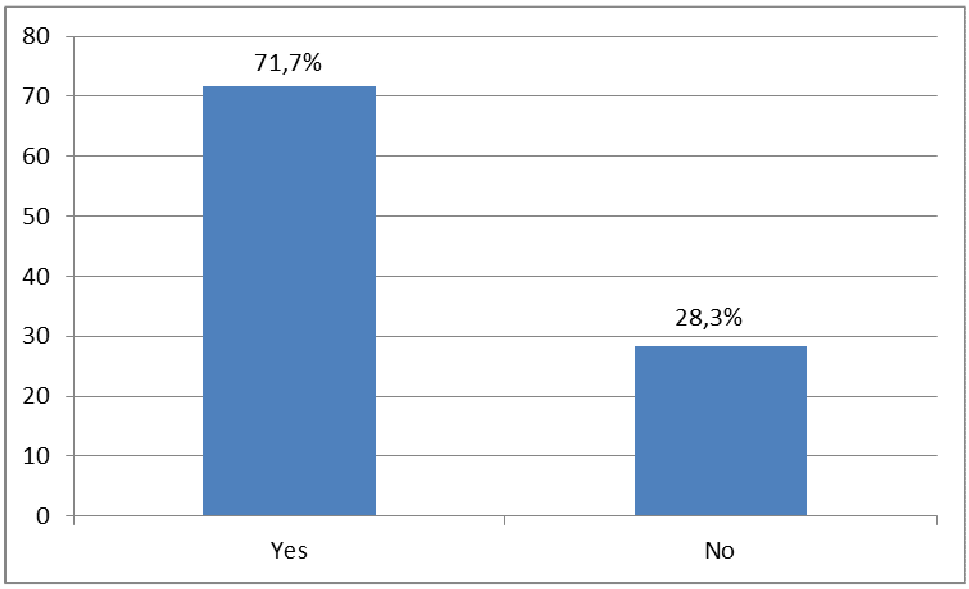


Figure 22 - Is it possible to exchange files between the two compartments WorkLinux and WorkWindows? (excluding clueless users)

The options for answers were given as follows:

- Store the data on a USB key and hand it to my professor. He/She may then review the data in his/her Windows compartment. (38 participants)
- Data exchange from the VSPL compartment in general isn't possible. (23 participants)
- Copy the data locally to my Windows compartment and use the Windows compartment to send it via Email to my professor. He/She may then view the data within his/her Windows compartment. (22 participants)
- Copy the data locally to my Windows compartment and use the Windows compartment to send it via Email to my professor. He/She may then copy the data from his/her Windows compartment to the VSPL compartment and view the data. (16 participants)

The correct answer was option d: Data is encrypted when copied to other TVDs and may only be decrypted by TVDs of same type, thus the only valid solution was option d. This answer was selected by 16 members (15.38%) of the study.

Table 3 summarized the results in this section of the survey. At average, 17.38% of all participants have selected a correct answer. 37.65% of all participants in this survey have provided answers to all or the majority of the questions, while others have either not selected any option at all or the option "*I don't know*" (which was only available for one question in this set of questions).

We try to provide some thoughts about why so few people remember the goals of the project and the information given during the introductory course. We checked if there is any correlation between the people who don't use the TrustedDesktop system regularly and the "I don't know" answers. After evaluating the results, we can conclude that there isn't such a correlation and the fraction of people who use the system regularly and "don't know" is the same as the fraction of those who don't use the system regularly and "don't know".

We have also found out that clueless participants are equally distributed among the degree programs, e.g., the participants who study social sciences do not tend to select the “I don’t know” option more often that the people who study IT.

A reason for the inability to provide an answer might be the fact that most of the participants only work with one compartment and haven’t experimented starting more. Also, the time frame between the introductory course and this questionnaire is over a year. Such a long time is actually enough to forget any then given and later not repeated information.

Many participants may feel obliged to fill out the survey and just select random answers as only one question of this set had the option “I have no idea”.

A lot of the participants have a second laptop at home and use the TrustedDesktop laptop very rare. It could be that they use the laptop even less than stated in the survey.

Again, the results varied significantly across different questions, possibly due to different levels in difficulty and due to different choices for options. At questions where the option “I don’t know” was unavailable, show a slightly different result tendency towards correctly answered questions. Also when we exclude the “clueless” answers from the evaluation the vast majority of the remaining answers is correct.

Question	14	17	19	22	Average
Correct answers	1%	21,15%	32%	15,38%	17,38%

Table 3 - Summary of questions regarding data exchange and encryption

To sum up, the results again indicate that more education and training on the novel security aspects of the architecture is needed and then the consequences become clear to the users.

2.2.3 Conclusion

With the TClouds field study on the TrustedInfrastructure Cloud we made a challenging study with end users to gain insight into the usability and comprehensibility of our novel and innovative security concepts of Trusted Virtual Domains. The participants used our TrustedDesktops as trustworthy end-points to access the services provided in the TrustedInfrastructure Cloud. This trustworthy end-to-end security offers security, privacy and trustworthiness on a nowadays unmatched level for commodity hardware and cloud offerings. The results of the study are quite promising and also indicate the direction of further research and development:

- The users were able to do their everyday work without being hindered by the security mechanisms. Some will voluntarily continue to use the system.
- As the security does not depend on the “proper usage” by the user the security goals to isolate domains and to control the information flow was guaranteed at all times.
- There is a need for further education on the security concepts and architecture (like Trusted Virtual Domains) to make the consequences understandable to the users. This is not surprising for a paradigm shift away from discretionary access control to pervasive information flow control.
- There is the need to further improve the user interface to make the security measures more comprehensible for the user.

2.3 Healthcare and Smart Lighting Use Cases Final End User Interviews and Questionnaires

In order to judge as better as possible the requirements that TClouds Infrastructure aims to achieve we have presented in D3.3.3 the surveys that A3 (namely, the Healthcare Scenario and the Smart Lighting System Scenario) was going to perform to their respective stakeholders.

The idea behind is that requirements' weighting has better relevance if comes directly from the business needs of final users.

As described in D3.3.3, the survey (whose results are described in the next paragraphs) have been produced trying to focus on judging the high-level requirements defined while building the cloud infrastructure.

2.3.1 Surveys' results

In this section we will show the survey results and how they have been conducted.

We will start by describing the healthcare Scenario and then we will move on to the SLS scenario.

2.3.1.1 Healthcare scenario

Healthcare survey has been conducted mainly online, involving the three stakeholders (Doctors, Patients and Developers). While for patients and for Developers has been a straightforward approach, with doctors we have more difficulties and we had to interview them one by one.

Healthcare surveyors are not a quantity statistically relevant, however we have a good approximation of the real stakeholders' thoughts related to security, transparency and availability of data into a cloud environment. Moreover, the face-to-face interview with the doctors have been extremely helpful to understand forces and strength of an approach lithe the Healthcare platform and survey results provides their tangible point of view. Also some important figures in the IT scenario of San Raffaele Hospital has been directly interviewed (such as the actual CIO (chief IT officer), the CMO (chief medical officer) and other IT responsible of smaller IT unity within San Raffaele) and they provided a brighter point of view of cloud adoption in healthcare realities.

2.3.1.1.1 Survey Demographics

We aimed to reach a number of 60 surveyed stakeholders, divided in 20 patients, 20 developers, and 20 doctors. We managed to reach 53 people surveyed, among which there has been:

- a face to face interview with the CIO of San Raffaele Hospital,
- a face to face interview with the CIO of Laboraf laboratories (that are specialized in blood analysis and work strictly with San Raffaele Hospital.
- a face to face interview with the Chef medical Officer of San Raffaele Hospital
- a face to face interview with an oncologist of the National Institute of Cancer Study

We managed to have the questionnaire filled by:

- 20 patients, composed mostly by any people (since anyone is/has been a patient, under different form), age between 25 and 50 years old
- 20 developers (evenly distributed among those developing software for hospitals and those developing commercial software. Were also present CIOs and CEOs.

- 13 doctors, among which, oncologists, general practitioners, neo-doctors, a Chef Medical Officer and First Aid doctors. Age between 30 and 65 years old

2.3.1.1.2 Surveys execution

In the following we will show the surveys outcome. Recalling the survey design described in D3.3.3, patients, developer and doctors have been interviewed or have filled up the questionnaire. The interviews have been done face to face with the related stakeholder and the outcome of the interview have been used to fill up the questionnaire and to feed D1.3.3 (business analysis of healthcare scenario).

All the interviewed stakeholders have been invited to see a small presentation tailored on the type of the stakeholder (that is, patients, developers and doctors were viewing similar presentation, customized on their point of view). Presentations can be seen here:

- Developer: http://www.slideshare.net/MarcoAbi/tclouds-t-paasdeveloperfinal
- Patients: http://www.slideshare.net/MarcoAbi/tclouds-t-paaspatientsfinal
- Doctors: http://www.slideshare.net/MarcoAbi/tclouds-t-paasdoctorsfinal

The questionnaires have been performed by using an ad-hoc installation of LimeSurvey tool (https://www.limesurvey.org/en/) (see Figure 23)

Figure 23 - TPaaS survey homepage

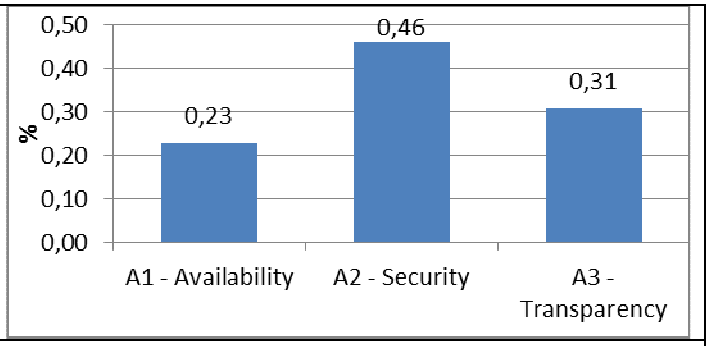
In D3.3.3 we introduced the Survey strategy and the scoring system for all the answers. The final results are now described

2.3.1.1.3 Patient Survey Outcome

The final score has been obtained weighting the rank of every question answer. To have more information on how to read the outcome tables and how the score is obtained, please refer to the Appendix 1.

QUESTION 1:
If you were using an online platform to log, save and share your personal health information, how important would the following topics be to you?

%	Score	Answer		1st	2nd	3rd
0,23	3,65	A1		1	11	8
0,46	7,40	A2		12	5	3
0,31	4,95	A3		7	4	9
		Weights		10	5	1

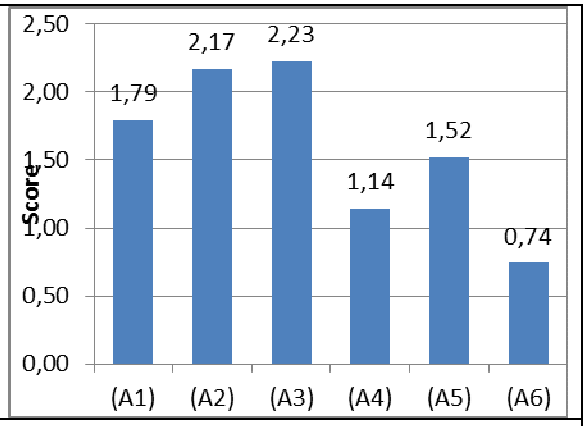


Where:
- A1: Availability 24/7 of my health information
- A2: Security of my health information (such as sharing rules and data encryption)

A3: Transparency and control of third parties accessing my information (third parties could be doctors, applications, friends and family)

QUESTION 2:
In scope of "Transparency and control of third parties accessing your information", how important are the following topics to you?

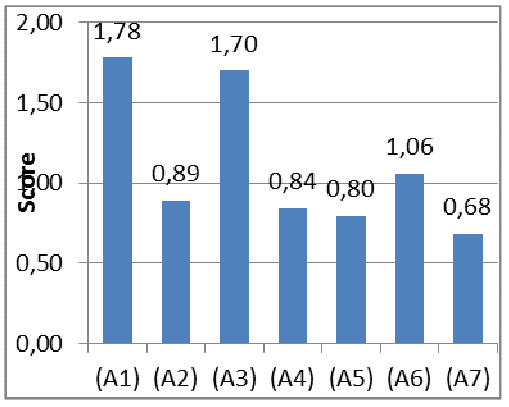
Score	Answer		1st	2nd	3rd	4th	5th	6th
1,79	(A1)		6	2	3	4	1	4
2,17	(A2)		6	6	0	8	0	0
2,23	(A3)		6	2	10	2	0	0
1,14	(A4)		0	3	3	3	9	2
1,52	(A5)		1	7	1	2	9	0
0,74	(A6)		1	0	3	1	1	14
	Weights		10	8	6	4	2	1



Where:
- A1: It must be possible for me to hide some information from my doctor.
- A2: It must be possible to add and remove persons who are allowed to access to my information.
- A3: The system must show who (and when) has been viewing or changing my health information
- A4: The system must show who (and when) has been viewing or changing the information of someone else I am allowed to access to
- A5: It must be visible to my doctor, when I have changed my information that I shared with him
- A6: I would be happy to give my data anonymized to third parties

QUESTION 3:
In scope of "Availability 24/7 of my health information", how important are the following topics to you?

Score	Answer		1st	2nd	3rd	4th	5th	6th	7th
1,78	(A1)		10	4	2	2	1	0	1
0,89	(A2)		0	3	3	5	3	1	5
1,70	(A3)		7	5	5	0	3	0	0
0,84	(A4)		0	3	1	6	1	8	1
0,80	(A5)		0	2	3	3	6	0	6
1,06	(A6)		2	3	4	0	3	8	0
0,68	(A7)		1	0	2	4	3	3	7
	Weights		10	8	6	4	3	2	1

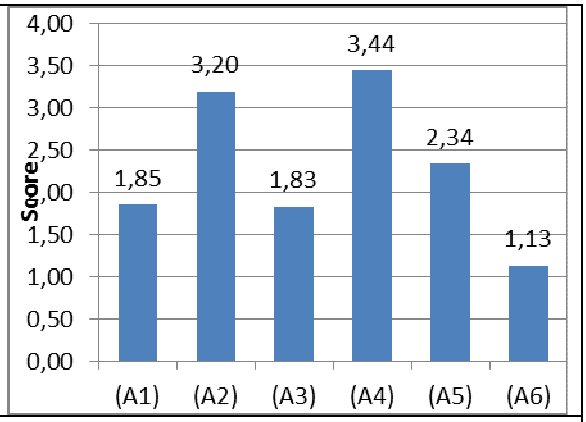


Where:

- A1: All my health information must be always available to my doctor
- A2: Only the most important health information must be always be available to my doctor
- A3: All my health information must always be available to me
- A4: The third party application I use to enter, view and edit my health information should always have access to my information on the platform
- A5: If the service is not available, then the third party application should being able to work anyway with a local copy of my health information.
- A6: There must always be enough space to hold my data
- A7: The loading time of a page in the apps I use has to be acceptable (eg. No more than 5 seconds)

QUESTION 4:
In scope of "Security of my data", how important are the following topics to you?

Score	Answer		1st	2nd	3rd	4th	5th	6th
1,85	(A1)		3	3	2	0	8	6
3,20	(A2)		8	3	5	4	0	2
1,83	(A3)		0	4	3	5	4	5
3,44	(A4)		9	2	1	5	1	0
2,34	(A5)		0	8	2	3	6	1
1,13	(A6)		0	0	7	3	1	6
	Weights		10	8	6	4	2	1



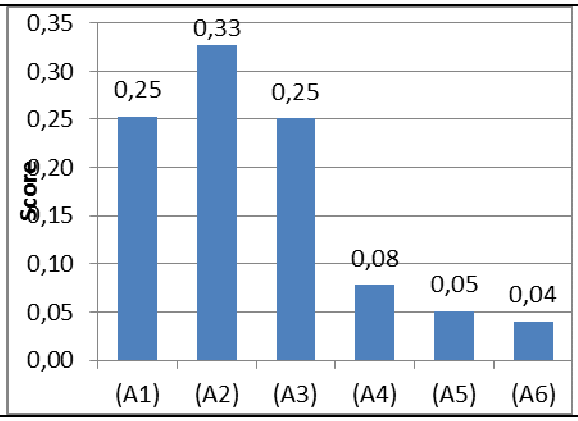
Where:

- A1: My data must be saved in location that are compliant with the legislation of my country
- A2: My data must be safe from attackers and data leakage
- A3: If an attacker is able to steal my health information, he can't read it anyway because they are encrypted
- A4: If I want, I have to be able to delete my data (no copies are maintained into the system)
- A5: If I want, I have to be able to delete my data and chose if I want them permanently deleted or anonymously deleted

A6: If I want to delete some health information that cannot be removed for legal issues (e.g. clinical data produced by an hospital), the system should stop me

QUESTION 5:
Please rank the following sentences:

Score	Answer		1st	2nd	3rd	4th	5th	6th
6,89	(A1)		6	0	9	2	1	0
8,94	(A2)		10	5	2	0	0	0
6,84	(A3)		2	12	1	0	4	0
2,13	(A4)		0	2	4	12	2	0
1,39	(A5)		0	0	2	6	12	0
1,10	(A6)		2	1	2	0	1	20
	Score		10	8	6	4	2	1



Where

- A1: I should been able to print directly from the web-based platform my health reports
- A2: It must be possible to change my data (e.g. medicine intake logs) on a later moment, for example when I forgot to enter it, or discover a mistake.

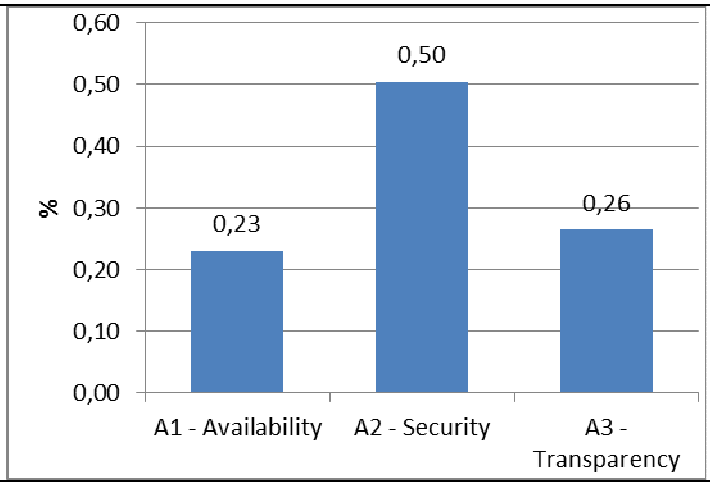
- A3: If I want I have to be able to export my data to take it into another service
- A4: I am willing to give my anonymous health information for scientific research
- A5: I am willing to give my anonymous health information for government policies
  A6: I am willing to give my anonymous health information for marketing research

2.3.1.1.4 Doctor Survey Outcome

The final score has been obtained weighting the rank of every question answer. To have more information on how to read the outcome tables and how the score is obtained, please refer to the Appendix 1.

QUESTION 1:
If you were using an internet platform to log, save and share your patients' data, how important would the following topics be to you?

Score	Answer		1st	2nd	3rd
3,69	A1		3	2	8
8,08	A2		8	5	0
4,23	A3		2	6	5
	Weight		10	5	1

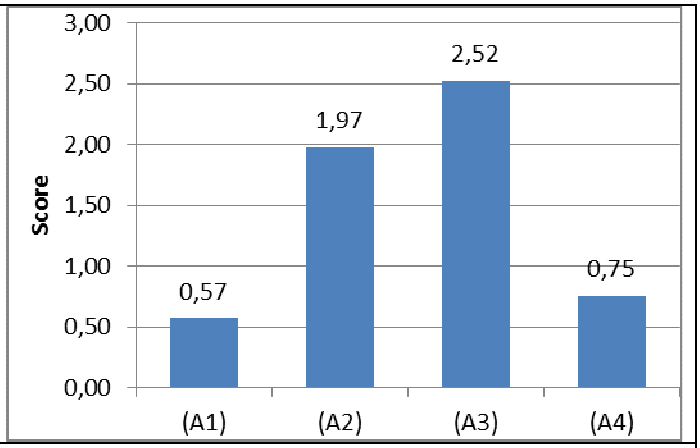


Where:
- A1: Availability 24/7 of my patients' health data
- A2: Security of my patients' health data

A3: Transparency and control of third parties accessing my patient's information (third parties could be doctors, applications, friends and family)

QUESTION 2:
In scope of "Transparency and control of third parties accessing Patient's health information", how important are the following topics to you

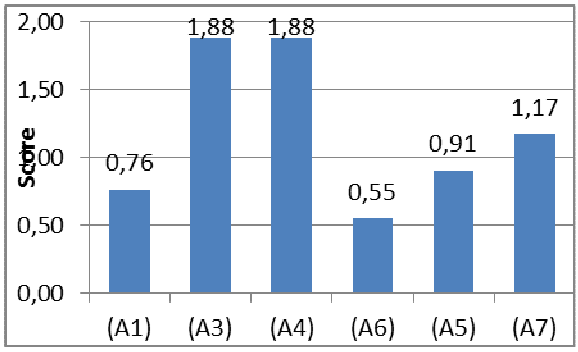
Score	Answer		1st	2nd	3rd	4th
0,57	(A1)		0	0	5	8
1,97	(A2)		2	11	0	0
2,52	(A3)		11	2	0	0
0,75	(A4)		0	0	8	5
	Weight		10	7	4	1



Where:
- A1: It must be possible for my patient to hide health information from me.
- A2: It must be visible to me, when my patient has changed his/her health information.
- A3: The system must show who (and when) has been changing the data I am allowed to access
- A4: It must be possible for my patient to audit access I did to his/her health information

QUESTION 3:
In scope of "Availability 24/7 of your patients' data", how important are the following topics to you?

Score	Answer		1st	2nd	3rd	4th	5th	6th
0,76	(A1)		0	0	2	6	2	3
1,88	(A3)		2	10	1	0	0	0
1,88	(A4)		8	1	2	1	1	0
0,55	(A6)		0	1	1	2	0	9
0,91	(A5)		0	1	4	2	5	1
1,17	(A7)		3	0	3	2	5	0
	Weight		10	8	6	4	2	1

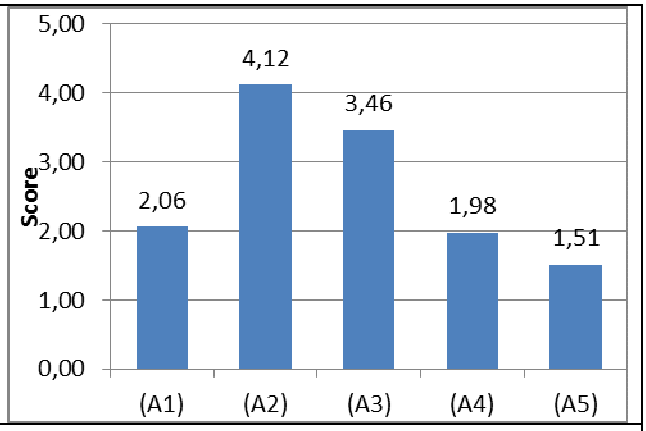


Where:

- A1: The loading time of a page the apps I use, may not be more than 5 seconds
- A3: The patient's health information must always be available to me
- A4: If the platform is not available, then the third party application should being able to work anyway with a local copy of my patient's health information
- A5: There must always be enough space to hold my patients' data
- A6: The patient's data must always be available to my patient
- A7: Given a patient, I must be able to specify which key info should be available 24/7 for him/her

QUESTION 4:
In scope of "Security of my patient's data", how important are the following topics to you?

Score	Answer		1st	2nd	3rd	4th	5th
2,06	(A1)		4	0	0	2	7
4,12	(A2)		7	5	0	0	1
3,46	(A3)		2	7	4	0	0
1,98	(A4)		0	0	8	3	2
1,51	(A5)		0	1	1	8	3
	Weight		10	7	5	3	1



Where:

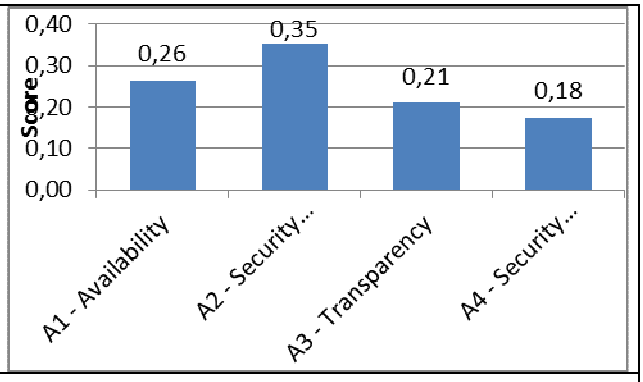
- A1: My patient's data must be saved in location that are compliant with the legislation of my country
- A2: My patient's data must be safe from attackers and data leakage
- A3: If an attacker is able to steal health information, he can't read them because they are encrypted
- A4: If patients want, they have to be able to delete their data (no copies are maintained into the system)
- A5: If patients want, they have to be able to delete their data and chose if they want them permanently deleted or anonymously deleted

2.3.1.1.5 Developer Survey Outcome

The final score has been obtained weighting the rank of every question answer. To have more information on how to read the outcome tables and how the score is obtained, please refer to the Appendix 1.

Question1
If you were using a cloud enabled platform to log, save and share health personal data, how important would the following topics be to you?

Score	Answer		1st	2nd	3rd	4th
5,80	A1		6	6	2	6
7,75	A2		10	5	5	0
4,60	A3		1	6	9	4
3,85	A4		3	3	4	10
	Weight		10	7	4	1

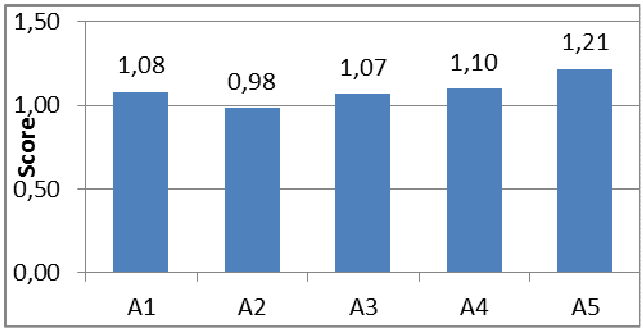


Where:

- A1: Availability 24/7 of health information into the system
- A2: Security of health information
- A3: Transparency and control of third party application accessing users' health information
- A4: Security of developer's data

QUESTION 2:
In scope of "Transparency and trust in other parties applications", how important are the following topics to you?

Score	Answer		1st	2nd	3rd	4th	5th
1,08	(A1)		1	7	6	4	2
0,98	(A2)		4	2	4	5	5
1,07	(A3)		4	3	4	6	3
1,10	(A4)		3	6	4	3	4
1,21	(A5)		8	2	2	2	6
	Weight		10	7	5	3	1

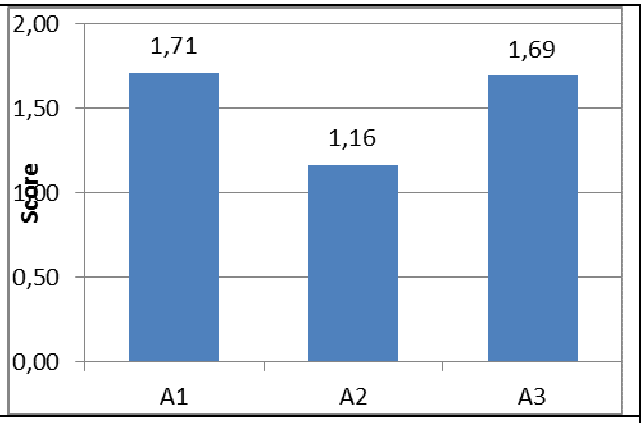


Where:

- A1: The platform must log the CRUD (Create, Read, Update, Delete) activity of third party application
- A2: It must be possible for a developer to audit all the activity that the application is doing on the platform
- A3: All log are depersonalized and does not allow developer to know the identity of who performed a certain activity
- A4: App's user Health information can be used (under user consensus) in a depersonalized form to extract meaningful data (such as for research or marketing purposes)
- A5: An application should not being able to provide/share user health information to other third parties (without user's consensus)

QUESTION 3:
In scope of "Availability and flexibility of your data and application", how important are the following topics to you?

Score	Answer		1st	2nd	3rd		
1,71	(A1)		14	0	9		
1,16	(A2)		2	10	5		
1,69	(A3)		4	10	6		
	Weight		10	5	1		

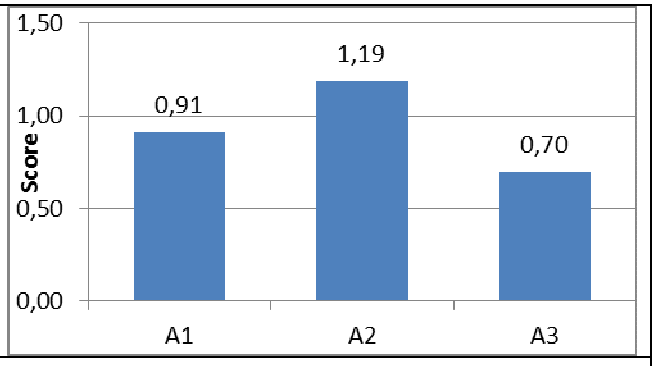


Where:

- A1: My third party application should always have access to the platform's API. The service should be available 24/7 with no downtime.
- A2: Applications have the chance to save locally data retrieved from the system
  A3: Data should always be available through duplication and distribution

QUESTION 4:
In scope of "Security of my data", how important are the following topics to you

Score	Answer		1st	2nd	3rd		
0,91	(A1)		8	3	9		
1,19	(A2)		8	11	1		
0,70	(A3)		4	6	10		
	Weight		10	5	1		

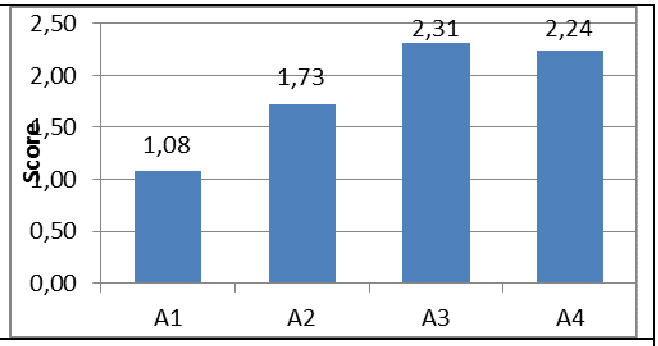


Where:

- A1: When deleting the developer account, all my (developer) details must be delete
- A2: When a developer deletes an app, all the data of the given app should be deleted (not the health data, but only the data related to the app itself
- A3: When a developer deletes an app. all the health data generated by the app should be deleted as well

QUESTION 5:
In the scope of “Security of health information"

Score	Answer		1st	2nd	3rd	4th	
1,08	(A1)		0	6	4	9	
1,73	(A2)		5	4	5	6	
2,31	(A3)		7	4	5	2	
2,24	(A4)		8	6	6	3	
	Weight		10	7	4	1	



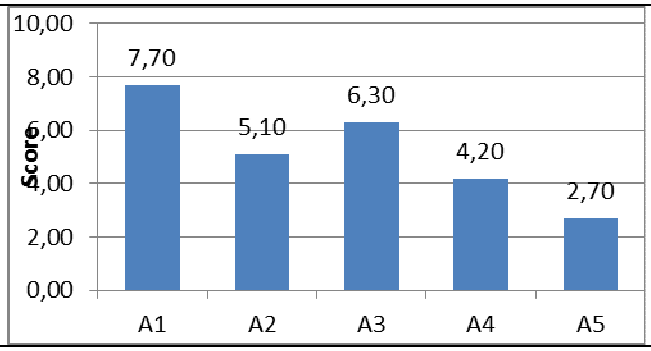
Where:

- A1: I must be able to define the minimum policy requirements that an app user has to accept in order to use the application properly
- A2: Health information should be always encrypted and decryption keys are not in developers hand
- A3: Local storage is encrypted and decryption keys resides into the platform itself. The

developer uses a specific library provided by the system in order to be able to decrypt and use the information
- A4: User's data must be saved in location that are compliant with the legislation

QUESTION 6:
Score the sentence

Score	Answer		1st	2nd	3rd	4th	5th
7,70	(A1)		10	6	0	4	0
5,10	(A2)		0	8	6	5	1
6,30	(A3)		6	4	7	0	3
4,20	(A4)		4	2	2	4	8
2,70	(A5)		0	0	5	7	8
	Weight		10	7	5	3	1



Where:
- A1: You are interested in developing consumer application (related with PHR data)
- A2: You are interested in developing professional application (related with EHR data)
- A3: You are interested in connect devices to the platform
- A4: Build an application that is able to get user health information that comes from other applications/devices other than yours
- A5: Build an application knowing that the data that the app saves into the platform can be shared with other application

2.3.1.1.6 Surveys conclusion

PATIENTS

Transparency

Results about this topic shows how patient are incline to know more about their data usage from other people. While they generally wants to know more about their personal data usage, they tends to don't care much about data shared from others with them.

Availability

Results about this topic shows that patients consider availability of data to doctors important as well as data availability to themselves. However not much can be say about service availability, with seems to be a minor issue

Security

Security plays an important role in regards of patient users. First of all they highly prefer to have an "exit strategy", they want to have the possibility to remove their account and all their data into it. This is also an interesting business driver. Secondly, as expected and as normal thought, they want overall security of the platform, preserving their data from attackers and data leakage. There is a chance that users are willing to leave their data anonymously into the platform whenever they want to un-register.

For patients, having an "exit strategy" represent a good option to jump into the service, moreover, they feel the need to have total control over their data (from add to remote, to change it). However, we noticed a very low interest in sharing data in a "broadcasted" fashion, in which data may be used for other purposes such as scientific research, marketing or government purposes. Despite this, we think that discovering new business driver that move people to share their data, could be an interesting approach in order to find new adoption schemes.

DOCTORS

Doctors, like patients, place security as the most important factor that a cloud platform may have (and, in general, any internet service). Transparency and Availability have shown two main different groups. The first composed of General Practitioners and doctors working more comprehensively on the whole health status. Such categories prefer to have transparency more than availability, in detail, they prefer to know whether a patient is changing/adding/deleting data. The second group we can find specialized doctors, first Aid Specialist and researchers that prefer availability over transparency.

Transparency

For a doctor, transparency means "to have the ability to see when and which data their patient is changing". Either if the data s changed directly by the patient or by someone else that has granted this right.

Doctors don't like to allow patients to hide health data to them and don't like (or don't care) whether their access is monitored by the patients as well.

Availability

In regards of availability, doctors consider very important to have the patients data always available (First-Aid Doctors were pushing more on this aspect), this aspect has been used as driver for business analysis as well (please refer to D1.3.3 for more details on SCR[1] health records). For specific health condition (e.g.: drug allergies, diabetes, etc...) doctors have expressed their interest to have the chance to flag which data is considered a must-to-have into a Summary Care Record. They don't care much, instead, about giving to the patient the ability to access data as well.

Security

Security plays an important role, and doctors care about security of patients data. This is a bit in counter-tendency in respect to the previous question, probably due to the fact that in our interviewed group we may find doctors that have legal implication on patient's data treatment, and they care of patients data as much as they care to their legally issues.

DEVELOPERS

As natural thought, also developers place security as the most important aspect of a cloud system. They also care to have high availability (understandable, since having downtime of the systems means not being able to provide a reliable service to their direct clients) and transparency. They feel more to place in a secure fashion more users' data than their data itself.

Transparency

For developers transparency is a broad concept in this sense we have found an overall interest in any transparency feature.

Availability

In regards of Availability for developers, once the platform is able to provide 100% uptime, they don't really feel the need to have local storage for the app itself. Of course this would replay to the internet connection the availability issue. Developers feels also the need to have availability through duplication and distribution.

Security

[1] SCR: Summary Care Record. It represents the bare minimum information that a doctor needs in order to continue with any medical activity.

Under the umbrella of the security, developers place at the first place the "exit strategy" of app deletion. They understand that data is a value and they prefer to remove all the data of the app and maintain the generated data. Also developer account is felt as sensitive data, and by removing the developer account they want that all data should be deleted as well.

We were expecting to have a low interest in deleting all the data that the app generates (while deleting an app) instead we found an interesting quota of developers that prefer to remove ALL the data while removing the app. This is because within the interviewed there were professional EHR/hospital SW houses, feeling the concept of data removal a broader concept that embraces also legal implications and customer needs.

In regards of the security of patient data, developers like the possibility to have a tool able to manage local data storage and data transmission to the platform. This is might be due because specific API to be used into the third party application allows a faster time-to-market and simplifies the application development. Also geo-location of data and legal compliancy of data management

2.3.1.2 Smart Lighting System scenario

2.3.1.2.1 Updated Strategy

In D3.3.3 we identified and characterized three different stakeholders for the smart lighting use case: municipalities, utilities and vendors. These stakeholders are interrelated through their business areas. Vendors supply products and services to the utility which are used to provide services to the municipality. By other words, the utility is a client of the vendor and the municipality is a client of the utility. In Portugal, public lighting also works in this way; Portuguese municipalities attributed a number of concessions to EDP for exploitation and maintenance. Public lighting is an important subject of concern to municipalities and to the utilities in terms of personal welfare, security and cost efficiency. Public lighting costs have a great impact on municipality budgets once they are translated directly into the electrical bill that needs to be paid. On the other hand, public lighting is considered a factor that contributes to the safety of persons, property and the society in general. While analyzing what would be the best approach for presenting the survey to each stakeholder, we decided that we would take three different approaches. Regarding municipalities, the approach would be: 1) personal interview, 2) presentation of the online questionnaire and 3) online reply. Regarding utilities and vendors: 1) contact by phone call, 2) presentation of the online questionnaire by email and 3) online reply. Further details are given in the next sections.

2.3.1.2.2 Utilities and vendors

Within the TClouds consortium there is an utility (EDP) and a public lighting vendor (EFACEC). We decided to take advantage of this situation; therefore, we prepared two identical online questionnaires. The utilities' questionnaire is available at http://www.surveymonkey.com/s/B23HYSK and it was filled by EDP, while the vendors' questionnaire is available at http://www.surveymonkey.com/s/BYRBXHC and it was sent by email to EFACEC, which was asked to fill it and to forward the link to other possible valuable contributors.

2.3.1.2.3 Municipalities

The TClouds consortium not includes any municipality. In facts, municipalities are expected to have low knowledge of the information security area. Moreover, although they are involved in several projects with EDP, they are not aware of the work that we are developing in TClouds. We decided to approach municipalities with direct interviews in order to present the TClouds project and to raise their awareness towards privacy and cyber security issues. This would also allow us to manage expectations towards the smart lighting solution, which is important for preventing a bad impact in EDP-municipality business relationships. Interviews included a guided walk through the survey questions in order to answer any doubts and we

also provided a link to the online survey. We also encouraged our contacts to forward this link internally to other possible valuable contributors. The municipalities' questionnaire is available at http://www.surveymonkey.com/s/8KWLN2G. Reminder emails were also sent after the interviews took place in order to foster more contributions.

2.3.1.2.4 Interviews

We consulted EDP's Board of Directors in order to decide which municipalities we would contact. The involvement of the Board was required due to the sensible relationship between EDP and Portuguese municipalities. We decided to contact the municipality of Évora (http://www.cm-evora.pt/en) in Alto Alentejo, and we also decided to contact an energy agency in Algarve (http://www.areal-energia.pt/). On May 13th we traveled more than 700km and spent a whole day in order to conduct two interviews of two hours each in Évora and Faro (see Figure 24).

Figure 24 - Map of Portugal from Google maps; A marks the starting point; B and C mark interview locations

2.3.1.2.4.1 Évora

Évora is a Portuguese city in the interior with around 56.000 inhabitants. Many tourists come to Évora every year for culture richness as it is well known for its churches, museums and Roman ruins. Public lighting is mainly used to light the streets and many monuments. It was chosen for the survey because its municipality is already aware and collaborates with EDP's most innovative projects. The largest EDP smart metering pilot project with more than 31.000 smart meters is in Évora (http://www.inovcity.pt/en/Pages/homepage.aspx) and it is very well received by the municipality and its inhabitants. In the near future new functionalities will be implemented such as a new public lighting management system. Its functional specification is the basis for smart lighting's specification. The main difference is that it does not use cloud computing nor TClouds' security and resilience features.

We met with the municipality Engineer that is responsible for EDP's projects. We presented TClouds and our survey's objectives which were received with high interest. We used the first questions as example for explaining basic privacy and cyber security concepts and the different levels of the security scale that is presented in the questionnaire. We found that municipalities' concerns are higher and much closer to utilities' concerns than we first thought, which is reflected in the detailed results in the next section. We left the online survey link in his possession, we made ourselves available to answer any doubts regarding any other question and we left to the next interview location.

2.3.1.2.4.2 Faro

Algarve is the south-most province of Portugal with 451.000 inhabitants. Its beaches to the Atlantic Ocean, warm water and sunny summer days make it perhaps the most attractive Portuguese location for foreign tourists. In the summer days, streets, restaurants, bars and casinos are full in the evening and through the night. Public lighting is important to make Algarve's night life secure for people who walk the streets. In 2014 EDP's smart metering pilot project will be extended by 100.000 new smart meters to seven new locations, including Faro and Olhão in Algarve.

We met with an Engineer from the energy agency AREAL, Agência Regional de Energia e Ambiente do Algarve, in Faro. This agency acts as adviser and representative for municipalities in the province of Algarve. The initial approach was similar to what we used in Évora. We started by presenting the TClouds project and then we continued to the questionnaire. We had two objectives for this interview: 1) to take advantage of the agency's experience with the municipalities by asking them to answer to the questionnaire themselves and 2) to ask them for collaboration in the process of distributing explaining the questionnaire to Algarve's municipalities. AREAL was really interested in the project and agreed with both requests. They also encouraged future collaboration. Once again, we left the online survey link in his possession and we made ourselves available to answer any doubts regarding survey questions.

2.3.1.2.5 Smart Lighting System survey conclusion

Question ID		Corresponding requirements	Average value to municipality	Average value to utility	Average value to vendor
Q1		ASSECREQ1	5	10	10
Q2		ASSECREQ2	6	9	10
Q3		ASSECREQ3	5	8	10
Q4		ASSECREQ4	8	10	10
Q5		ASSECREQ5	9	10	10
Q6		ASSECREQ6	3	10	10
Q7		ASSECREQ6	6	9	10
Q8	S1	ASSECREQ2 ASSECREQ3			x
	S2	ASSECREQ4	x	x	
Q9	S1	ASSECREQ1 ASSECREQ5	x	x	x
	S2	ASSECREQ6			
Q10	Yes	ASSECREQ3	x	x	x
	No	ASSECREQ4			

Table 4 - Smart lighting average survey answers represents the average of survey answers received from May 13th to August 31st.

As described in D3.3.3, questions #8 to #9 allow us to compare requirements with one another and to validate the obtained ratings. In details:

- **#8.** Answers to this question show that municipalities and utility value resilience/ intrusion tolerance (REQ4) more than intrusion prevention (REQ2, REQ3) which is compliant with prioritization results. Smart lighting is part of the smart grid critical infrastructure which must still be able to operate and also to maintain its integrity when a part of it is damaged and/ or fails. People's lives and other infrastructures depend on it. These answers also show that municipalities and utilities are able to understand the benefits of complex resilience concepts such as those that are presented by the state machine replication (BFT-SMaRt) component, although they are fairly new and difficult to understand. The answers of vendors show a different perspective. Although intrusion prevention and resilience/ intrusion tolerance are rated with equal values, when asked to choose between one and the other, they choose the first option. This is understandable. The systems that they provide should be secure to avoid intrusions in the "first line of defense". Resilience is equally important and should be there just in case but it should not be necessary to use it.
- **#9.** Answers to this question show that the respondents value integrity (REQ1 and REQ5) more than availability (REQ6) which is compliant with prioritization results. It is important that information is available but, if it does not maintain its integrity, it might lead to wrong and potentially dangerous decisions.
- **#10.** All respondents answered "Yes" to this question, which means they all understand that the balance between intrusion prevention (REQ3) and intrusion detection (related with REQ4) in smart lighting depends on available security technologies (available TClouds' components), despite of REQ4 being more valued then REQ3.

As final result, obtained prioritization ratings are trustworthy. Therefore they are also valuable for evaluating the Smart Lighting System use case.

2.4 Priority tables

In order to build the priority table of all the requirements, we mapped all the answers into the related requirements.

Recalling Table 3 of D3.3.3 where we mapped the questionnaire answers with the requirements addressed, we built the following mapping table (Table 5) for healthcare:

Requirement #			LREQ1	LREQ2	LREQ3	LREQ4	LREQ5	AHSECREQ1	AHSECREQ2	AHSECREQ3	AHSECREQ4	AHSECREQ5	AHSECREQ6	AHSECREQ7	AHSECREQ8	AHPRIVREQ1
Survey for Developer	Q2	S1	0	0	0	0	1,077	0	1,077	0	0	0	1,077	0	1,077	0
		S2	0	0	0	0	0,983	0	0,983	0,983	0	0	0,983	0,98	0,983	0,983
		S3	0	0	0	0	0	0	0	0	0	0	0	0	0	1,066
		S4	0	0	0	0	0	0	0	0	0	0	0	0	0	1,098
		S5	0	0	0	0	0	1,213	0	0	0	0	0	0	0	1,213
	Q3	S1	0	0	0	0	0	0	0	0	1,708	1,71	0	0	0	0
		S2	0	0	0	0	0	0	0	0	1,163	1,16	0	0	0	0
		S3	0	1,691	0	0	0	0	0	0	0	0	0	0	0	0
	Q4	S1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
		S2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
		S3	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Requirement #			LREQ1	LREQ2	LREQ3	LREQ4	LREQ5	AHSECREQ1	AHSECREQ2	AHSECREQ3	AHSECREQ4	AHSECREQ5	AHSECREQ6	AHSECREQ7	AHSECREQ8	AHPRIVREQ1
	Q5	S1	0	0	0	1,075	0	0	0	0	0	0	1,075	0	0	0
		S2	1,726	0	0	0	0	1,726	0	0	0	0	0	0	0	0
		S3	2,309	0	0	0	0	0	0	0	0	0	0	0	0	0
		S4	0	0	2,236	0	0	0	0	0	0	0	0	0	0	0
	Q6	S1														
		S2														
		S3														
		S4														
		S5														
Survey for Patients	Q2	S1	0	0	0	0	0	0	0	0	0	0	0	1,79	0	0
		S2	0	0	0	0	0	0	0	0	0	0	0	2,17	0	0
		S3	0	0	0	0	2,228	0	2,228	0	0	0	2,228	2,23	0	0
		S4	0	0	0	0	1,145	0	1,145	0	0	0	1,145	1,14	0	0
		S5	0	0	0	0	1,516	0	1,516	0	0	0	1,516	1,52	0	1,516
		S6	0	0	0	0	0	0	0	0	0	0	0	0	0	0,743
	Q3	S1	0	0	0	0	0	0	0	0	1,779	0	0	0	0	0
		S2	0	0	0	0	0	0	0	0	0,89	0	0	0	0	0
		S3	0	0	0	0	0	0	0	0	1,7	0	0	0	0	0
		S4	0	0	0	0	0	0	0	0	0	0,84	0	0	0	0
		S5	0	0	0	0	0	0	0	0	0	0	0	0	0	0
		S6	0	0	0	0	0	0	0	0	0	0	0	0	0	0
		S7	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Q4	S1	0	0	1,85	0	0	0	0	0	0	0	0	0	0	0
		S2	0	3,195	0	0	0	0	0	0	0	0	0	0	3,195	0
		S3	1,828	0	0	1,828	0	1,828	0	0	0	0	0	0	0	0
		S4	0	0	0	0	3,443	0	0	0	0	0	0	0	0	0
		S5	0	0	0	0	2,336	0	0	2,336	0	0	0	0	0	2,336
	Q5	S1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
		S2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
		S3	0	0	0	0	0	0	0	0	0	0	0	0	0	0
		S4	0	0	0	0	0	0	0	0	0	0	0	0	0	2,128
		S5	0	0	0	0	0	0	0	0	0	0	0	0	0	1,388
		S6	0	0	0	0	0	0	0	0	0	0	0	0	0	1,103
Survey for Doctors	Q2	S1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
		S2	0	0	0	0	0	0	1,973	0	0	1,97	1,973	1,97	0	0
		S3	0	0	0	0	0	0	2,522	0	0	2,52	2,522	2,52	0	0
		S4	0	0	0	0	0	0	0	0	0	0,75	0	0,75	0	0
	Q3	S1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
		S2	0	0	0	0	0	0	0	0	1,882	1,88	0	0	0	0
		S3	0	0	0	0	0	0	0	0	0	0	0	0	0	0
		S4	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Requirement #			LREQ1	LREQ2	LREQ3	LREQ4	LREQ5	AHSECREQ1	AHSECREQ2	AHSECREQ3	AHSECREQ4	AHSECREQ5	AHSECREQ6	AHSECREQ7	AHSECREQ8	AHPRIVREQ1
		S5	0	0	0	0	0	0	0	0	0,905	0,91	0	0	0	0
		S6	0	0	0	0	0	0	0	0	1,172	1,17	0	0	0	0
	Q4	S1	0	0	2,058	0	0	0	0	0	0	0	0	0	0	0
	Q4	S2	4,116	4,116	0	4,116	4,116	4,116	0	4,116	0	0	0	0	4,116	0
	Q4	S3	3,456	0	0	3,456	0	0	0	0	0	0	0	0	0	0
	Q4	S4	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Q4	S5	0	0	0	0	0	0	0	0	0	0	0	0	0	1,514

Table 5 - Overview of healthcare requirement priority given by the average of each question mapped. Please note Question 6 of the survey to the developer that does not find mapping with requirements since its outcome has more value for the business perspective. Mapping has been taken from Table 3 of D3.3.3

The numbers in the cells corresponds to the final answer score for the given requirement.. All the requirements' score have been averaged in order to obtain the prioritization map as shown below in Table 6:

Requirement #	Value/asset #1 **Value for patients** Factor	Value/asset #2 **Value for doctors** Factor	Value/asset #3 **Value for developer** Factor	Rating	Variance
LREQ1	1,83	3,79	2,02	2,54	1,166174
LREQ2	3,20	4,12	1,69	3,00	1,498826
LREQ3	1,85	2,06	2,24	2,05	0,037356
LREQ4	1,83	3,79	1,08	2,23	1,958067
LREQ5	1,83	4,12	1,03	2,32	2,566386
AHSECREQ1	1,83	4,12	1,47	2,47	2,061526
AHSECREQ2	1,63	4,12	1,03	2,26	2,678168
AHSECREQ3	2,34	4,12	0,98	2,48	2,469781
AHSECREQ4	1,46	1,32	0,98	1,25	0,059381
AHSECREQ5	0,84	1,53	1,44	1,27	0,139356
AHSECREQ6	1,63	2,25	1,04	1,64	0,361674
AHSECREQ7	1,77	1,75	0,98	1,50	0,201199
AHSECREQ8	3,20	4,12	1,03	2,78	2,510563
AHPRIVREQ1	1,54	1,51	1,09	1,38	0,063181

Table 6 - Final prioritization table for the three healthcare stakeholders

The questionnaire outcome and the mapping between the answers and the requirements leads to the table above, that shows the final requirement prioritization.
Please consider that Table 6 does not represents "how much" a single requirement is important, but has to be used as "judgment tool" in order to understand (in the case of some

requirements are not completely satisfied) the impact that the given requirement may have in relation with A3 stakeholders.

2.4.1 Smart Lighting System prioritization table

Survey results allowed us to prioritize security requirements as listed in Table 7. It must be noted that all SLS requirements have been rated from 4 to 10, being that trustworthy communications is the highest rated requirement, which is closely followed by resilience.

Requirement name	Requirement ID	Value to municipalities	Value to utility	Value to vendor	Priority rating
Trustworthy Audit	ASSECREQ1	5	10	10	8
Trustworthy infrastructure	ASSECREQ2	6	9	10	8
Trustworthy persistence engine	ASSECREQ3	5	8	10	8
Resilient	ASSECREQ4	8	10	10	9
Trustworthy communications	ASSECREQ5	9	10	10	10
High performance & scalable	ASSECREQ6	4	9	10	8

Table 7 - Smart lighting prioritization table

Chapter 3

Validation Activity Results

Chapter Authors & contributors:
Marco Abitabile (FCSR), Martin Deutschmann, Sebastian Ressi (TEC), Sören Bleikertz (IBM), Norbert Schirmer (SRX), Mihai Bucicoiu (TUDA), Alysson Bessani, Marcel Santos (FFCUL), Paolo Smiraglia, Roberto Sassu (POLITO), Johannes Behl and Klaus Stengel (TUBS)

3.1 Activities for Healthcare scenario

3.1.1 Crypto as a Service Validation activity

In this chapters is executed the validation activity of Crypto as a Service component. The validation activity as described in D3.3.3 has been further refined reaching the state described in Table 8.

Activity ID	**SBS+SVM_1**
Activity type	Proof of concept
Activity description	The Home Healthcare appliance is deployed and running onto the Trustworthy OpenStack TClouds prototype. The Home Healthcare databases VMs are encrypted according to the description in the reference documents 1- Import the certified key and encrypt the Home Healthcare VMs images a. get the public key of the TPM from the TClouds server b. use Crypto_aaS python script that encrypts the VM HDD 2- Upload the encrypted and the unencrypted images into the Trustworthy OpenStack TClouds prototype 3- Check the images nature and verify their states 4- Launch the unencrypted Virtual Machines 5- Check that the hard disk is accessible from the administration side by hexdump it on a specific pattern 6- Stop the unencrypted VM and launch the encrypted version 7- Test on the Xen node, that an administrator cannot access the HDD (because it's encrypted) without the key, and the key is never revealed. Try to hexdump it on a specific pattern 8- Stop the encrypted VM 9- Look for encryption key on the whole hosts hard disk 10- Check that in idle state the encrypted disk still remains encrypted 11- Check that proxy –domc instance is properly intercepting all the IO activity.
Acceptance Criteria	The activity is passed if either point 5 reveal the pattern into the disk while on point 7 it fails
References Documents:	(TClouds factsheet 03 - Cryptography) (Deliverable D2.1.2, 2012) (Deliverable D2.4.2, 2012)

Requirements Satisfied	LREQ1, LREQ2, AHSECREQ1, AHSECREQ2

Table 8- Validation activity outline

3.1.1.1 Crypto as a service features

Crypto as a Service (CaaS) component is intended to increase cloud user security and privacy my adopting cryptographic techniques. It allows establishment of secret-less client VMs and securely separate client's cryptographic primitives and credentials. Crypto as a Service enhances security standards within the cloud infrastructure reducing risks such as:

- external attacks which exploit vulnerabilities in web service deployed within the client VM
- malicious co-located Clients which might compromise the isolation between Virtual Machines
- insider Attackers at the provider side which exploit their privileges.

CaaS can be configured to be used in two different ways:

- As Secure Virtual Device: forms a transparent layer between the client VM and peripheral devices (storage disk or network card) and encrypts all I/O data streams to/from those devices similar to full-disk encryption or Virtual Private Networks (VPN). We also use this layer as a convenient building block to protect the VM images and VM states during pro-visioning to the cloud (i.e., the client uploads only encrypted images to the cloud), migration be-tween cloud nodes (i.e., the VM state is transferred only in encrypted form between cloud nodes), and storage.
- As Virtual Security Module: emulates a virtual hardware security device, like an HSM, attached to the client VM. The client's workload can leverage this security module as a secure credential storage to protect his high value cryptographic keys from unauthorized access by external attackers or insider attacks. Additionally, the client can also load custom trusted code into the security module and leverage it as a customized crypto engine, e.g., to securely maintain SSL/TLS communication channels by outsourcing the security-sensitive key management operation into the isolated Virtual Security Module.

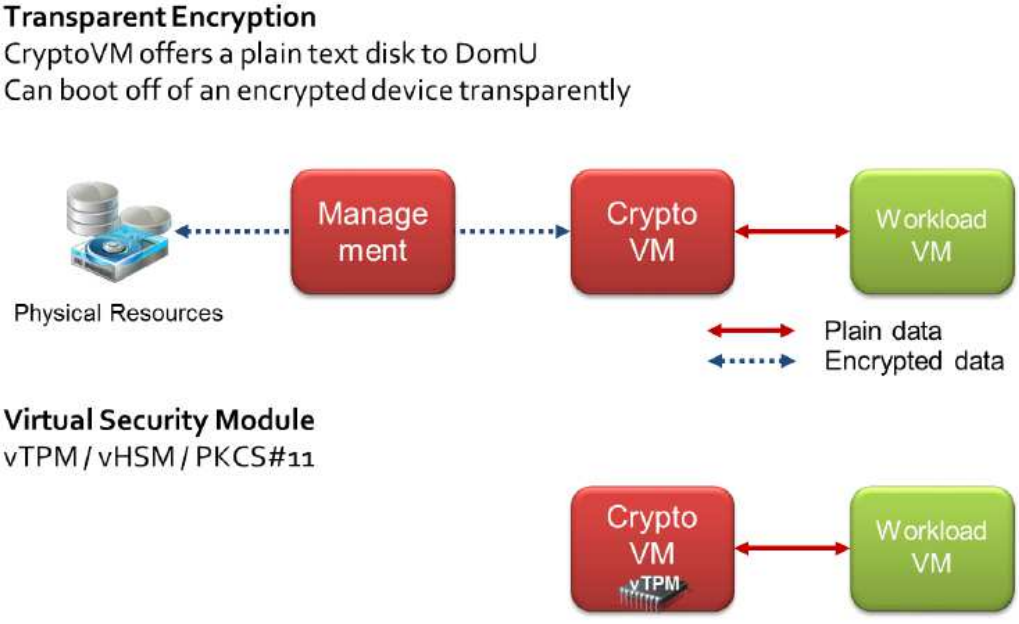


Figure 25 - CaaS configuration

3.1.1.2 Validation scenario

In order for the validation to proceed the following scenario has been taken into place. CaaS has been configured to work as **Secure Virtual Device**. In this mode the HDD of the VM is encrypted with a one-time generated key. In order to be able to start a VM using this HDD, the key is bind with the TPM and a particular state of the software, i.e., the Xen hypervisor used is tailored such that all memory management and TPM related functionality is move to DomT to block potential attacks from the administrator.

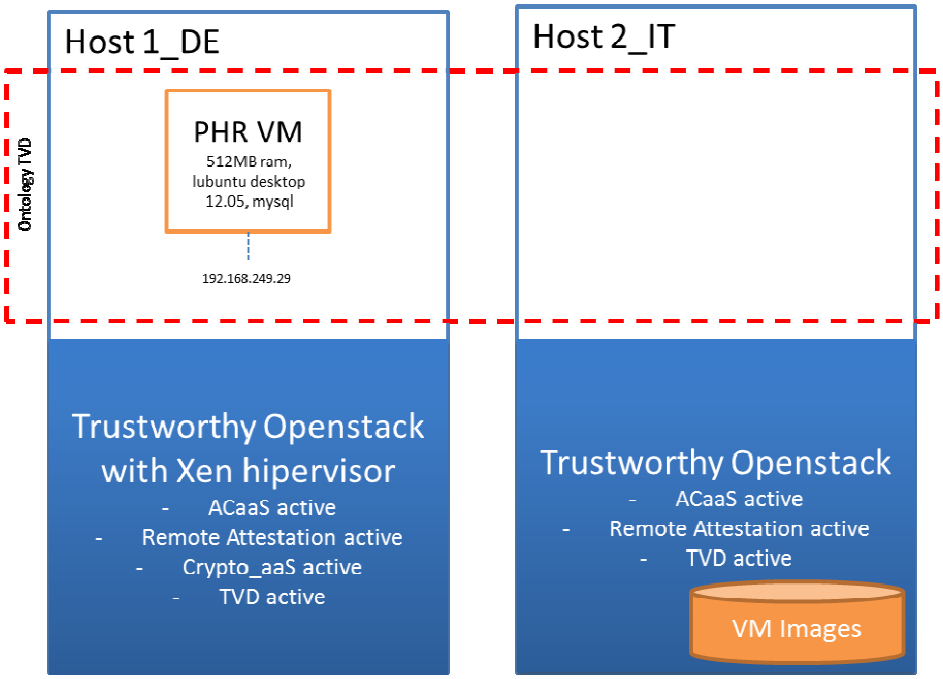


Figure 26 - Validation scenario

Figure 26 shows the environment used in order to validate Crypto as a Service component.

It has been used two hosts, one (Host1_DE) that is a physical machine in which run TClouds Trustworthy OpenStack prototype with Xen hypervisor, the other (Host2_IT) that is another physical machine that runs TClouds Trustworthy OpenStack with KVM hypervisor.

Host2 act as the main controller of TClouds Infrastructure and it contains all the images available on the cloud (among these the PHR VM image resides as well).

Since the encrypted PHR VM needs to run on the Xen node (Host1), at instantiation time ACaaS component will force sending the image from Host2 to Host1.

The Virtual Machine used in this scenario is the PHR VM of TPaaS Healthcare appliance. Recalling it from Deliverable D.3.1.3, PHR VM consists of a remote MySql database that stores all the PHR data of TPaaS Healthcare appliance.

3.1.1.3 Validation setup

CaaS is a component completely transparent for the VM. There are just little tweaks to be done in order to make CaaS to work properly.

The PHR VM used contains only one partition. At this stage of the development Crypto as a service may create inconsistency states if the VM HDD has more than one partition in its hard disk.

PHR virtual machine needs to be encrypted beforehand, this is done by obtaining the public key of TMP module of Host1 and the private key of the user owning the Virtual Machine. This allows Host1 only to be able to decipher it and start it. The details of this procedure are detailed into the next chapter.

3.1.1.4 Validation execution

In this chapter is implemented the validation activity itself. Each point in Table 8 is executed and exposed in order to collect evidences that will support the conclusion chapter.

3.1.1.4.1.1 Encryption of the VM

Encryption of the PHR virtual machine is done by using the TPM module's public key. This step can be done remotely, on the user site, not in the cloud.

CaaS's encryption features consist in a python script that creates the encrypted Image and produces the decryption key to allow TPM to decipher it.

The encryption is a straightforward process that consists in:

```
//deployer.py [-h] [-k KEYFILE] in-vm out-vm out-vmcb
#deployer.py –k domt_pubkey.bin PHR.vmdk PHR_crypto.vmdk vmcb.key
```

Where:

- [input] domt_pubkey.bin is the TPM public key of Host1 (host where the VM will be started)
- [input] PHR.vmdk is the hard disk of PHR virtual machine
- [output] PHR_crypto.vmdk is the encrypted hard disk
- [output] vmcb.key is the private key used for HDD encryption, encrypted with the TPM public key to provide to Host1

Clear and cyphered hard disk files have been uploaded on TClouds infrastructure.

```
# glance image-create  --is-public true --disk-format qcow2 --container-
format bare --name "PHR-Encrypted" --caas_domc 1 < VM/disk.img

+------------------+--------------------------------------+
|     Property     |     Value                            |
+------------------+--------------------------------------+
|     caas_domc    |     True                             |
|     checksum     |     f9b6df588a1bc99ea0ef487c2fc67210 |
|     container_format     |     bare                     |
|     created_at   |     2013-09-18T21:14:00              |
|     deleted      |     False                            |
|     deleted_at   |     None                             |
|     disk_format  |     qcow2                            |
|     id           |     f910ead9-f60f-4c31-b283-18bf3af592b9     |
|     is_public    |     True                             |
|     min_disk     |     0                                |
|     min_ram      |     0                                |
|     name         |     PHR-Encrypted                    |
|     owner        |     None                             |
|     protected    |     False                            |
|     size         |     595591168                        |
|     status       |     active                           |
|     updated_at   |     2013-09-18T21:14:36              |
```

```
+------------------+--------------------------------------+

# glance image-show f910ead9-f60f-4c31-b283-18bf3af592b9

+------------------+--------------------------------------+
|     Property     |     Value                            |
+------------------+--------------------------------------+
|     caas_domc    |     True                             |
|     checksum     |     f9b6df588a1bc99ea0ef487c2fc67210 |
|     container_format |     bare                         |
|     created_at   |     2013-09-18T21:14:00              |
|     deleted      |     False                            |
|     disk_format  |     qcow2                            |
|     id           |     f910ead9-f60f-4c31-b283-18bf3af592b9 |
|     is_public    |     True                             |
|     min_disk     |     0                                |
|     min_ram      |     0                                |
|     name         |     PHR-Encrypted                    |
|     protected    |     False                            |
|     size         |     595591168                        |
|     status       |     active                           |
|     updated_at   |     2013-09-18T21:14:36              |
+------------------+--------------------------------------+
```

All the necessary tests will be done directly into the host with administrative privileges (see Listing 27)

```
hsr@ubuntu:~/Crypto_aaS$ ls -l
total 1510696
-rwxrwxr-x 1 hsr  hsr        7215 Aug  2 10:46 deployer.py
-rw-rw-r-- 1 hsr  hsr         284 Aug  2 10:46 domt_pubkey.bin
-rw-rw-r-- 1 hsr  hsr  1546920448 Aug  2 10:49 PHR.vmdk
-rw-rw-r-- 1 hsr  hsr        4558 Aug  2 10:46 Re R R R validation demo.txt
-rw-r--r-- 1 root root        256 Aug  6 09:01 vmcb_key.key
hsr@ubuntu:~/Crypto_aaS$ sudo python3 deployer.py -k domt_pubkey.bin PHR.vmdk /media/secondDisk/PHR_crypto.vmdk vmcb_key.key
*** TU-Darmstadt CaaS/SBS cloud deployer ***

Python: 3.2.3 (default, Apr 10 2013, 06:11:55)
[GCC 4.6.3]

STAGE 1: encrypting VM
----------------------
Encrypting file of 1.4GB: 0%...25%...50%...75%...100%  in 109s (13.6MB/s)
=> 3021329 ESSIV-encrypted sectors written from PHR.vmdk to /media/secondDisk/PHR_crypto.vmdk

STAGE 2: encrypting symmetric key
---------------------------------
read 284 bytes from domt_pubkey.bin
(TSPI) Attempting: creating object which holds RSA pubkey...
(TSPI) Attempting: loading the RSA pubkey...
(TSPI) Attempting: creating object which holds plaintext...
(TSPI) Attempting: binding plaintext to the pubkey...
(TSPI) Attempting: extracting the ciphertext...
(TSPI) success: bind() finished. Length of ciphertext: 256 bytes
256 bytes written with RSA ciphertext to vmcb_key.key
graceful exit
hsr@ubuntu:~/Crypto_aaS$ sudo ccrypt -d -K domt_pubkey.bin /media/secondDisk/PHR_crypto.vmdk
ccrypt: /media/secondDisk/PHR_crypto.vmdk: key does not match -- unchanged
hsr@ubuntu:~/Crypto_aaS$ sudo ccrypt -d -K vmcb_key.key /media/secondDisk/PHR_crypto.vmdk
ccrypt: /media/secondDisk/PHR_crypto.vmdk: key does not match -- unchanged
hsr@ubuntu:~/Crypto_aaS$ file PHR.vmdk
PHR.vmdk: VMware4 disk image
hsr@ubuntu:~/Crypto_aaS$ file /media/secondDisk/PHR_crypto.vmdk
/media/secondDisk/PHR_crypto.vmdk: data
hsr@ubuntu:~/Crypto_aaS$
```

Listing 27 – cyphering the PHR hard disk

3.1.1.4.1.2 Check images sent on TClouds infrastructure

By accessing on TClouds dashboard is possible to see both images, encrypted and unencrypted, have been uploaded (Figure 28).

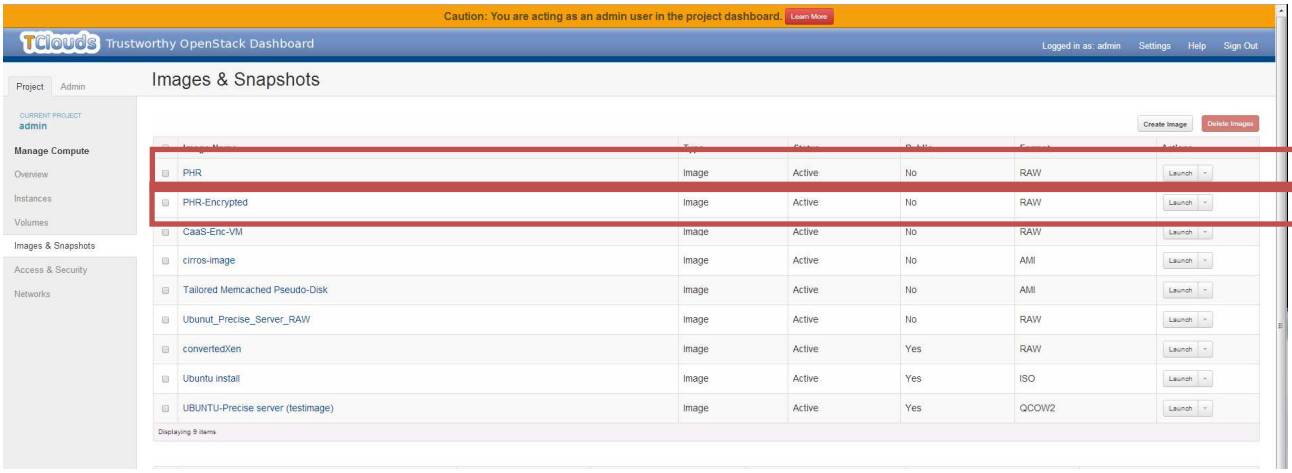


Figure 28 - List of VM available on TClouds infrastructure

The image above shows the actual VM images present into TClouds infrastructure. Among the others can be noticed the PHR and PHR-Encrypted images.

By accessing with administrative privileges it is also possible to run console commands that shows more information (Listing 29).

Notice that administrative privileges are not provided to cloud customer.

```
root@tclouds-stack:/home/marco# glance image-list
+--------------------------------------+---------------------------------+-------------+------------------+------------+--------+
| ID                                   | Name                            | Disk Format | Container Format | Size       | Status |
+--------------------------------------+---------------------------------+-------------+------------------+------------+--------+
| 01aa6be5-2177-45a7-9cfa-3778c04585a3 | convertedXen                    | raw         | bare             | 6422528000 | active |
| 0221ba3d-ab19-48fb-a1b1-bf0e2d7e6bfe | UBUNTU-Precise server (testimage) | qcow2       | bare             | 251527168  | active |
| 0385f905-9bc7-4450-bd0e-0cd83ab612e1 | Tailored Memcached (DHCPfix v1) | aki         | aki              | 5393720    | active |
| 4b33c3f8-3cd0-4241-b4f3-3dfdf329bd14 | PHR                             | raw         | bare             | 7515144192 | active |
| 845b1059-1abe-4f86-954a-0cd0e4aad474 | CaaS-kernel                     | aki         | aki              | 2833216    | active |
| 98602c6e-8b1e-4512-b003-909a3b106590 | Ubunut_Precise_Server_RAW       | raw         | bare             | 2147483648 | active |
| 9ca06268-734a-44f4-a4ef-36a07f25bcad | CaaS-Enc-VM                     | raw         | bare             | 595591168  | active |
| ba6c5f2c-4e11-4185-baf2-576f4dfd863e | Tailored Memcached              | aki         | aki              | 5393656    | active |
| bb0558c6-ce8a-4583-8d4c-94b19c3877ae | cirros-kernel                   | aki         | aki              | 4731440    | active |
| cb0d9307-7d57-4d8a-bfe8-e31682970a8b | cirros-image                    | ami         | ami              | 25165824   | active |
| df924066-c4d8-4ec6-abe4-fd6c70fed69f | Tailored Memcached Pseudo-Disk  | ami         | ami              | 1048576    | active |
| f9741a40-abf8-4788-bf4a-a6c111c2961e | PHR-Encrypted                   | raw         | bare             | 7515144192 | active |
+--------------------------------------+---------------------------------+-------------+------------------+------------+--------+
root@tclouds-stack:/home/marco#
```

Listing 29 - List of images present on the Cloud Infrastructure, host2 node

The Image above shows the results of the command `glance image-list` this command shows all the images present on the infrastructure. Please notice, again the PHR and PHR encrypted images with their respective ID.

Below, instead, can be seen the physical image disks files:

```
root@tclouds-stack:/var/lib/glance/images# ls -l
insgesamt 23985640
-rw-r----- 1 glance glance 6422528000 MÃ¤r 21 14:15 01aa6be5-2177-45a7-9cfa-3778c04585a3
-rw-r----- 1 glance glance  251527168 MÃ¤r  5 17:04 0221ba3d-ab19-48fb-a1b1-bf0e2d7e6bfe
-rw-r----- 1 glance glance    5393720 Jul 16 12:29 0385f905-9bc7-4450-bd0e-0cd83ab612e1
-rw-r----- 1 glance glance   25165824 Feb 28 18:25 08b017a9-fed6-4062-9446-9c050ce698cc
-rw-r----- 1 glance glance 7515144192 Aug 23 12:32 4b33c3f8-3cd0-4241-b4f3-3dfdf329bd14
-rw-r----- 1 glance glance    2254249 Jul  5 09:33 7a4e25da-fe72-4f6c-9e88-90ae0a4ad6b7
-rw-r----- 1 glance glance    2833216 Jul 11 19:55 845b1059-1abe-4f86-954a-0cd0e4aad474
-rw-r----- 1 glance glance 2147483648 MÃ¤r 21 14:25 98602c6e-8b1e-4512-b003-909a3b106590
-rw-r----- 1 glance glance  595591168 Jul 12 13:40 9ca06268-734a-44f4-a4ef-36a07f25bcad
-rw-r----- 1 glance glance    5393656 Jul 15 16:46 ba6c5f2c-4e11-4185-baf2-576f4dfd863e
-rw-r----- 1 glance glance    4731440 Jul  5 09:32 bb0558c6-ce8a-4583-8d4c-94b19c3877ae
-rw-r----- 1 glance glance   25165824 Jul 23 10:37 cb0d9307-7d57-4d8a-bfe8-e31682970a8b
-rw-r----- 1 glance glance    1048576 Jun 26 14:25 df924066-c4d8-4ec6-abe4-fd6c70fed69f
-rw-r----- 1 glance glance 7515144192 Aug 22 14:56 f9741a40-abf8-4788-bf4a-a6c111c2961e
-rw-r--r-- 1 root   root            8 Aug 23 17:47 magic.mgc
root@tclouds-stack:/var/lib/glance/images# file 4b33c3f8-3cd0-4241-b4f3-3dfdf329bd14
4b33c3f8-3cd0-4241-b4f3-3dfdf329bd14: Linux rev 1.0 ext4 filesystem data, UUID=345e8f90-6307-453a-b79c-2f9bdab00799
(needs journal recovery) (extents) (large files) (huge files)
root@tclouds-stack:/var/lib/glance/images# file f9741a40-abf8-4788-bf4a-a6c111c2961e
f9741a40-abf8-4788-bf4a-a6c111c2961e: data
root@tclouds-stack:/var/lib/glance/images#
```

Listing 30 - list of all the image files into TClouds Infrastructure, host2 node

To check the nature of the two files we run `file` command that reads the mime-type of the file to determine its content:

```
root@tclouds-stack:/var/lib/glance/images# file 4b33c3f8-3cd0-4241-b4f3-3dfdf329bd14
4b33c3f8-3cd0-4241-b4f3-3dfdf329bd14: Linux rev 1.0 ext4 filesystem data, UUID=345e8f90-6307-453a-b79c-2f9bdab00799
 (needs journal recovery) (extents) (large files) (huge files)
```

Listing 31 - File command on the clear PHR image

```
root@tclouds-stack:/var/lib/glance/images# file f9741a40-abf8-4788-bf4a-a6c111c2961e
f9741a40-abf8-4788-bf4a-a6c111c2961e: data
root@tclouds-stack:/var/lib/glance/images#
```

Listing 32 - file command on the cyphered PHR image

The two images above shows the information of the non-cyphered PHR file and of its cyphered version.

In order to check the effective encryption of the two images, it has been chosen to perform an hexdump of the two image files by looking for a specific pattern that we know is present into the PHR VM. In particular we want to look for the word: “PATIENT_PHR”. Since the PHR VM consists in a simple linux machine with a remote MySql server running, PATIENT_PHR word has been used to define a specific table within the database.

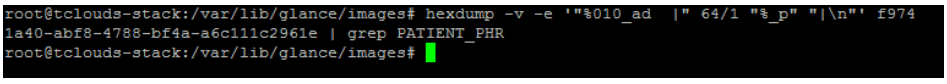
```
root@tclouds-stack:/var/lib/glance/images# \
> hexdump -v -e '"%010_ad  |" 64/1 "%_p" "|\n"' 4b33c3f8-3cd0-4241-b4f3-3dfdf329bd14 | \
> grep PATIENT_PHR
```

We obtained this result:

Listing 33 - output of hexdump command on clear PHR image

It can be seen that clearly the content of the disk can be read and, moreover, it consists of the actual MySql table definition.

Instead, by running the same command on the encrypted image we obtained the following result:



Listing 34 - hexdump command and output on encrypted image disk

That confirms that the file contains data that has been completely scrambled.

3.1.1.4.1.3 Launching the clear VMs and checking its live disk

Up to now we discovered that the two images that has been uploaded on TClouds infrastructure have been correctly recognized by Trusted Infrastructure. Moreover, by inspecting the content of the HDD, stored in the OpenStack database, we have seen that clearly the unencrypted VM's content is accessible at administration level while the encrypted one is not.

It is now time to launch the images. We will start with the clear image, we will enter its console and we will write a file. Then we will check the file content by “hexdumping” the live hard disk that the VM is using to look for the content previously written into the file.

Before proceed with the launch we can check which other instances are running via the admin command line:

```
root@xen-compute:/home/marco# xl list
Name                                        ID   Mem VCPUs      State   Time(s)
Domain-0                                     0  1023     8     r-----    2627.2
Domain-T                                     1  1024     1     -b----       2.0
root@xen-compute:/home/marco#
```

Figure 35 - list of the “domains” virtual machines available into TClouds Infrastructure

We can see here the Domain-T, which is responsible, in CaaS, for all the memory management operations, i.e., Dom0 will contact it for any memory-related functions.

The image can be started directly from the Trusted OpenStack dashboard:

1- first select the image (PHR) and give a name (PHR_SanRaffaele) and select in which TVD the Virtual Machine should be

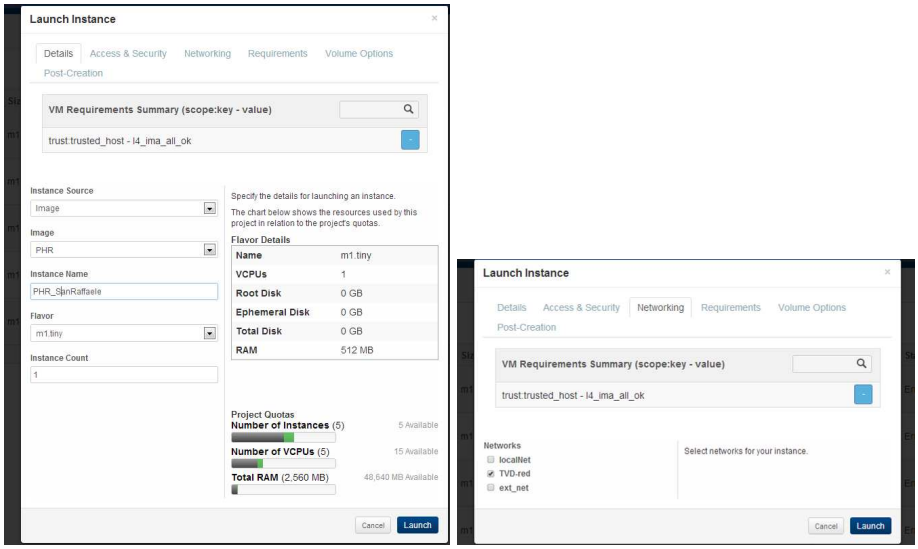


Figure 36 - PHR VM deployment (1)

2- Define the requirement for this VM. In this case we want the VM to run on the Xen node

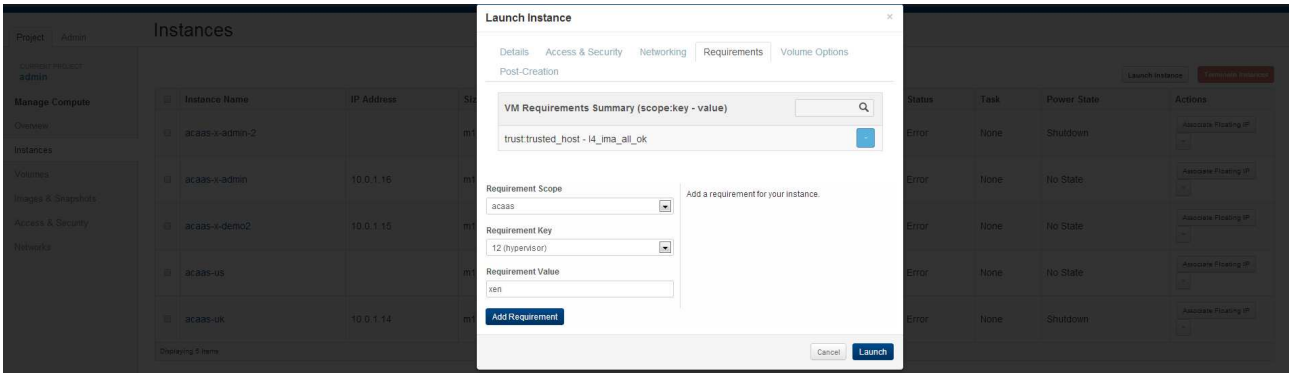


Figure 37 - PHR VM deployment (2)

3- Then launch it

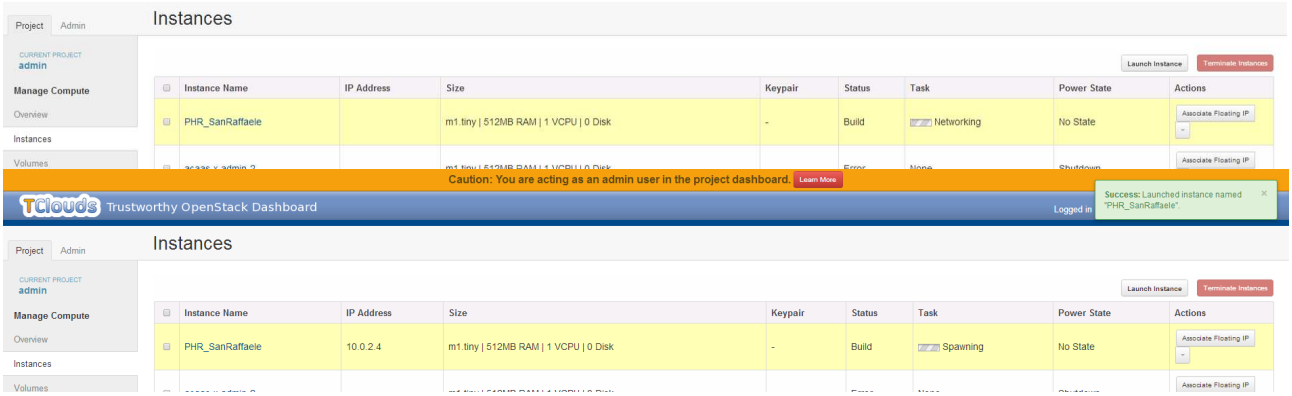


Figure 38 - Networking and Spawning phase of PHR VM deployment

The VM takes a while to start due to the fact that the image is flying from one host (Host2) to the other (Host1), with a HDD of 7GB.

4- Once the VM is running we can use the administration privileges in order to control the VM and check its hard disk

We can check its running state also from the admin command line:

```
root@xen-compute:/home/marco# xl list
Name                                        ID   Mem VCPUs      State   Time(s)
Domain-0                                     0  1023     8     r-----    3355.4
Domain-T                                     1  1024     1     -b----       2.5
instance-000000f6                           12   512     1     -b----       5.2
instance-000000f6-domc                      13    64     1     r-----      55.0
root@xen-compute:/home/marco#
```

Figure 39 - VMs running into the infrastructure

In this case `instance-000000f6` is our PHR VM whose state is “running”. Crypto as a Service component works in such a way that another special VM is started every time a normal VM is started as well. In our case `–domc` VM is a sort of empty container since the VM we have started is a clear VM, that has not been ciphered.

Please note that `–domc` VM will be analyzed and discussed in the next steps, when the cyphered VM will be started.

In order to access the VM console with administration credentials, it is necessary to issue `xl console` command:

```
root@xen-compute:/home/marco# xl console instance-000000f6

cthylla@cthylla-VirtualBox:~$
```

Figure 40 - accessing PHR vm's console

From now on all the commands that are issued will be redirected directly on PHR VM (note `cthylla@cthylla-VirtualBox` prompt, that corresponds with the PHR VM console.

At this point we will create a little textual file with a specific test pattern. The idea is to check, with administrative privileges, that the clear disk is normally accessible and readable by any cloud administrator. Later in this chapter we will do the same exercise with the cyphered

PHR VM, we will show that this process will fail. In order to do this step we will issue the following command:

```
#echo VaLVaLVaLVaLVaLVaLVaLVaLVaLVaLValVaLVaL > ValidationTestFile.txt
```

to create a file (`ValidationTestFile.txt`) containing the specific "`VaLVaLVaL….ValVaLVaL`" pattern.

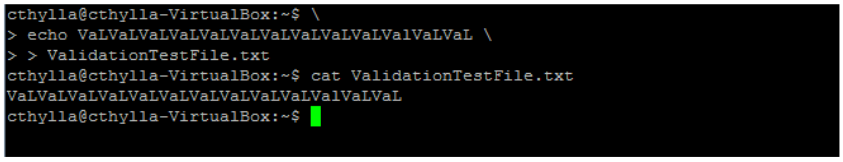


Figure 41 - creation of specific file to look for its pattern in the next steps

Now we will perform an HexDump on the live disk:

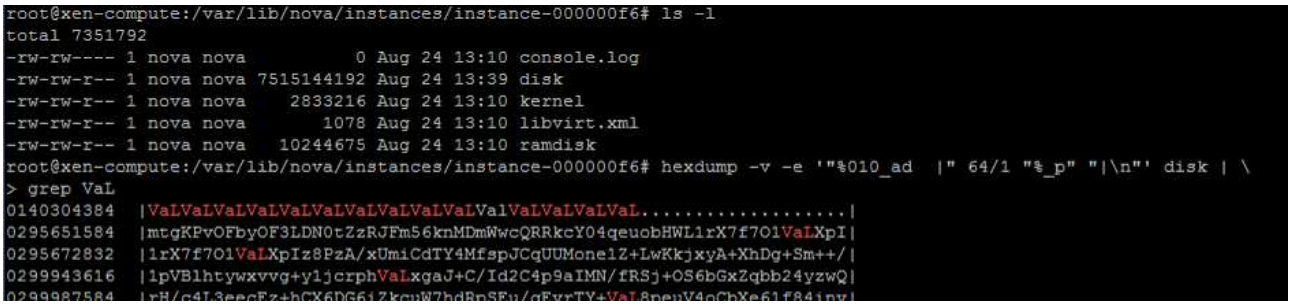


Figure 42 - hexdump and search for the specific pattern

As expected the file content that has been written into the disk can be read with no difficulties by the cloud admin.

3.1.1.4.1.4 Launching the encrypted VMs and checking its live disk

This task is executed similarly to the previous task, as the encryption is transparent for the user. However, while HexDumping its hard disk we should not be able to extract anything useful.

Setup the instance:

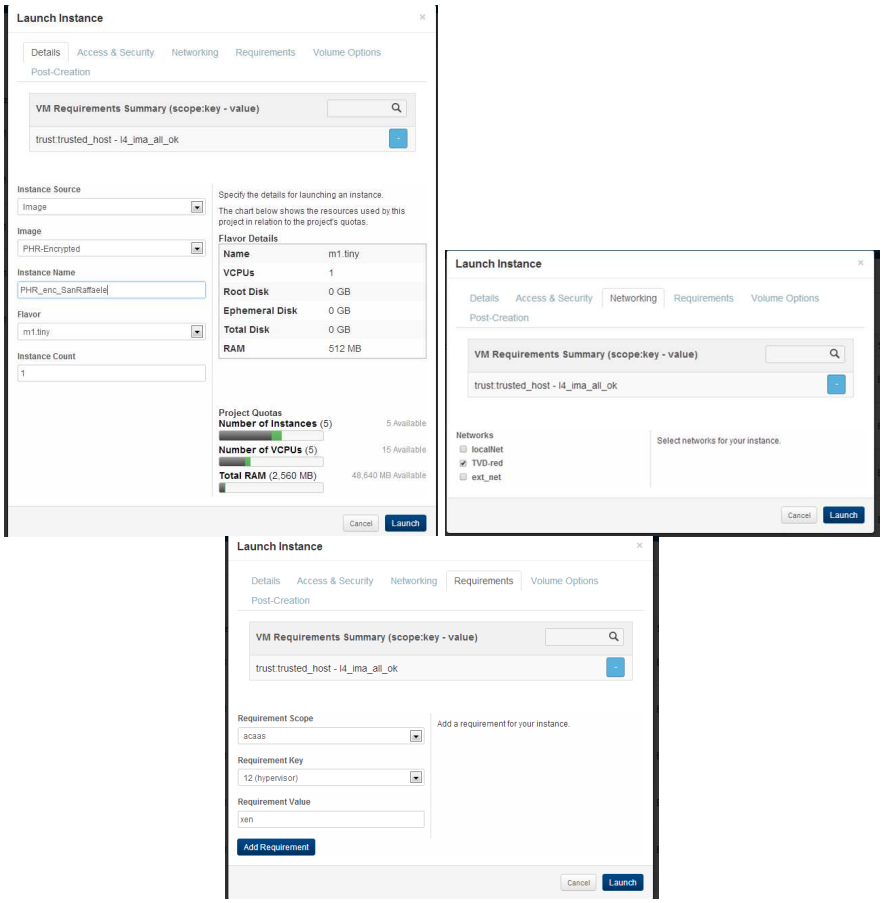


Figure 43 - deployment steps for encrypted PHR VM

Figure 44 - encrypted PHR vm correctly deployed and running

Check instances running via Admin console:

```
root@xen-compute:/var/lib/nova/instances# xl list
Name                                        ID   Mem VCPUs      State   Time(s)
Domain-0                                     0  1023     8     r-----   92715.1
Domain-T                                     1  1024     1     -b----       2.9
instance-000000f6-domc                      13    64     1     r-----  100817.9
instance-000000f7                           14   512     1     -b----      16.6
instance-000000f7-domc                      15    64     1     -b----       3.3
root@xen-compute:/var/lib/nova/instances#
```

Figure 45 - VMs running into the infrastructure

Access the PHR_encrypted console, write a text file with a specific text pattern and hexdump its file to look for that specific pattern:

```
root@xen-compute:/var/lib/nova/instances/instance-000000f7# xl console instance-000000f7

cthylla@cthylla-VirtualBox:~$ \
> echo VaLVaLVaLVaLVaLVaLVaLVaLVaLVaLVaL \
> > ValidationTestFile.txt
cthylla@cthylla-VirtualBox:~$ root@xen-compute:/var/lib/nova/instances/instance-000000f7#
root@xen-compute:/var/lib/nova/instances/instance-000000f7#
root@xen-compute:/var/lib/nova/instances/instance-000000f7# ls -l
total 7351792
-rw-rw---- 1 nova nova          0 Aug 25 16:58 console.log
-rw-rw-r-- 1 nova nova 7515144192 Aug 25 17:39 disk
-rw-rw-r-- 1 nova nova    2833216 Aug 25 16:58 kernel
-rw-rw-r-- 1 nova nova       1083 Aug 25 16:58 libvirt.xml
-rw-rw-r-- 1 nova nova   10244675 Aug 25 16:58 ramdisk
root@xen-compute:/var/lib/nova/instances/instance-000000f7# hexdump -v -e '"%010_ad  |" 64/1 "%_p" "|\n"' disk | grep VaLVaL
root@xen-compute:/var/lib/nova/instances/instance-000000f7#
```

Figure 46 - Hexdumping the encrypted disk

As expected the encrypted file does not reveal any useful information and cloud administrator cannot access its data.

3.1.1.4.1.5 Checking –domc disk I/O capabilities

As briefly described in the previous chapters, `-domc` instances are started-up simultaneously while starting any VM. `–domc` images act as proxy for all the IO memory needs and it takes care to encrypt/decrypt data from/to memory.

We performed a test that consist in writing 10 MB of data into the disk. We analyzed the `–domc` console (picture below, on the right) log, in order to see the actual bytes that has been written to the disk. As we can see in the picture below the proxy works properly.

```
root@xen-compute:/var/lib/nova/instances/instance-000000f7# xl console instance-000
000f7

cthylla@cthylla-VirtualBox:~$ dd if=/dev/zero of=test bs=1M count=10
10+0 record dentro
10+0 record fuori
10485760 byte (10 MB) copiati, 0,00895226 s, 1,2 GB/s
cthylla@cthylla-VirtualBox:~$ ls -l
totale 10276
drwxr-xr-x 2 cthylla cthylla     4096 gen  7  2013 Documenti
-rw-rw-r-- 1 cthylla cthylla        0 ago 25 17:26 echo
drwxr-xr-x 2 cthylla cthylla     4096 gen  7  2013 Immagini
drwxr-xr-x 2 cthylla cthylla     4096 gen  7  2013 Modelli
drwxr-xr-x 2 cthylla cthylla     4096 gen  7  2013 Musica
drwxr-xr-x 2 cthylla cthylla     4096 gen  7  2013 Pubblici
drwxr-xr-x 2 cthylla cthylla     4096 gen  7  2013 Scaricati
drwxr-xr-x 2 cthylla cthylla     4096 gen  7  2013 Scrivania
-rw-rw-r-- 1 cthylla cthylla 10485760 ago 25 18:18 test
-rw-rw-r-- 1 cthylla cthylla       40 ago 25 17:39 ValidationTestFile.txt
drwxr-xr-x 2 cthylla cthylla     4096 gen  7  2013 Video
cthylla@cthylla-VirtualBox:~$ root@xen-compute:/var/lib/nova/instances/instance-000
000f7#
root@xen-compute:/var/lib/nova/instances/instance-000000f7#
```

```
~~~blkback stats (51713): 0 outstanding, 365.373845 MiB read, 79.81224 MiB written
~~~blkback stats (51713): 0 outstanding, 365.373901 MiB read, 79.81224 MiB written
~~~blkback stats (51713): 0 outstanding, 365.373901 MiB read, 79.81272 MiB written
~~~blkback stats (51713): 0 outstanding, 365.373901 MiB read, 79.81304 MiB written
~~~blkback stats (51713): 0 outstanding, 365.373901 MiB read, 79.81304 MiB written
~~~blkback stats (51713): 0 outstanding, 365.373901 MiB read, 79.81348 MiB written
~~~blkback stats (51713): 0 outstanding, 365.373901 MiB read, 79.81372 MiB written
~~~blkback stats (51713): 0 outstanding, 365.373901 MiB read, 79.81372 MiB written
~~~blkback stats (51713): 0 outstanding, 365.373901 MiB read, 79.81428 MiB written
~~~blkback stats (51713): 0 outstanding, 365.373901 MiB read, 79.81428 MiB written
~~~blkback stats (51713): 0 outstanding, 365.373901 MiB read, 79.81428 MiB written
~~~blkback stats (51713): 0 outstanding, 365.373901 MiB read, 79.81428 MiB written
~~~blkback stats (51713): 0 outstanding, 365.373901 MiB read, 79.81456 MiB written
~~~blkback stats (51713): 0 outstanding, 365.373905 MiB read, 79.81492 MiB written
~~~blkback stats (51713): 0 outstanding, 365.373905 MiB read, 89.91788 MiB written
~~~blkback stats (51713): 0 outstanding, 365.373905 MiB read, 89.91788 MiB written
~~~blkback stats (51713): 0 outstanding, 365.373905 MiB read, 89.91788 MiB written
~~~blkback stats (51713): 0 outstanding, 365.373905 MiB read, 89.91788 MiB written
~~~blkback stats (51713): 0 outstanding, 365.373905 MiB read, 89.91820 MiB written
root@xen-compute:/home/marco#
```

Figure 47 - showing -domc console and read/write operations

3.1.1.5 Conclusion

All the tests we performed during activity has shown clearly that Crypto as a Service works as expected and it is actually increasing the overall security of TClouds Infrastructure.

The technique used to maintain the keys hidden to the Administrator (via the use of the special DomainT instance) guarantees that tampering of VM data is extremely difficult to achieve, even for cloud administrator.

During the execution of the validation activity we noticed a little drawback that will be fixed in next releases: the special instance proxy `–domc` is not properly destroyed when the user destroys its encrypted counterpart image. This can be solved by directly destroy the `–domc` instance manually by issuing `xl destroy` command.

Performance tests has not been performed, since out of scope of this project, however we can imagine that live encryption an decryption of the data has an obvious and inevitable cost in terms of performances (mainly CPU resource), in addition we have to consider the amount of memory that –domc instance uses (around 66MB). In terms of business dimension this is

translated in an increase of prices that can be justified for the high added value that such component provides.

Thanks to Crypto as a Service features built in into TClouds Infrastructure and thanks to its completely transparent functionalities to the final user, Healthcare Appliance can benefit of higher security and privacy capabilities.

Requirements' assessment

LREQ1 – Confidentiality of personal data – Crypto as a Service build the foundation of transparent encryption of data for VMs that are not aware of cryptography. CaaS transparently encrypts storage used by Healthcare VMs so that data that is processed in the cloud management layer (DomainT) is always encrypted and hence provides confidentiality since no plain text data leaves the cyphered VM.

LREQ2 (Availability and Integrity of personal data) & AHSECREQ1 (Confidentiality of stored and transmitted data) & AHSECREQ2 (Integrity of stored and transmitted data) – Likely as in LREQ1, integrity of personal data is satisfied: data is protected so that tampering becomes evident and integrity can be verified. Availability, however, cannot be satisfied since the cloud owner is in control of encrypted VM (he can start and stop the VMs)

In conclusion we can assess that the Validation activity of Crypto as a Service component, part of TClouds Infrastructure, has SUCCESSFULLY PASSED.

3.1.2 ACaaS, Ontology TVD, Remote Attestation Validation Activities

ACaaS, Ontology TVD and Remote Attestation validation activities has been performed simultaneously since the three component works tightly coupled together. Their features (VMs separation and user requirement satisfaction) and their nature (component of TClouds OpenStack Infrastructure) to be completely transparent at application level makes executing all together their validation activities a natural process.

Activity ID	**Remote_1**
Activity type	Proof of Concept
Activity description	The activity is performed by employing the features offered by Access Control as a Service (ACaaS) in addition to the capabilities of the Remote Attestation subsystem under evaluation. The OpenAttestation server and all Trustworthy OpenStack services are running. The database used by RA Verifier (a component of the Remote Attestation Service) has already been populated with digests from Ubuntu packages fetched from a remote repository. In all SRX nodes (tclouds-stack with KVM hypervisor, xen-compute with XEN hypervisor), executables and libraries being used come from installed packages and all software is up to date. Deployment Activities (from the Trustworthy OpenStack Dashboard): 1. Define the location requirement a. Go to the Admin tab, Security Properties panel b. Click on "Add Requirement" button on the top right corner c. Fill in the form with the following values: "location" (Specifications), "Location of the Compute node" (Description), "it,de" (Options) 2. Define the security properties of the KVM and the XEN nodes a. From the previous panel, click on "Add Security Property"

	button on the bottom right corner b. Fill in the form with the following values: “tclouds-stack” (Host Name), “location” (Property Name), “de” (Property Value) c. Repeat the procedure but fill in the form with the following values: “xen-compute” (Host Name), “location” (Property Name), “it” (Property Value) 3. Create new flavor m1.verytiny a. Go to the Admin tab, Flavors panel b. Click on “Create Flavor” button on the top right corner 4. Fill form fields with the following values: “m1.verytiny” (Name), “1” (VCPUs), “256” (RAM MB), “0” (Root Disk GB), “0” Ephemeral Disk GB) 5. Add an integrity requirement to the Flavor “m1.verytiny” a. Go the Admin tab, Flavors panel b. Select the “Edit Extra Spec” option for the Flavor “m1.verytiny” c. Fill in the form with the following values: “trust” (Scope), “trusted_host” (Key), “l4_ima_all_ok” (Value) d. Click on “Add Extra Spec” button to submit the form e. Verify that in the index page of Flavors the column “Extra Specs” of the Flavor “m1.verytiny” contains the text “trust: trusted_host” 6. Add an integrity requirement to the existing Flavor “m1.tiny” a. Repeat all steps of point 5 but select the Flavor “m1.tiny” 7. Perform the network configuration activities of the Ontology_1 validation activity Validation activities (from the Trustworthy OpenStack Dashboard): 1. Launch the “ERH_IT” virtual machine of the healthcare scenario with the specified integrity requirement (no location requirement) a. Go the Project tab, Instances panel b. Click on “Launch Instance” button on the top right corner c. Fill in form fields and ensure that the Flavor “m1.verytiny” is selected d. Select the network “TVD-healthcare” in the Networking tab e. Go to the Requirements tab, set the following values: “acaas” (Requirement Scope), “2 (location)” (Requirement Key), “it” (Requirement Value), click on “Add Requirement” button f. Click on “Launch” button to start a new virtual machine 2. Simulate an attack on the KVM host by downgrading a software package a. Log into the KVM host through ssh b. Execute the command “apt-get install ntpdate=1:4.2.6.p3+dfsg-1ubuntu3” to downgrade ntpdate to a previous version c. Reboot the KVM node d. Execute: router add –net 192.168.249.0/24 gw 130.192.1.86 e. Execute: router add –net 192.168.250.0/24 gw 130.192.1.87 f. Execute: ntpdate 3. Launch the “Appliance” virtual machine with the same integrity requirement specified before and location set to “de” a. Perform the steps a-d listed for point 1 but specify “m1.tiny” as flavor b. Go to the Requirements tab, set the following values: “acaas” (Requirement Scope), “<req id> (location)” (Requirement Key), “de” (Requirement Value), click on “Add Requirement”

	button c. Click on “Launch” button to start a new virtual machine 4. Launch the previous virtual machine with the same integrity requirement specified before and location set to “it” a. Perform the same steps listed for point 3 but set “it” as Requirement Value 5. Launch the “PHR” and “ERH_DE” virtual machines with a lower integrity requirement specified before and location set to “de” a. Perform the steps a-e listed for point 1 but set “de” as Requirement Value b. In the Requirements tab, set the following new values: “trust” (Requirement Scope), “trusted_host” (Requirement Key), “l3_ima_pkg_not_security_updates” (Requirement Value), click on “Add Requirement” button c. Click on “Launch” button to start the new virtual machines Cleanup Activities (from the Trustworthy OpenStack Dashboard): 1. Remove the integrity requirement from the Flavor “m1.verytiny” a. Go to the Admin tab, Flavors panel b. Select the “Edit Extra Spec” option for the Flavor “m1.verytiny” c. Click on “Delete Extra Spec” button in the first row of the table
Acceptance Criteria	The Activity is passed if: • Point 1c: row with content: “trust:trusted_host - l4_ima_all_ok” and row with content: “acaas:<req id> - it” in the VM Requirements Summary • Point 1: In the index page of Instances (Admin tab), the virtual machine created is running on xen-compute host (the XEN node) • Point 3b: row with content: “trust:trusted_host - l4_ima_all_ok” and row with content: “acaas:<req id> - de” in the VM Requirements Summary • Point 3: the virtual machine is not instantiated • Point 4a: row with content: “trust:trusted_host - l4_ima_all_ok” and row with content: “acaas: <req id> - it” in the VM Requirements Summary • Point 4: the virtual machine is instantiated in the xen-compute host (the XEN node) • Point 5a: row with content: “trust:trusted_host - l4_ima_all_ok” and row with content: “acaas: <req id> - de” in the VM Requirements Summary • Point 5b: first row replaced with content: “trust:trusted_host – l3_ima_pkg_not_security_updates” in the VM Requirements Summary • Point 5: the virtual machines are instantiated in the tclouds-stack host (the KVM node)
References Documents:	(TClouds factsheet – Remote attestation, 2013) (Deliverable D2.1.2, 2012) (Deliverable D2.4.2, 2012) (Intel Open Attestation SDK)
Requirements satisfied	LREQ1, AHSECREQ6, AHSECREQ7

Table 9 - Remote Attestation Validation Activity

Activity ID	**Ontology_1**
Activity type	Proof of Concept
Activity description	The Quantum service is running on the KVM node. The Ontology-based Reasoner subsystems (Libvirt daemon + Libvirt Quantum Agent) are running on both the KVM and XEN nodes. Network configuration activities (with Trustworthy OpenStack Dashboard): • Create TVD-healthcare network o Go to Project tab, Networks panel o Click on “Create Network” button o Specify “TVD-healthcare” as network name (Network tab) o Specify “192.168.249/24” as network address (Subnet tab) o Specify 192.168.249.254 as gateway IP (Subnet tab) o Click on “Create” button • Create the attacker’s network o Perform the same steps as for the TVD-healthcare network but specify “Attacker-net” and “192.168.250.0/24” respectively as network name and address • Perform additional configuration steps o Log into the KVM node o Execute: source /root/keystonerc o Execute: quantum router-create router1 o Execute: quantum router-create router2 o Execute: quantum subnet-list o Execute: quantum subnet-update <ID subnet with cidr 192.168.249/24> --enable_dhcp=False o Execute: quantum router-interface-add <ID router1> <ID subnet with cidr 10.1.0.0/24> o Execute: quantum router-interface-add <ID router2> <ID subnet with cidr 10.2.0.0/24> o Execute: quantum net-list o Execute: quantum router-gateway-set <ID router1> <ID ext_net network> o Execute: quantum router-gateway-set <ID router2> <ID ext_net network> o Execute: router add –net 192.168.249.0/24 gw 130.192.1.86 o Execute: router add –net 192.168.250.0/24 gw 130.192.1.87 VM Deployment activities (with the Trustworthy OpenStack Dashboard): • Deploy the healthcare scenario VMs as described in the Deployment Activities section for Remote_1 • Deploy two attacker VMs (one for each SRX node), by using the location requirement provided by ACaaS (in addition, select the Attacker-net network in the VM launch form, Networking tab). For the purpose of validation, use the currently registered image “cirros” and flavor “m1.verytiny” Validation activities (with the Trustworthy OpenStack Dashboard): 1. Obtain the IP address assigned to each VM of the healthcare scenario in the figure at the beginning of this document 2. Check whether each VM of the healthcare scenario can contact all others (on the same node, on different nodes). a. Execute (in the KVM node): quantum router-list

	b. Log into a VM through ssh (from the KVM node) by executing: ip netns exec qrouter-<router1 id> ssh <username>@<VM IP> c. Execute: ping <IP of the target VM> 3. Obtain the IP address assigned to each attacker VM in the Admin tab, Instances panel of the Dashboard 4. Check whether each attacker VMs can contact the VMs of the healthcare scenario (on the same node of the target VM, on a different node with respect to the target VM) a. Log into a VM through ssh by executing: ip netns exec qrouter-<router2 id> ssh cirros@<VM IP> [password: cubswin:)] b. Execute: ping <IP of the target VM>
Acceptance Criteria	The Activity is passed if: • At point 2, each VM of the healthcare scenario can contact all other VMs in the same TVD network • At point 4, each attacker's VM cannot contact any VM of the healthcare scenario
References Documents:	(TClouds factsheet - Ontology-based Reasoner, 2013) (Deliverable D2.3.1, 2011) (Deliverable D2.4.2, 2012) (Integration into Trustworthy OpenStack: to appear in D2.3.4, 2013)
Requirements satisfied	LREQ1, AHSECREQ6

Table 10 - Ontology TVD Validation Activity

Activity ID	**ACaaS_1**
Activity type	Proof of Concept
Activity description	Deployment activities: - Deploy the Italian EHR VM with the following requirements o The VM should not run in the same physical machine of the other EHR database VM (EHR_DE) o The VM should run on physical machines located in Italy - Deploy the German EHR VM with the following requirements o The VM should not run in the same physical machine of the Italian EHR VM o The VM should run on physical machines located in Germany o The VM should not run on the same physical machine of the Appliance VM - Deploy the Appliance VM - Deploy the PHR VM with the following requirement o The VM should not run in the same physical machine of the Appliance VM Validation activities 1- Check whether the requirements has been respected by manually inspecting the VM deployment o Done via OpenStack dashboard
Acceptance Criteria	The Activity is passed if: • At Point 1 all the VMs are deployed correctly • At point 2 no VM could be migrated
References	(Deliverable D2.3.2)

Documents:	(Deliverable D2.4.2, 2012)
Requirements satisfied:	LREQ3

Table 11 - Access Control as a Service Validation Activity

3.1.2.1 Remote Attestation Features

The Remote Attestation Service is a cloud subsystem responsible to assess the integrity of nodes in the cloud infrastructure through techniques introduced by the Trusted Computing technology.

This service gives significant advantages in the cloud environment. First, it allows cloud users to deploy their virtual machines on a physical host that satisfies desired security requirements, represented by integrity levels.

Secondly, this service allows cloud administrators to monitor the status of the nodes in an efficient way and to take appropriate countermeasures once a compromised host has been detected. For instance, administrators can isolate the host such that it cannot attack other nodes of the infrastructure.

3.1.2.2 Ontology TVD features

Cloud computing is one of the most promising technologies in these days since it allows a user to access a potentially unlimited set of virtualized resources but, at the same time, raises new security issues that are not present in the case of an ad-hoc infra-structure. Indeed, Virtual Machines (VMs) of different tenants are often executed on the same hardware and this increases the possibilities that a virtual resource is accessed by unauthorized entities, for example due to a wrong configuration.

One solution to address these issues is to consider a group of VMs as an unique entity on which a security policy must be coherently enforced. A model that has been developed for this purpose is the Trusted Virtual Domain (TVD).

A TVD consists of a set of Execution Environments or EEs (e.g. Virtual Machines) and an abstract communication channel which allows EEs to securely communicate over the physical network. Through the TVD concept it is possible to enforce the following security properties: Isolation: TVD members can communicate only among

themselves; Confidentiality/Integrity: communications among TVD members cannot be intercepted or modified by unauthorized entities; Trust: an EE can join a TVD only if the host which is running on satisfies the integrity properties specified in the TVD security policy.

3.1.2.3 AcaaS Features

The central component that man-ages the allocation of virtual resources of a cloud infrastructure's physical resources is known as the cloud scheduler. Currently available schedulers do not consider users' security and privacy requirements, neither do they consider the properties of the entire cloud infrastructure. For example, a cloud scheduler should consider the application's performance requirements and users security and privacy requirements. ACaaS is a novel cloud scheduler which considers both user requirements and infrastructure properties. It focuses on assuring users that their virtual resources are hosted using physical resources that match their proper-ties without getting users involved with understanding the details of the complex cloud infrastructure.

3.1.2.4 Validation Scenario

The three activities are tightly coupled together and they their execution will be done simultaneously.

First of all we will start by setting up all the TPaaS Healthcare VMs with all the necessary user requirements (locations of VMs, TVD affinity, integrity level…) than we will execute the activities and we will present their results.

Following is described the validation scenario adopted for these activities:

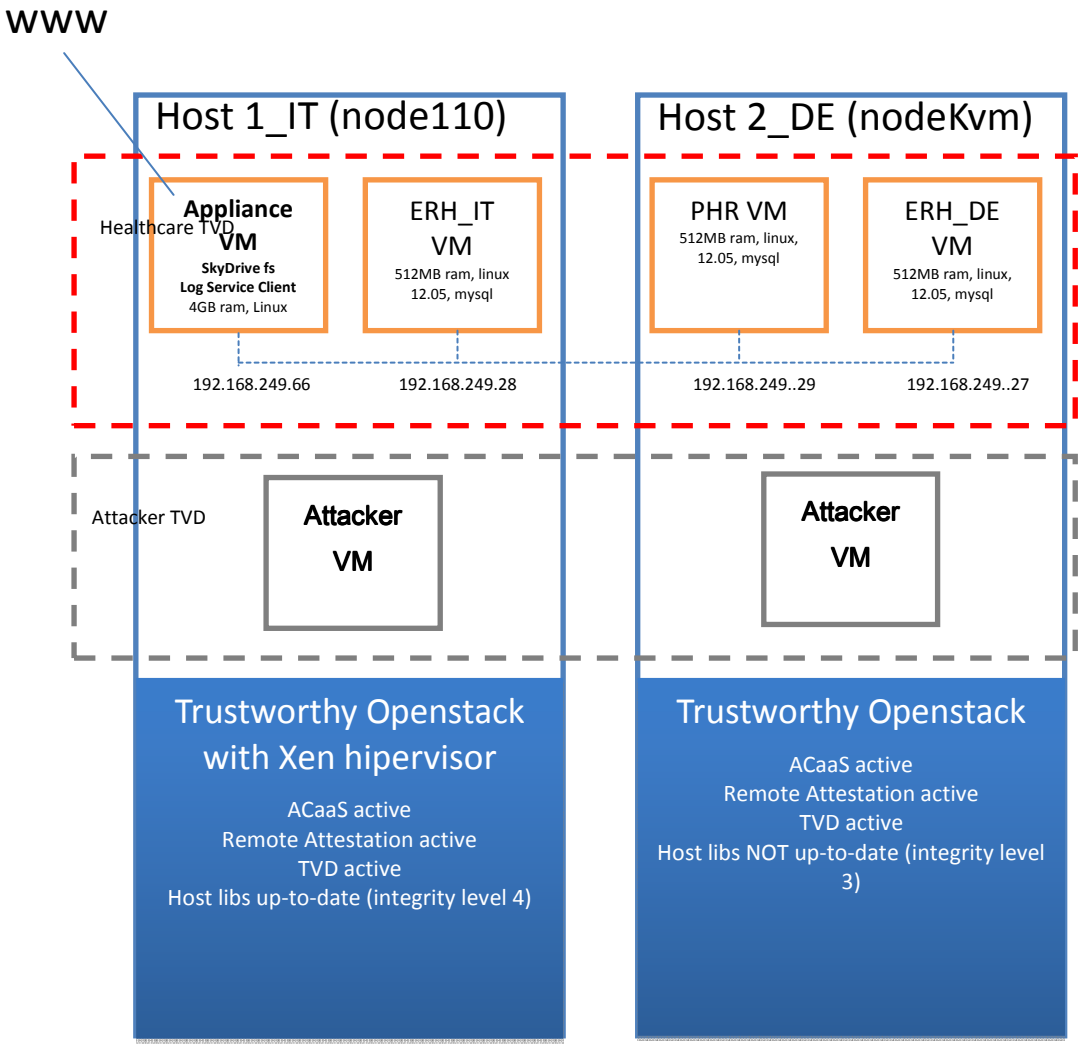


Figure 48 - Deployment scenario

For the validation scenario we have used the four Healthcare Platform Virtual Machines.

We have setup two hosts, one representing the Italian TClouds node and the other representing the German TClouds node. At the end of the validation activity we should reach the VM deployment configuration as above, in which the Appliance VM and the Italian EHR VM reside on the Italian node and where the PHR VM and the German VM reside on the German node. An Healthcare TVD will be created to confine all the virtual machines and deny any other VM but healthcare's one to access.

In this scenario the VMs will be deployed with specific user requirements:

Appliance VM:

- Integrity Level >= 4

PHR VM:

- Integrity Level >= 3

EHR_IT VM:

- Integrity Level >= 4
- Location = Italy

EHR_DE VM:

- Integrity Level >= 3
- Location = Germany

3.1.2.5 Validation Setup

Due to the nature of AcaaS, Ontology TVD and Remote Attestation (being part of the cloud infrastructure) all the features validated in these chapters are totally transparent to the VMs and they are unaware of the special environment surrounding them, thus we don't have any particular setup except for simply copying the Healthcare VMs into the TClouds Infrastructure. The Deployment itself is part of the validation process and will be shown during the execution.

However, the cloud owner must setup the environment in order to host the Healthcare VMs, namely the TVD networks, and all the virtual network infrastructure.

We started creating the TVD network:

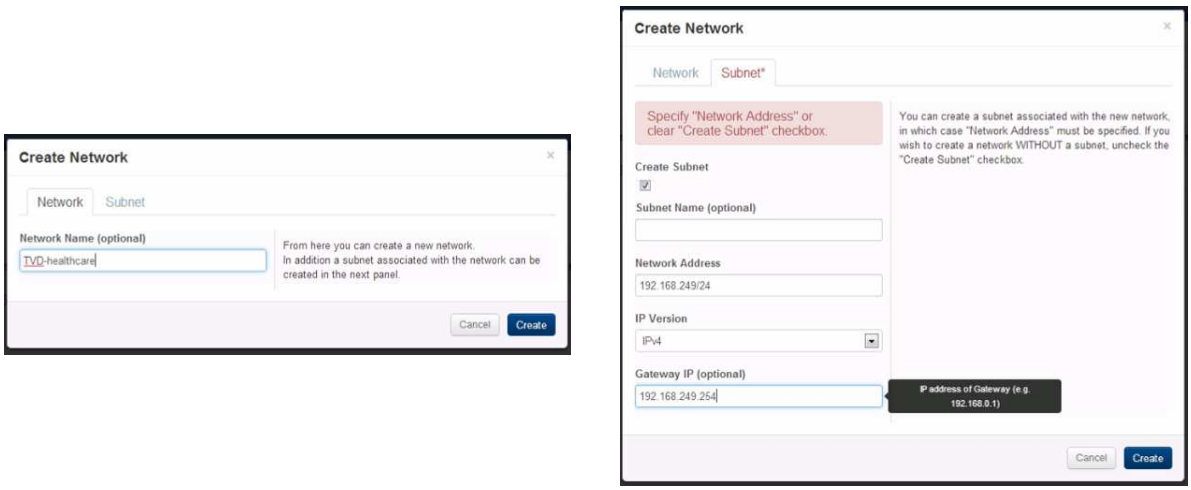


Figure 49 - TClouds Trustworthy OpenStack healthcare TVD creation

And then we created all the virtual networking devices (routers, subnets…).

```
root@node-kvm:/home/marco# quantum router-create router1
Created a new router:
+-----------------------+--------------------------------------+
| Field                 | Value                                |
+-----------------------+--------------------------------------+
| admin_state_up        | True                                 |
| external_gateway_info |                                      |
| id                    | 79cf58e7-e198-4927-ad95-e4de35689a9b |
| name                  | router1                              |
| status                | ACTIVE                               |
| tenant_id             | 2541dbf902ce4ca092d624793bc839df     |
+-----------------------+--------------------------------------+
```

Figure 50 - Creation of a new router for Healthcare network

```
root@node-kvm:/home/marco# quantum subnet-list
+--------------------------------------+------+-----------------+-----------------------------------------------------+
| id                                   | name | cidr            | allocation_pools                                    |
+--------------------------------------+------+-----------------+-----------------------------------------------------+
| 6fc07a9e-6656-40aa-9002-ce7245db9ee0 |      | 130.192.1.64/26 | {"start": "130.192.1.86", "end": "130.192.1.90"}    |
| e71c52c5-0ce7-4fef-a957-70ae69e3f7e2 |      | 192.168.249.0/24 | {"start": "192.168.249.1", "end": "192.168.249.253"} |
+--------------------------------------+------+-----------------+-----------------------------------------------------+
```

Figure 51 - List of network present in the host

As we can see there are two networks right now into TClouds. One belongs to the healthcare VMs, the other is the network that acts as the external network, attached with the rest of Internet.

```
root@node-kvm:/home/marco# quantum subnet-update e71c52c5-0ce7-4fef-a957-70ae69e3f7e2 --enable_dhcp=False
Updated subnet: e71c52c5-0ce7-4fef-a957-70ae69e3f7e2
```

Figure 52 - Updating Healthcare network to not to have DHCP server available (Healthcare VMs uses static IPs)

```
root@node-kvm:/home/marco# quantum router-list
+--------------------------------------+---------+-----------------------+
| id                                   | name    | external_gateway_info |
+--------------------------------------+---------+-----------------------+
| 79cf58e7-e198-4927-ad95-e4de35689a9b | router1 | null                  |
+--------------------------------------+---------+-----------------------+
```

Figure 53 - List of available routers into TClouds. Only the router that will be used for the Healthcare network is present

```
root@node-kvm:/home/marco# quantum router-interface-add 79cf58e7-e198-4927-ad95-e4de35689a9b e71c52c5-0ce7-4fef-a957-70ae69e3f7e2
Added interface to router 79cf58e7-e198-4927-ad95-e4de35689a9b
```

Figure 54 - Creation of a new virtual interface into the virtual router

```
root@node-kvm:/home/marco# quantum net-list
+--------------------------------------+----------------+--------------------------------------+
| id                                   | name           | subnets                              |
+--------------------------------------+----------------+--------------------------------------+
| 08e5e9d4-ee83-4b9e-b0c2-d4ed6592c57d | TVD-healthcare | e71c52c5-0ce7-4fef-a957-70ae69e3f7e2 |
| 22c35e19-c05c-43d6-aa97-e98520cad60f | ext net        | 6fc07a9e-6656-40aa-9002-ce7245db9ee0 |
+--------------------------------------+----------------+--------------------------------------+
```

Figure 55 - List of the available TVDs

```
root@node-kvm:/home/marco# quantum router-gateway-set 79cf58e7-e198-4927-ad95-e4de35689a9b 22c35e19-c05c-43d6-aa97-e98520cad60f
Set gateway for router 79cf58e7-e198-4927-ad95-e4de35689a9b
```

Figure 56 - Creation of a virtual gateway for the healthcare router.

```
root@node-kvm:/home/marco# route add -net 192.168.249.0/24 gw 130.192.1.86
```

Figure 57 - Creation of a new route to allow the Appliance Healthcare VM to access internet

Below is the schema of the environment we have just created:

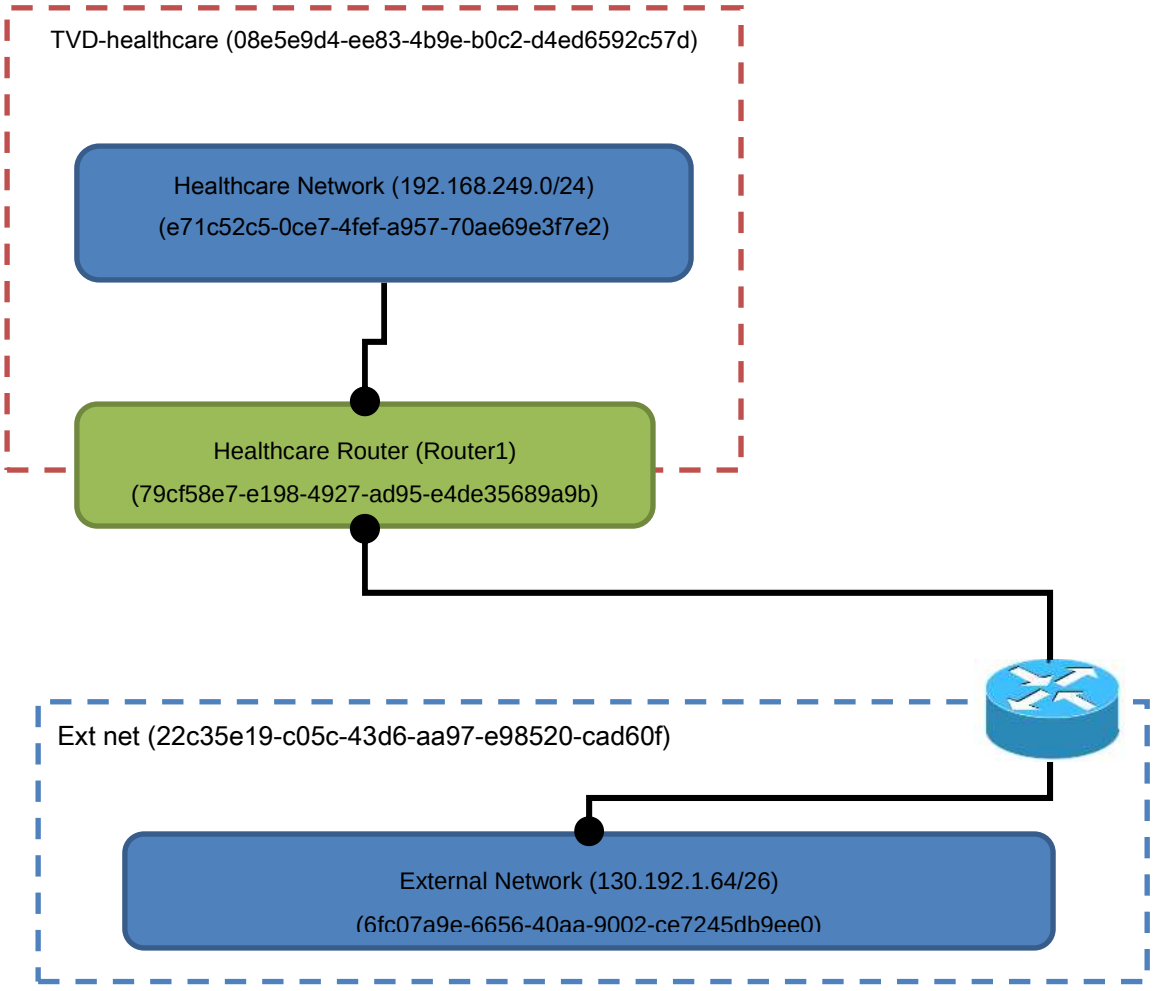


Figure 58 - Logical view of network and TVDs within TClouds infrastructure

Now the environment is ready to welcome all the Healthcare VMs

3.1.2.6 Validation Execution

The most of the Validation Execution can be done directly via TClouds Trustworthy OpenStack dashboard. All the deployment configuration are done from the dashboard, meanwhile to inspect the physical deployment and to make other tests we accessed as administrator directly to the host console.

3.1.2.6.1.1 Deployment

The first thing we did is to actually deploy all the healthcare VMs. Below the deployment of the EHR_IT VM.

It has been requested to deploy it into the Italian host (since it will holds medical data produced by Italian healthcare institutions) and that the host needs to have the highest integrity level (`l4_ima_all_ok`)

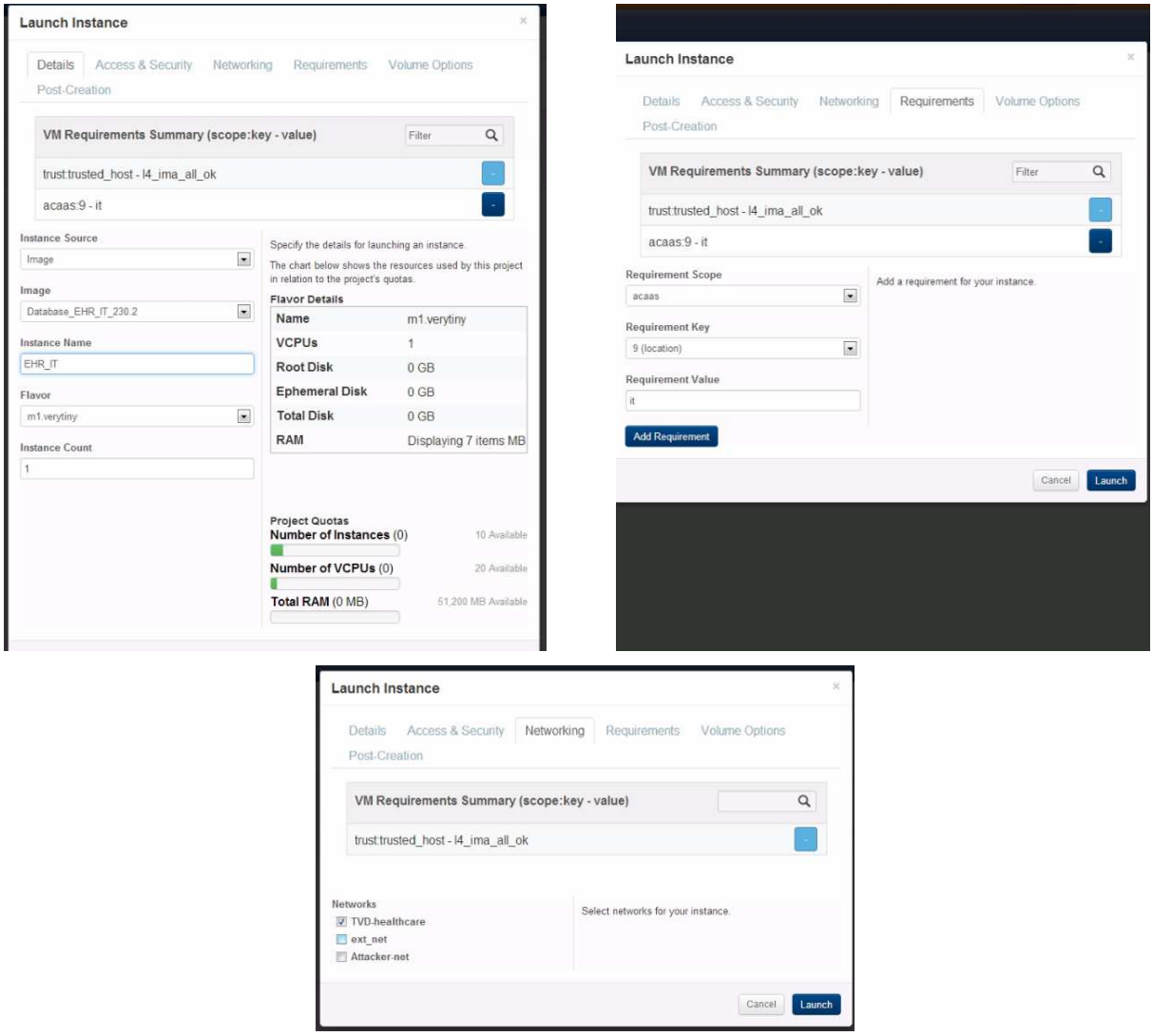


Figure 59 - Deployment of EHR_IT VM. Please notice: l4 integrity level, TVD-healthcare and IT location

Than we continued deploying the Appliance VM. This VM does not need any location requirement, it can be deployed into any host. Since the German host has a lower integrity level (l3) if we are going to require an higher integrity level (l4) the VM will be deployed into the Italian host since it is the only one that meets the requirements. We want to try anyway to force the deployment into the German host with an l4 integrity lever. We expect this attempt to fail.

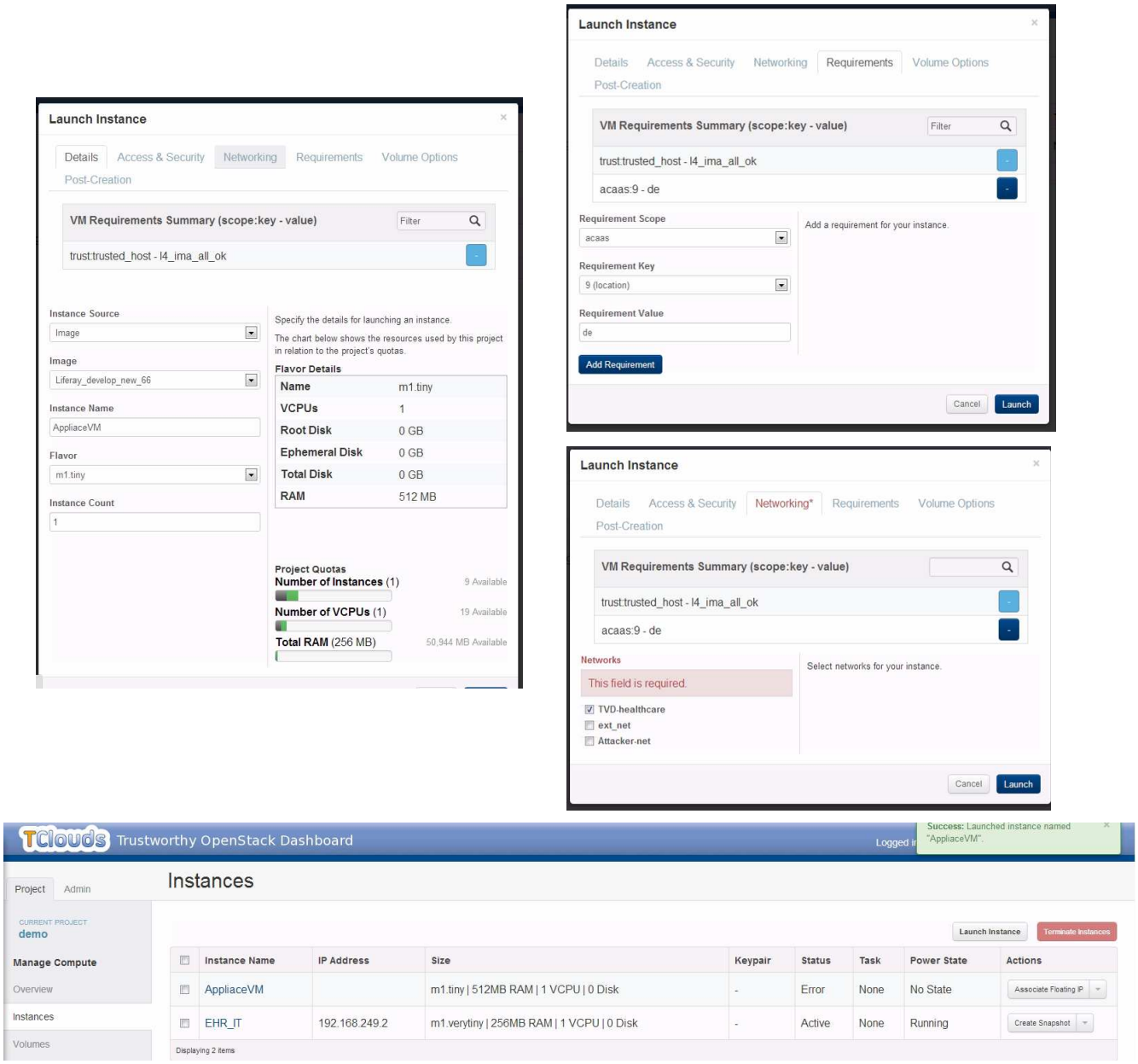


Figure 60 - Deployment of Appliance VM. Please notice: l4 integrity level, TVD-healthcare and DE location. The lower picture shows the deployment failure

As expected the Appliance VM has not been deployed since there aren't host available with the given user requirements.

We re-deployed the Appliance VM with the Italian location requirement and the integrity level to be at least l4.

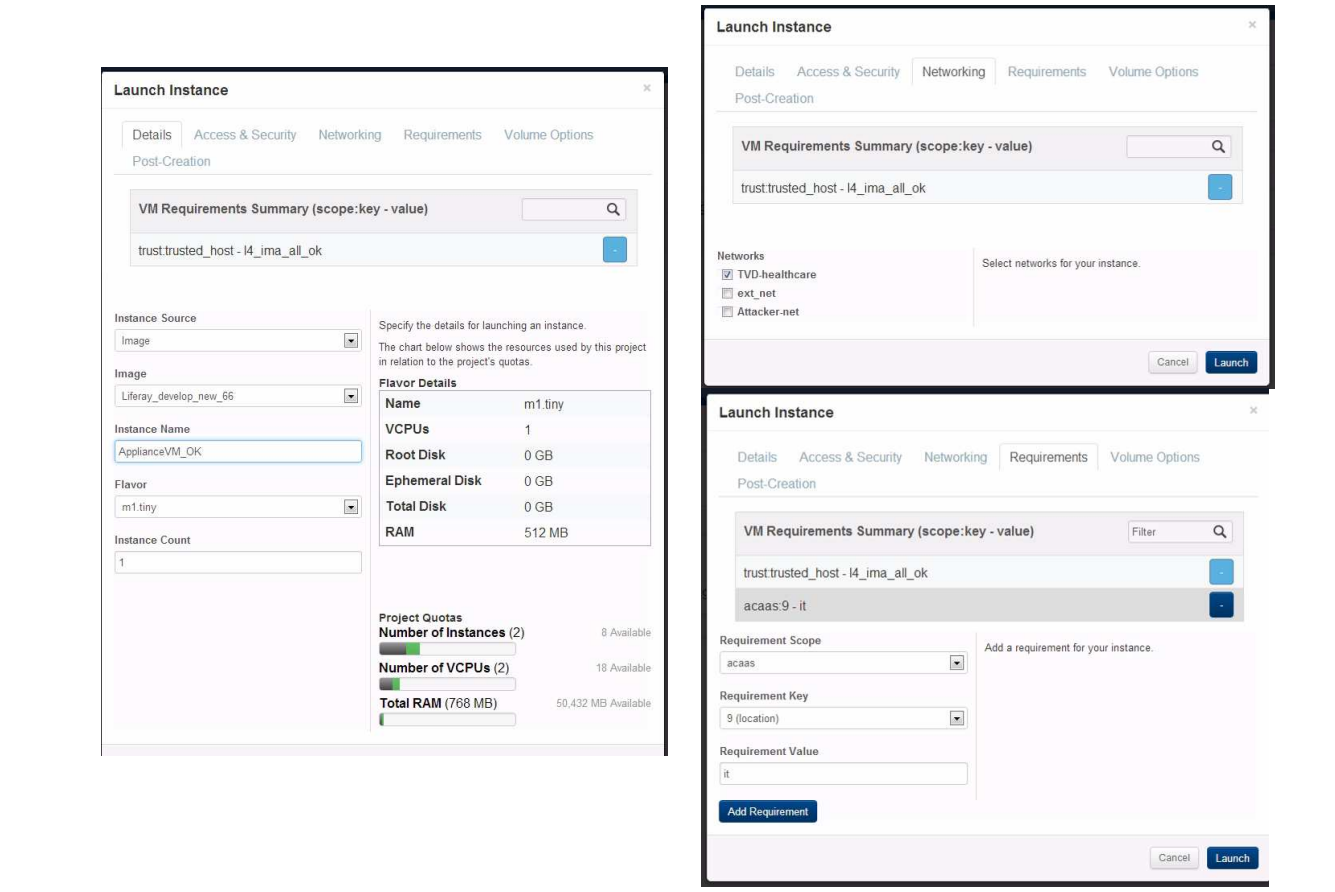


Figure 61 - Deployment of Appliance VM. Please notice: l4 integrity level, TVD-healthcare and IT location.

The next VM is the PHR VM. Also for this VM we decided to place requirements that cannot be met by any host, making the deployment to fail.

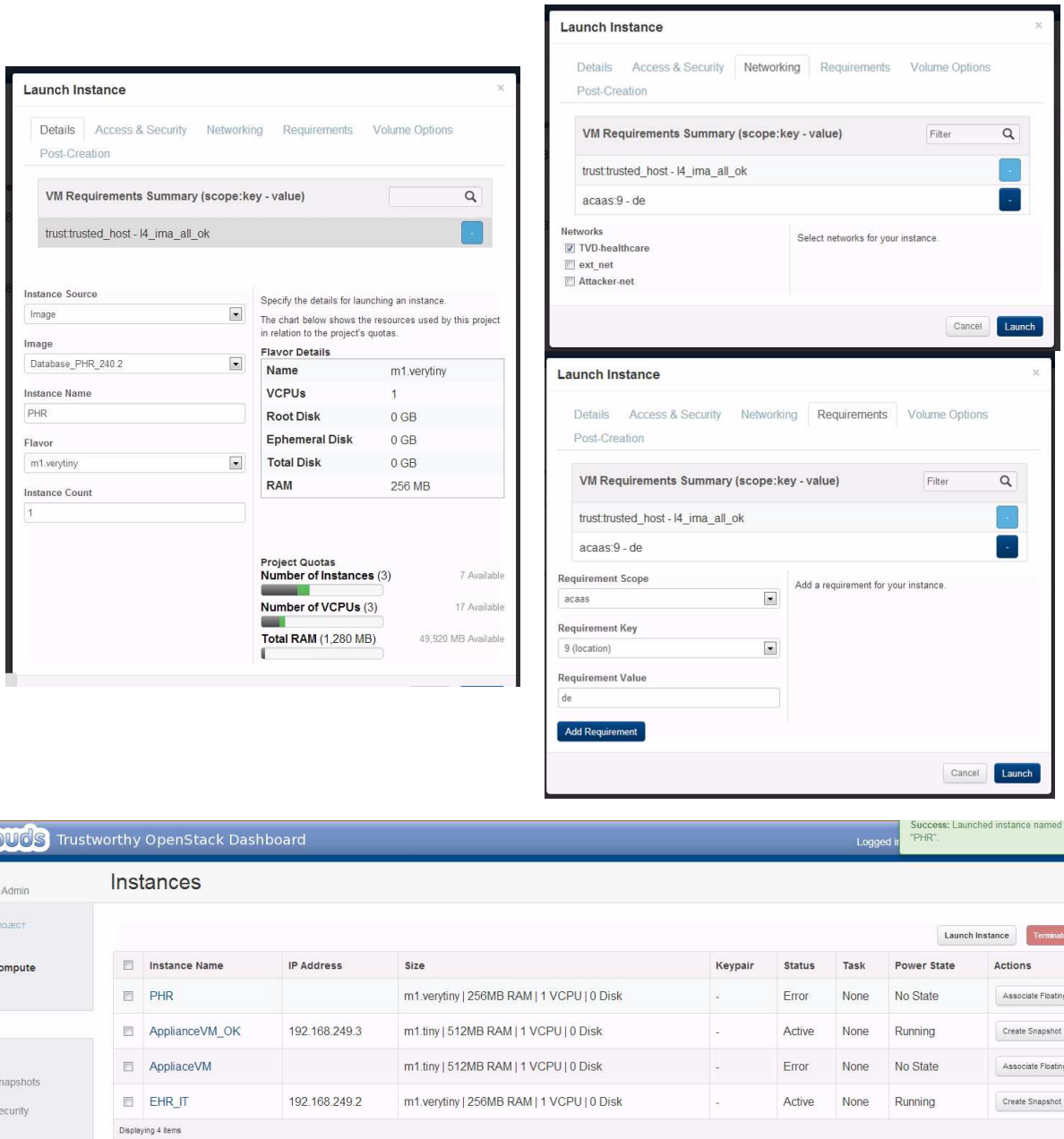


Figure 62 - Deployment of PHR VM. Please notice: l4 integrity level, TVD-healthcare and DE location. The lower picture shows the deployment failure

PHR VM doesn't have any location requirement nor doesn't need the highest integrity level since PHR data has not legal implication and is just data provided by the end user itself.

We re-deployed the VM, but at this time we asked for a lower integrity level.

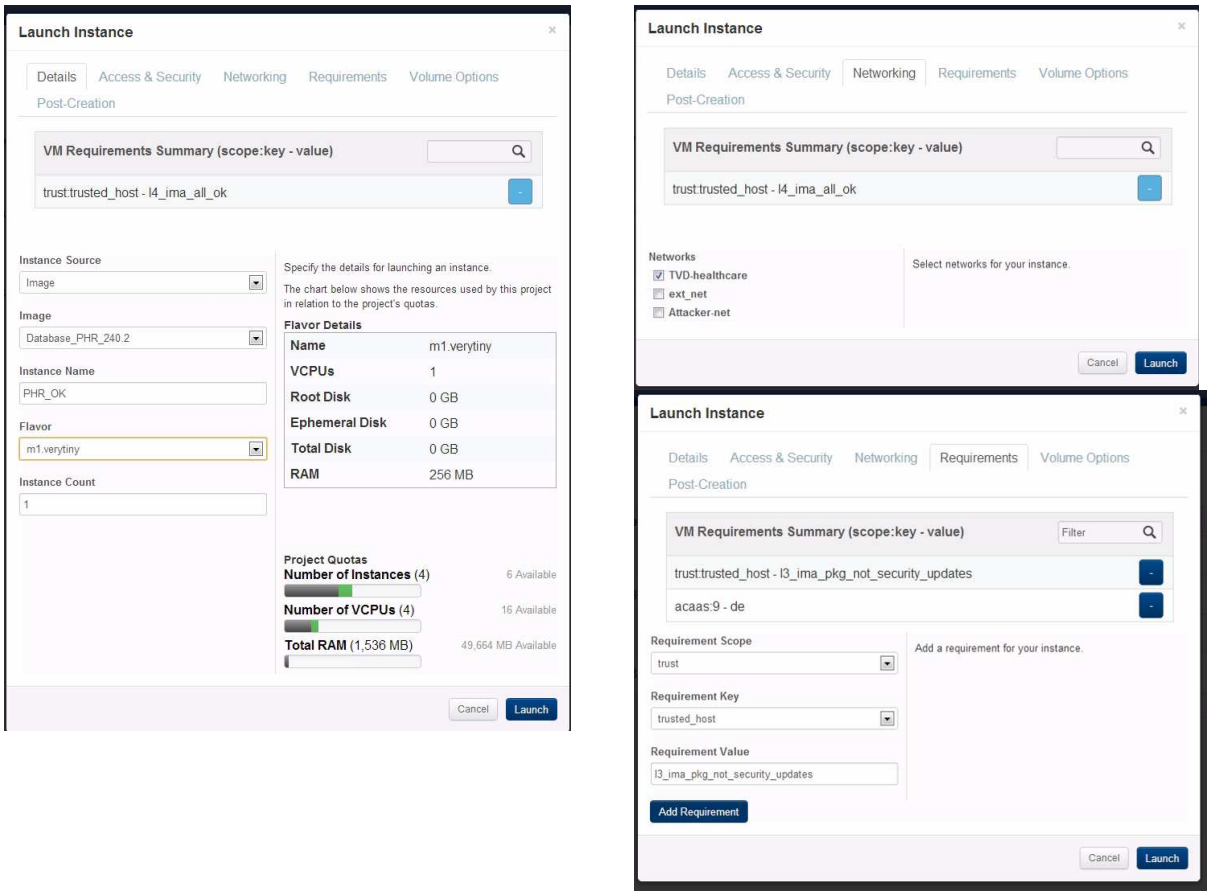


Figure 63 - Deployment of PHR VM. Please notice: l3 integrity level (done as last step. Default value is l4), TVD-healthcare and DE location.

Now it is the time of the German EHR VM. For the sake of the validation purposes, we decided to place the German VM into the German node (in order to be legally compliant, due to its data nature) and we placed a lover integrity level (l3) in order to allow AcaaS component to be able to deploy the VM in a host.

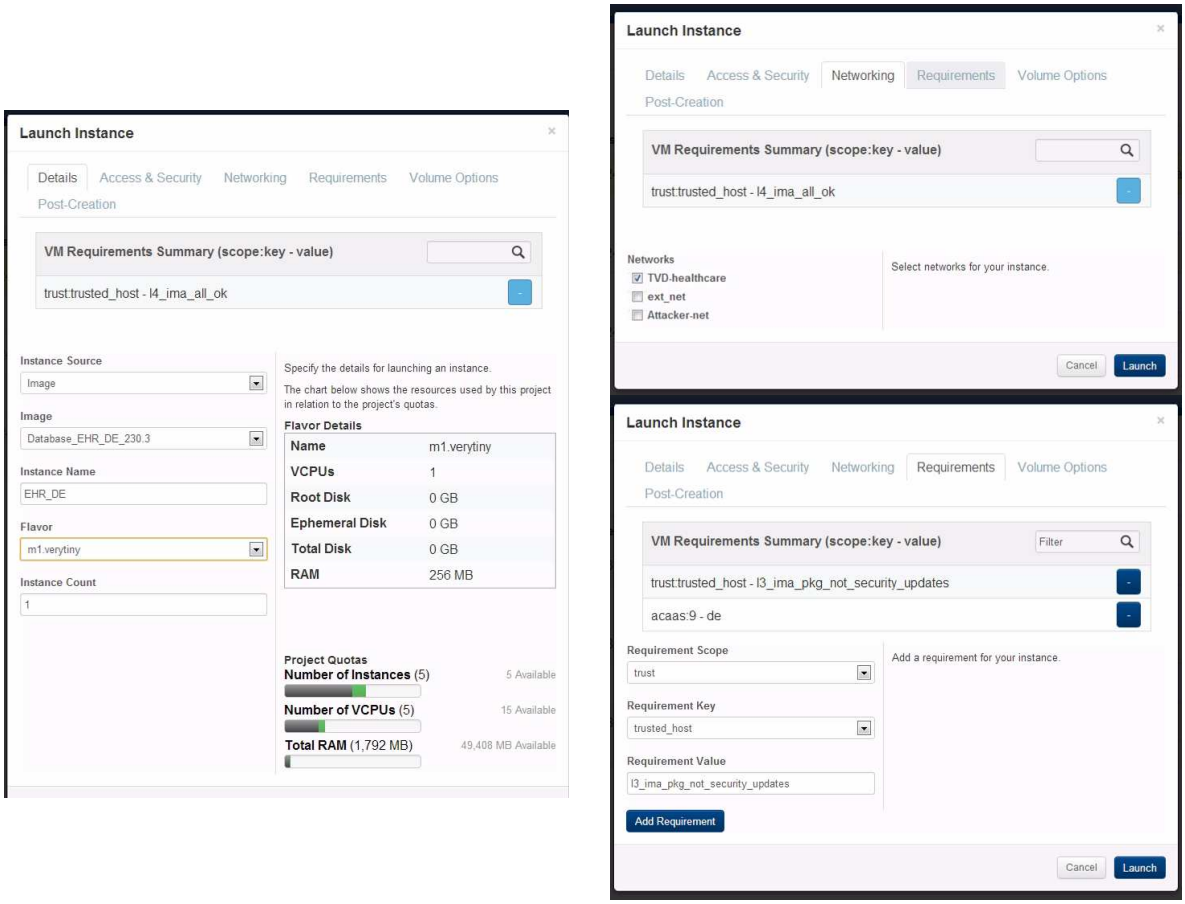


Figure 64 - Deployment of EHR_DE VM. Please notice: l3 integrity level (done as last step. Default value is l4), TVD-healthcare and DE location.

As we can see in the picture below, now he have been able to deploy all the Healthcare Virtual Machines.

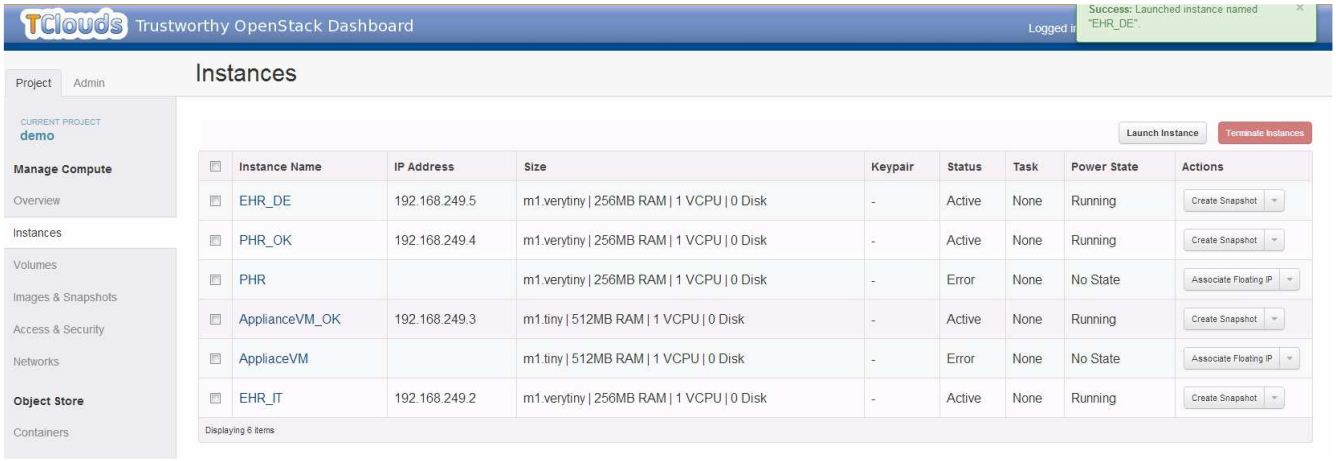


Figure 65 - Overview of deployed VMs into TClouds infrastructure

3.1.2.6.1.2 Check VMs location

To be sure that the VMs have been deployed in the correct host, we accessed into the administrator console and we run hypervisor commands to check the actual VM deployment location.

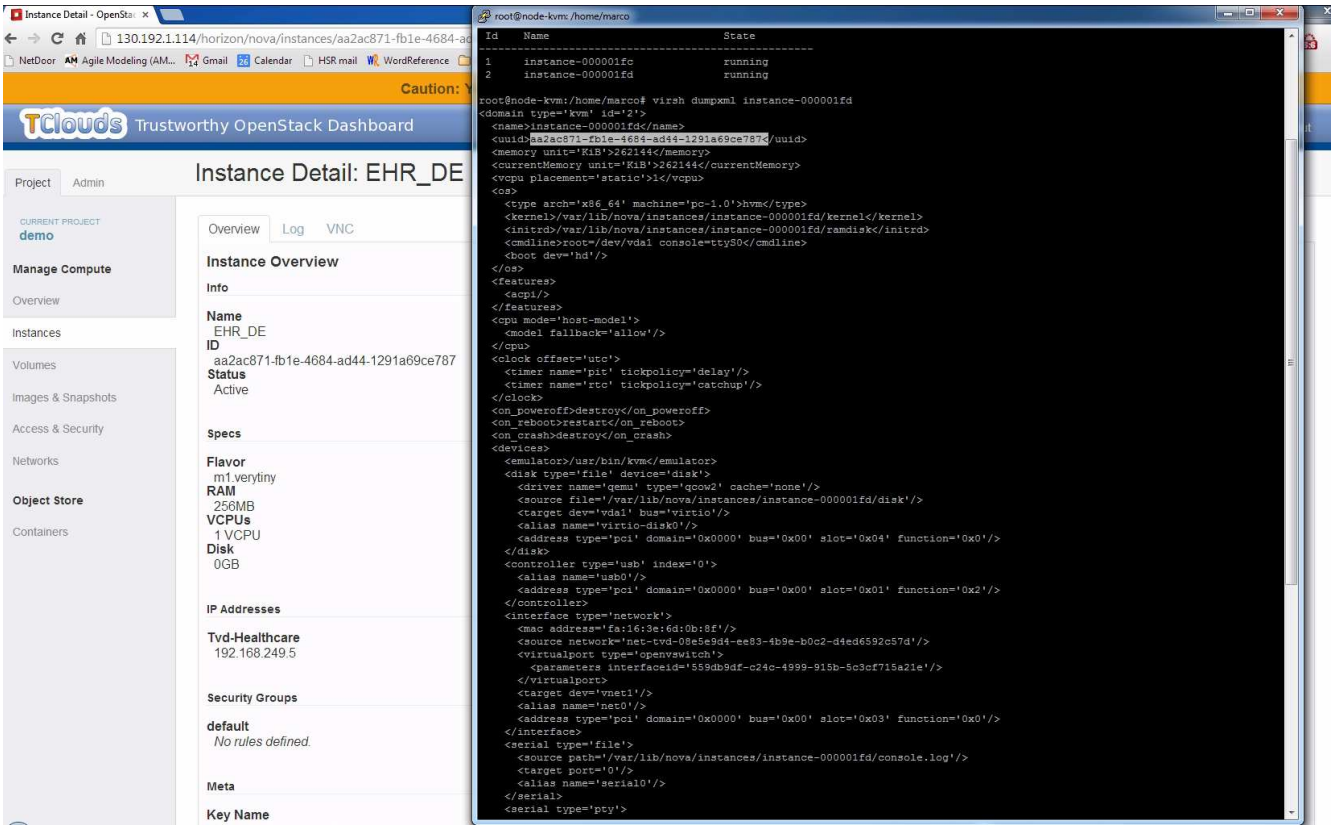


Figure 66 - Check location of EHR_DE VM. It results into Node-kvm (German Node). As expected.

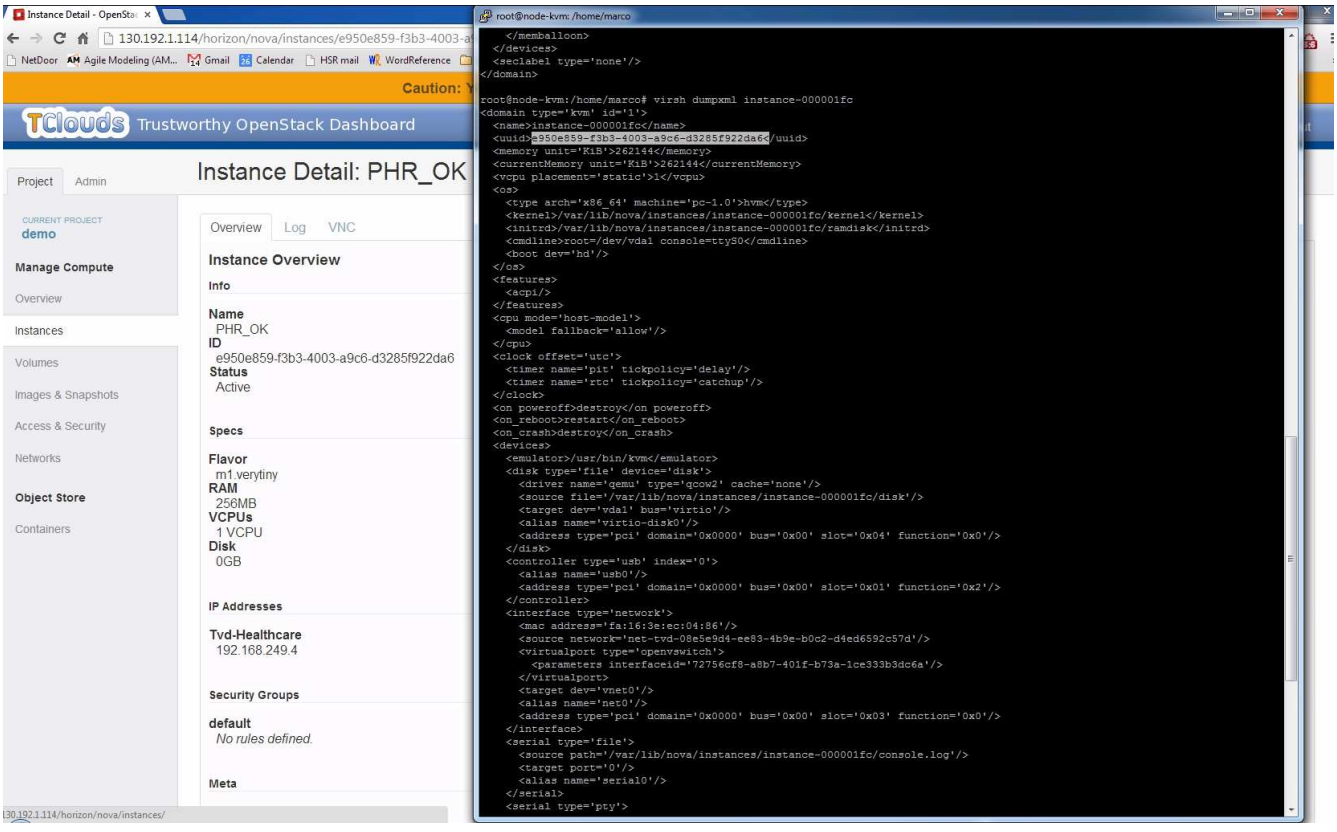


Figure 67 - Check location of PHR VM. It results into Node-kvm (German Node). As expected.

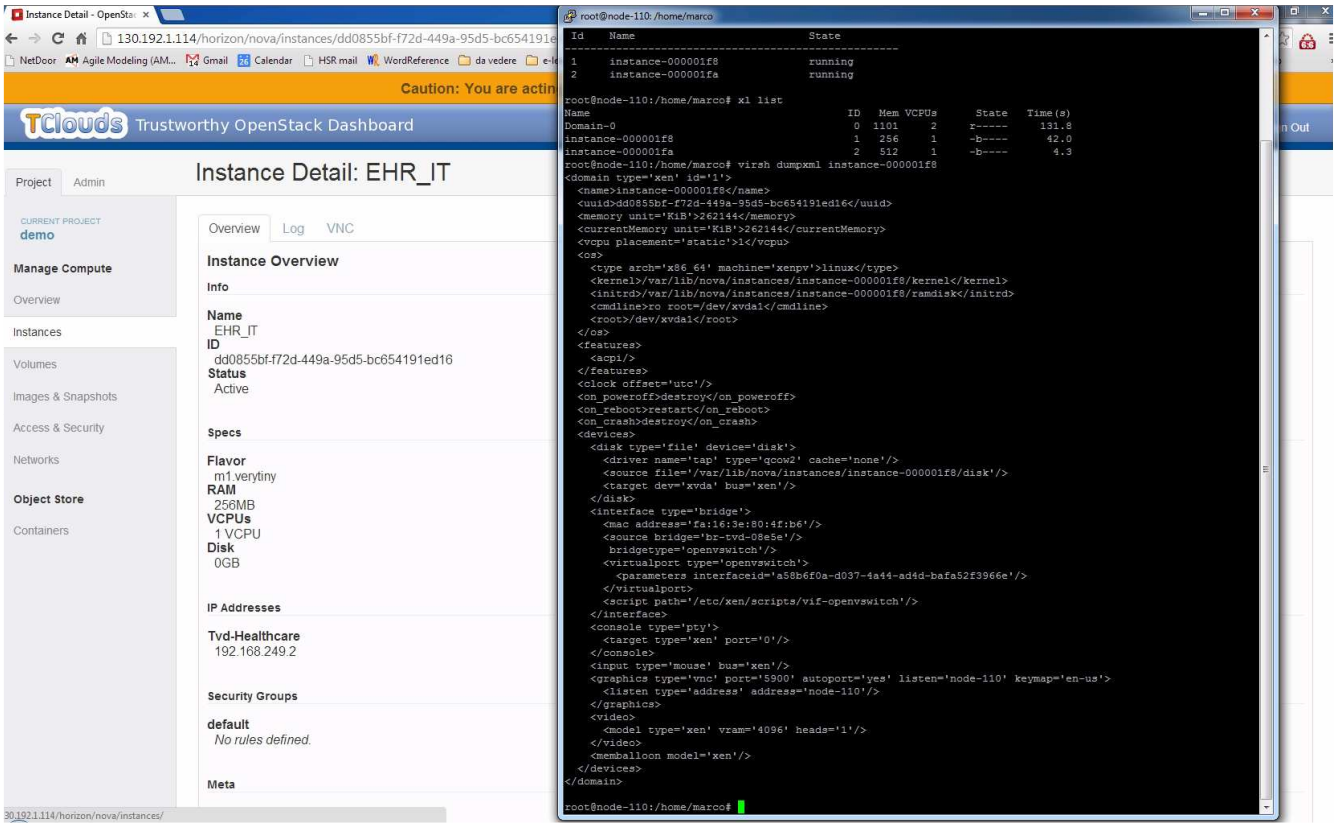


Figure 68 - Check location of EHR_IT VM. It results into Node-110(Italian Node). As expected.

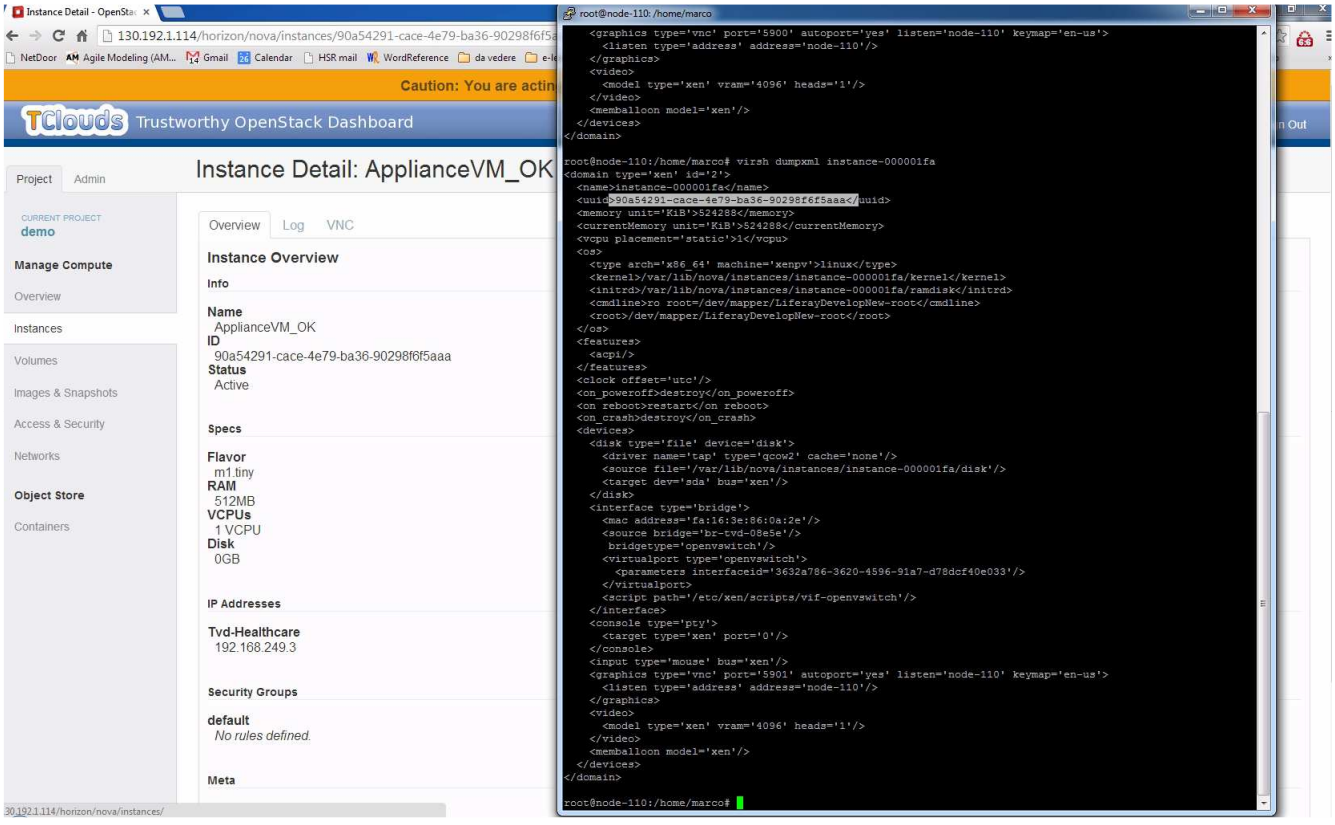


Figure 69 - Check location of Appliance VM. It results into Node-110 (Italian Node). As expected.

The previous images assess the effective location of all the four healthcare VMs. They reside all in the right locations.

3.1.2.6.1.3 Create Attacker VMs and network.

At this stage we have the four Healthcare VMs running inside the Healthcare TVD. Now we are going to deploy two malicious VMs (we called it Attacker VMs) into another TVD. We will show that the healthcare TVDs will not allow the Attacker VMs to access and vice-versa, creating a virtual barrier among the two networks.

The first things we are going to do is to access as administrator and create the virtual devices.

```
root@node-kvm:/home/marco# quantum router-create router2
Created a new router:
+-------------------------+--------------------------------------+
| Field                   | Value                                |
+-------------------------+--------------------------------------+
| admin_state_up          | True                                 |
| external_gateway_info   |                                      |
| id                      | 0678948d-e098-4bb5-8f49-d59a21218590 |
| name                    | router2                              |
| status                  | ACTIVE                               |
| tenant_id               | 2541dbf902ce4ca092d624793bc839df     |
+-------------------------+--------------------------------------+
```

Figure 70 - creation of new router for the Attacker network

```
root@node-kvm:/home/marco# quantum subnet-list
+--------------------------------------+------+-----------------+-----------------------------------------------------+
| id                                   | name | cidr            | allocation_pools                                    |
+--------------------------------------+------+-----------------+-----------------------------------------------------+
| 6fc07a9e-6656-40aa-9002-ce7245db9ee0 |      | 130.192.1.64/26 | {"start": "130.192.1.86", "end": "130.192.1.90"}    |
| b35fdc91-4d51-44b0-bce2-35590bba3f7e |      | 192.168.250.0/24 | {"start": "192.168.250.2", "end": "192.168.250.254"} |
| e71c52c5-0ce7-4fef-a957-70ae69e3f7e2 |      | 192.168.249.0/24 | {"start": "192.168.249.1", "end": "192.168.249.253"} |
+--------------------------------------+------+-----------------+-----------------------------------------------------+
```

Figure 71 - List of subnets available

```
root@node-kvm:/home/marco# quantum router-list
+--------------------------------------+---------+-----------------------+
| id                                   | name    | external_gateway_info |
+--------------------------------------+---------+-----------------------+
| 0678948d-e098-4bb5-8f49-d59a21218590 | router2 | null                  |
| 79cf58e7-e198-4927-ad95-e4de35689a9b | router1 | null                  |
+--------------------------------------+---------+-----------------------+
```

Figure 72 - list of available routers into TClouds

```
root@node-kvm:/home/marco# quantum router-interface-add 0678948d-e098-4bb5-8f49-d59a21218590 b35fdc91-4d51-44b0-bce2-35590bba3f7e
Added interface to router 0678948d-e098-4bb5-8f49-d59a21218590
```

Figure 73 - Add a new virtual network interface to Router2

```
root@node-kvm:/home/marco# quantum net-list
+--------------------------------------+-----------------+--------------------------------------+
| id                                   | name            | subnets                              |
+--------------------------------------+-----------------+--------------------------------------+
| 08e5e9d4-ee83-4b9e-b0c2-d4ed6592c57d | TVD-healthcare  | e71c52c5-0ce7-4fef-a957-70ae69e3f7e2 |
| 22c35e19-c05c-43d6-aa97-e98520cad60f | ext_net         | 6fc07a9e-6656-40aa-9002-ce7245db9ee0 |
| 6d3b2e44-12cb-4361-9e57-880ebc445bfb | Attacker-net    | b35fdc91-4d51-44b0-bce2-35590bba3f7e |
+--------------------------------------+-----------------+--------------------------------------+
```

Figure 74 - List of network available

```
root@node-kvm:/home/marco# quantum router-gateway-set 0678948d-e098-4bb5-8f49-d59a21218590 22c35e19-c05c-43d6-aa97-e98520cad60f
Set gateway for router 0678948d-e098-4bb5-8f49-d59a21218590
```

Figure 75 - linking of Attacker network with the infrastructure gateway

```
root@node-kvm:/home/marco# route add -net 192.168.250.0/24 gw 130.192.1.87
```

Figure 76 - Adding a new route from the attacker network to the gateway

Than we add the two Attacker VMs:

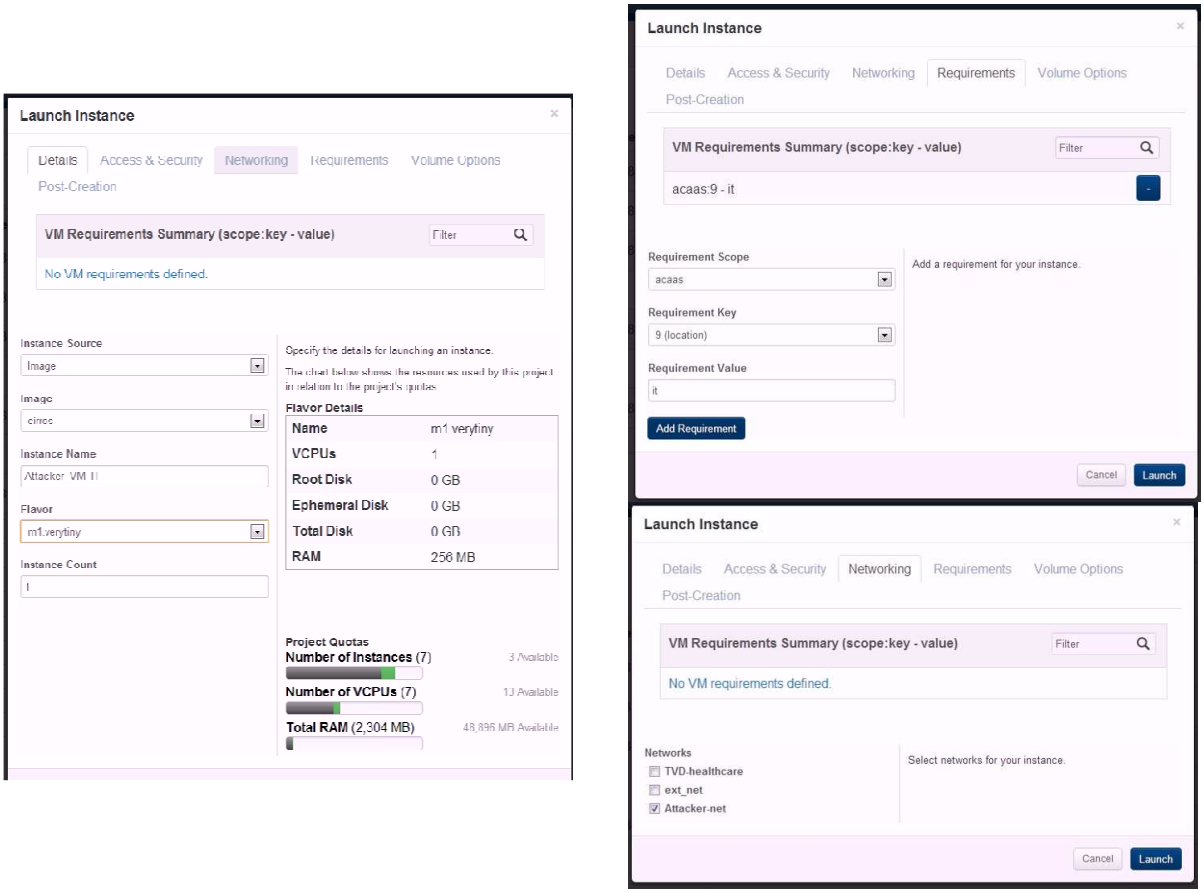


Figure 77 - Deployment of one of the two Attacker VM

Figure 78 - Overview of all the VMs deployed into the infrastructure.

The picture below describes the final virtual infrastructure available in order to have a grasp the new configuration:

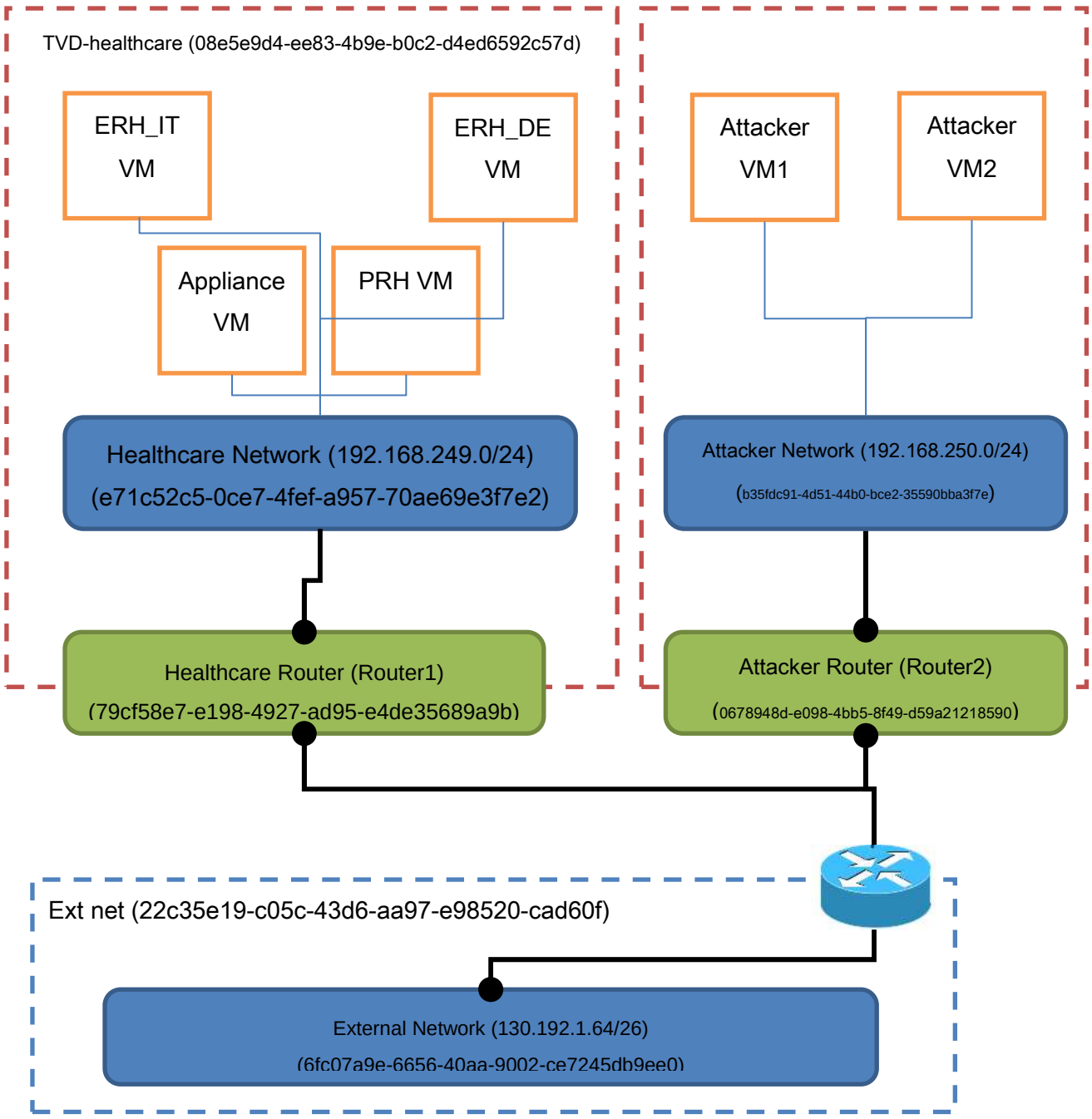


Figure 79 - Overview of the virtual networks configuration

At this stage we are ready to access the console of one of the Healthcare VM and we will try to ping either the two Attacker VM and the other healthcare VMs. We will repeat the test by entering into the attacker VM console as well.

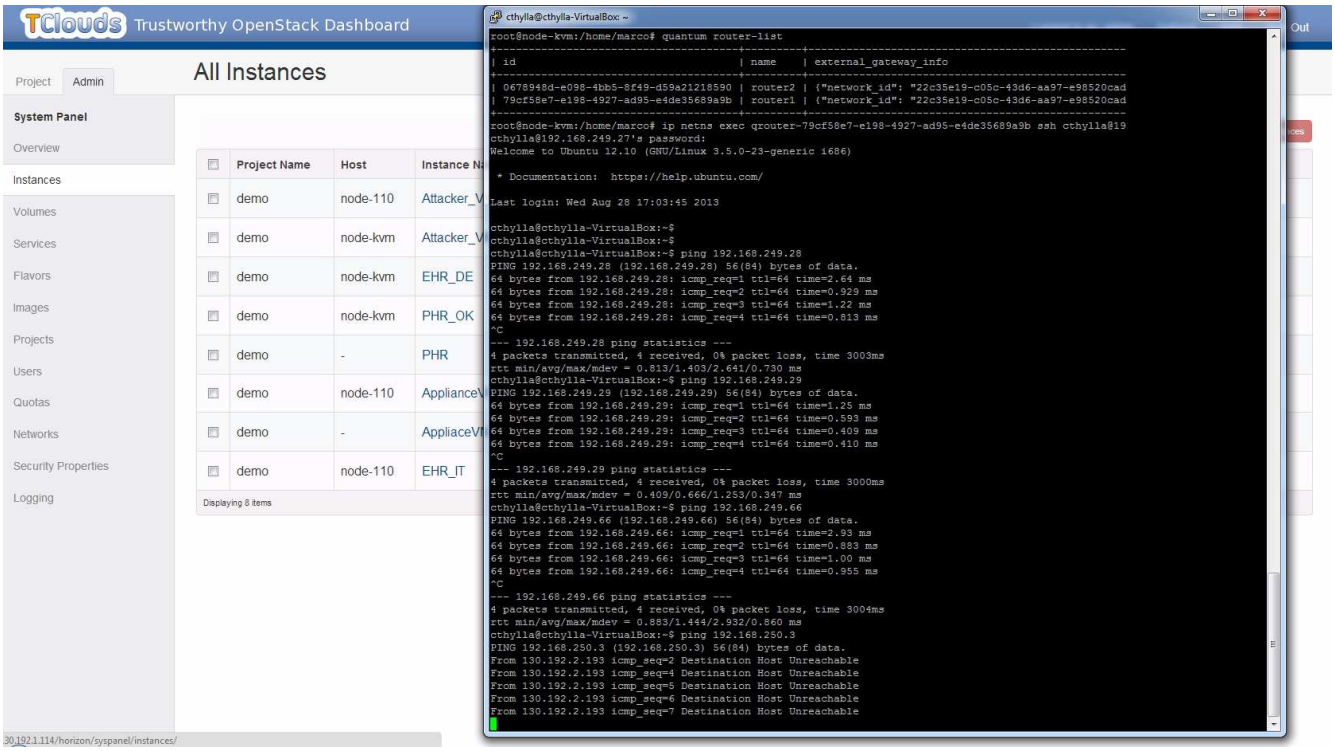


Figure 80 - Access to healthcare console e ping to all other VMs (please note the failure towards the attacker VMs)

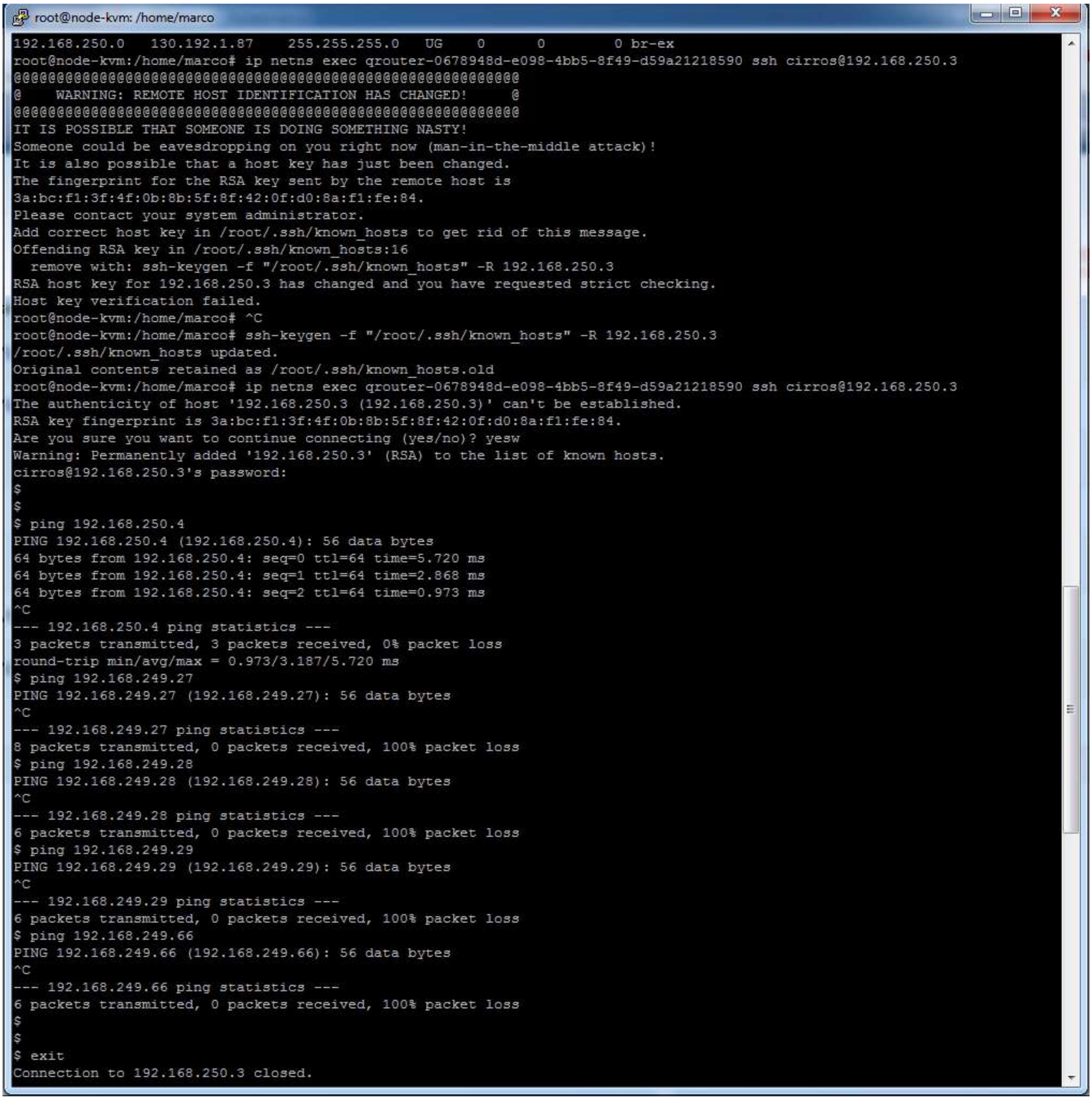


Figure 81 - access to the attacker console and ping to all the other VMs. Please note the inability to access to the healthcare VMs

As we can see it has not been possible to access the other VMs that are in the Attacker TVD and vice-versa.

3.1.2.7 Conclusion

For the Healthcare Scenario to control the location of the data and to have the assurance that a VM runs respecting the cloud customer requirements are very important feature that may facilitate cloud adoption.

The need to assure to Healthcare Customer that stored data resides in the same country has a huge impact on moving to the cloud. Many companies feel the risk of losing governance of their data and reject the idea of giving data to someone else.

The validation of Access Control as a Service, Remote Attestation and Ontology TVD has shown that they properly work. Allowing VM separation among different tenants and respect of user requirements.

Requirement's assessment

LREQ2 – Availability and integrity of personal data – & AHSECREQ5 – Availability of the application - & AHSECREQ1 – confidentiality of stored and transmitted data - Through the Trusted Virtual Domain concept, this subsystem helps satisfy these requirements by isolating the network connecting the virtual machines that run the user application. This avoids that an attacker intercepts the communication channel among TVD members or mounts a DOS attack.

LREQ3 - Control of location and responsible provider – AcaaS takes into account the properties of each cloud node available. These properties identifies the capabilities of the computing node. As seen in the validation execution, one of the properties is associate a physical location with the node, this location property can then be enforced ad VM deployment and the user request is respected by deploying on the correct node the VM.

LREQ5 – Transparency for the customer – Remote attestation guarantees transparency to the healthcare customer since it can give the proof of the integrity of the node

AHSECREQ6 – non repudiation – Remote Attestation subcomponent allows to prove that the application can be trusted and the integrity of the system is maintained.

AHSECREQ8 – Data source authentication – this requirement is satisfied because a virtual machine can communicate only with members of the same TVD

We can thus assess that Remote_1, AcaaS_1 and Ontology_1 Validation activities are SUCCESSFULLY PASSED.

3.1.3 Cheap BFT – Validation Activity

Here is described the validation of CheapBFT subsystem. CheapBFT's validation activities as described in D3.3.3 have been modified in order to accommodate the last changes in the platform development. In particular, it has been decided to maintain CheapBFT replicas only for the Log Service feature of TClouds, in order to proof its concept. Due to this change, the validation will be held accordingly to the demo scenario as described in D2.4.2.

Activity ID	**CheapBFT_1**
Activity type	Proof of concept
Activity description	CheapBFT is coupled with the Log Service subsystem. The system should be able to be resilient to one log tampering. 1- Deploy and setup the CheapBFT-LogService bundle 2- The log client writes continuously new log records and retrieves the stored log after every 100 write operations. 3- Compromise a log replica by deleting a record 4- Despite the induced error, the client should be able to retrieve the right data back since the error has been detected by CheapBFT, which has led to a protocol switch

	and thereby to the activation of the previously passive replica. As a consequence of the protocol switch, the resource usage should have been increased.
Acceptance Criteria	At step 2, no error should occur and at step 5, the client should still be able to get the right data back. At step 3, the passive replica should become active and start working to overcome the error occurred on the tampered replica
Requirements satisfied:	LREQ2, AHSECREQ2, AHSECREQ3, AHSECREQ4, AHSECREQ5, AHSECREQ8
References Documents:	(TClouds factsheet 09 Cheap BFT, 2013) (Deliverable D2.1.2, 2012) (Resource Efficient Byzantine Fault Tolerance, 2012) (Deliverable D2.4.2, 2012)

3.1.3.1 CheapBFT features

CheapBFT takes inspiration to the Byzantine fashion to overcome issues as fault tolerance and enhancing it with a wider spectrum of errors like software bugs, spurious hardware errors and intrusions.

Generally, Byzantine fault-tolerant algorithms require $3f+1$ replicas in order to tolerate f arbitrary faults. However, CheapBFT is able to rely on only $f+1$ active replicas in order to work properly during normal-case operation and switch to $2f+1$ active replica if an inconsistency among replica has been detected. Further, CheapBFT employs CASH, a novel FPGA-based trusted hardware module which is able to guarantee a small computing base as well as high performance rates.

The agreement protocol of CheapBFT consists of three sub-protocols: the normal-case protocol CheapTiny, the transition protocol CheapSwitch, and the fallback protocol MinBFT. During normal-case operation, CheapTiny makes use of passive replication to save resources; it is the first BFT agreement protocol that requires only $f+1$ active replicas backed by f passive ones. However, CheapTiny is only able to detect errors but not to tolerate them. Therefore, in case of suspected or detected faulty behavior of replicas, CheapBFT runs CheapSwitch to active the passive replicas and bring all non-faulty replicas into a consistent state. Having completed CheapSwitch, the replicas temporarily execute the MinBFT protocol, which involves $2f+1$ active replicas (i.e., it can tolerate up to f faults), before eventually switching back to CheapTiny.

3.1.3.2 Validation scenario

In this validation scenario we are going to assess the effective functionality of the CheapBFT system by using $f+1$ replicas when no inconsistent state is detected and $2f+1$ replicas when one of the active replicas is compromised.

For the purpose of validation, the TClouds Log Service feature has been selected to be used with the CheapBFT protocol. The number of faults to be recognized in this validation scenario is 1.

Following, the logical architecture used for the validation scenario is shown.

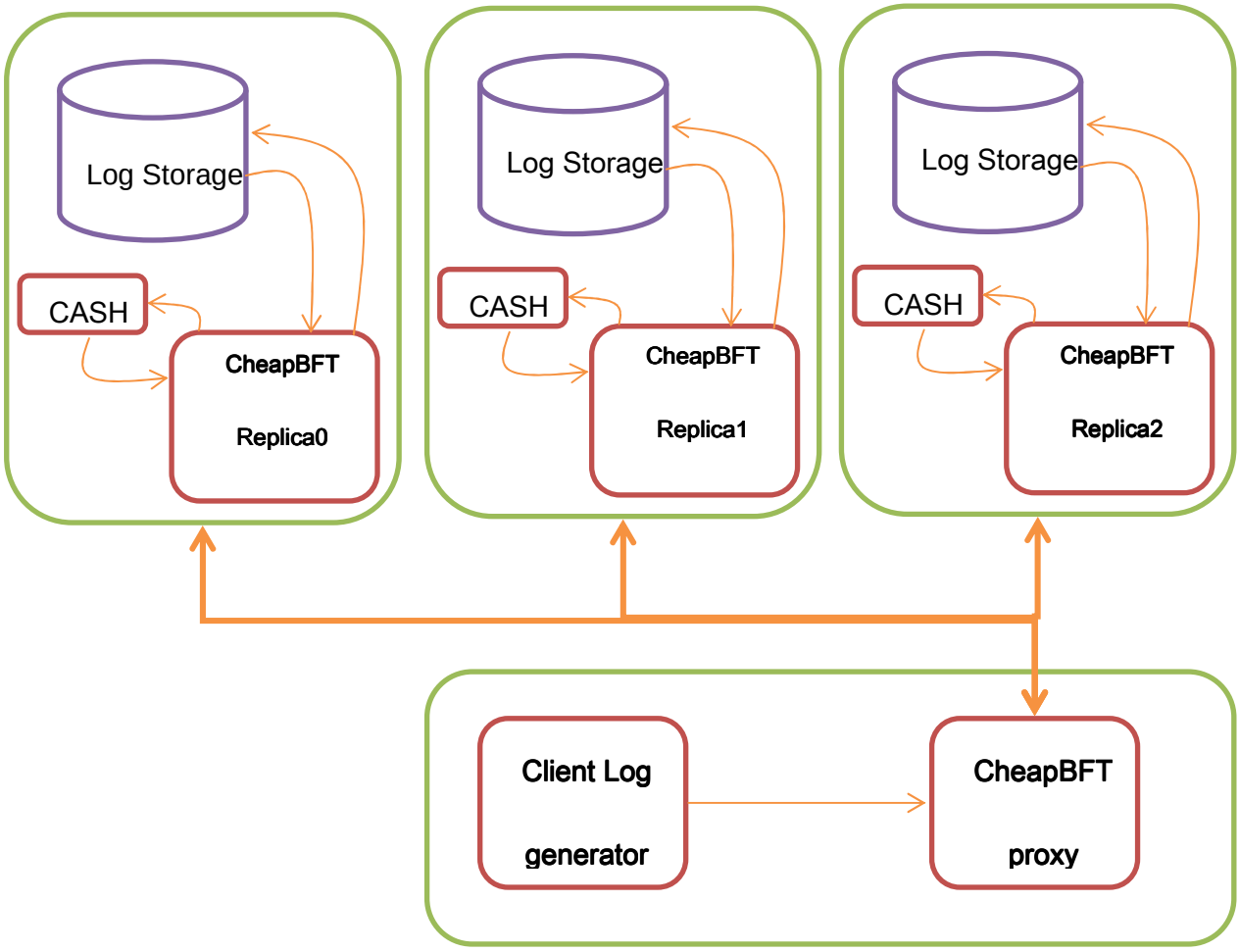


Figure 82 - Logical architecture of CheapBFT validation scenario

3.1.3.3 Validation setup

The validation activity has been held in San Raffaele facilities and since CheapBFT protocol requires CASH-compliant hardware, we opted to use its software implementation.

Thus, the architecture depicted in Figure 82 was hosted directly on one single machine. The following deployment scenario was used for the CheapBFT validation activity:

Machine Type: Virtual Machine
Operative System: Ubuntu 12.04
CPU: Quad Core 64Bit
RAM: 4GB
HD: 20GB
Required SW: Java virtual machine.

Description: CheapBFT-LogService bundle software has been installed in a specific directory at the file system. The CheapBFT-LogService bundle is a manually pre-configured CheapBFT protocol that works with the Log Service TClouds subsystem. It has been properly crafted in order to run 2 active replicas and 1 passive replica that run the Log Service Server component with a software replacement for the FPGA hardware module and one Log Service client connected by means of a CheapBFT proxy.

The Log Service client has been set up to write continuously new log entries to the Log Server and to retrieve the stored log in periodic intervals.

The CheapBFT-LogService bundle has been configured in such a way that it can be started in an easy and straightforward manner with a simple API. As you start the bundle system, it will automatically set up the two active replicas, the passive replica, and the client.

```
#./run_logsrv.bash start_demo
```

The system generates an extensive log file set. It allows direct and easy access to understand the behavior of the whole system.

3.1.3.4 Validation execution

In order to execute properly the validation, some specific care has been taken to monitor all the output fluxes and resource monitoring.

More specifically:

- The three replicas run on three different processes. All three replica processes has been monitored in order to analyze their resource usage to determine whether each replica is in active or passive state. For that purpose, the following command has been used:

  ```
  $sudo top -b -p <PID_number> -d 1 | grep <PID_number> --line-buffered > ReplicaOutput.log
  ```

Where <PID_number> is the pid number of the process of replica (0,1, or 2)

- The output of all created screens has been logged in order to easily process them. Since all the output runs on Linux “screens”, the screen program has been started with the –L option:

  ```
  $sudo screen –L
  ```

- It has been taken care also of the Log Service Storage, the state of the three replicas, and the used Cheap* protocols.

In order to run the CheapBFT protocol properly, some initial setup has been made:

```
# Configure CheapBFT as consensus protocol and a software based verification module
$./run_logsrv.bash setup_prot cheap.soft

# Start all components on localhost
$./run_logsrv.bash setup_hosts local

# Set up context related configuration
$./run_logsrv.bash setup_env valid

# Enable logging
$./run_logsrv.bash setup_logging con FINE
```

Than the screen program has been started in order to allow the bundle software to work:

```
$sudo screen –L
```

Than the whole demo has been launched:

```
$ ./run_logsrv.bash start_demo -- logsrv.retrvint 100
```

The above command will:

- Start the three LogService replicas, placing their output on three different screen windows
- Start the CheapBFT proxy
- Start the Log Service client continuously storing new entries and retrieving the stored log after every 100 store operations.

Log entries stored are simple dummy strings like the following:

```
Mon Aug 19 13:24:44 CEST 2013 520
```

Composed of 33 Bytes for each log line

At this stage, the system is up and running. It is possible to see the CPU resource usage in Figure 83 (refer from second 100 to second 244) in which Replica0 and Replica1 have high resource consumption while Replica2 (the passive one) has lower.

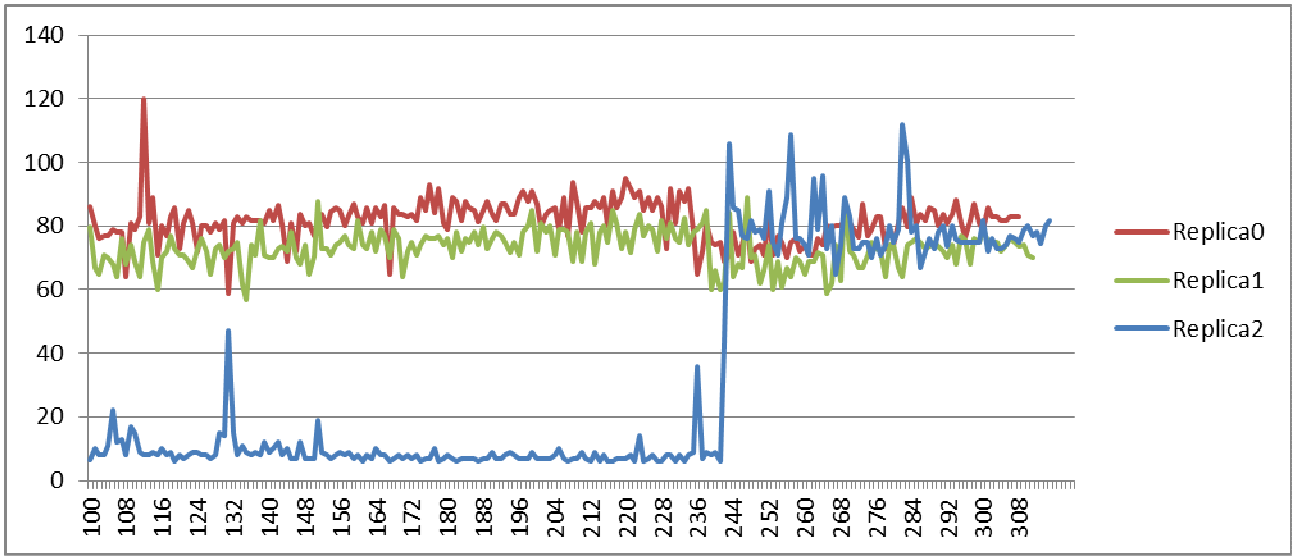


Figure 83 - CPU resource consumption of CheapBFT replica

Moreover, by watching the output file, we can see that the client is correctly sending and retrieving data:

```
logsrv.nclients      1
logsrv.nwarmclients  1
logsrv.warmup        0
logsrv.run           -1
logsrv.pause         10
O 2013-08-19 13:24:41.953 Using the autodetected NIO constraint level: 0
O 2013-08-19 13:24:42.062 Connecting to server 0: clientid 10 serveraddr localhost/127.0.0.1:13000
O 2013-08-19 13:24:42.082 Connecting to server 1: clientid 10 serveraddr localhost/127.0.0.1:13010
O 2013-08-19 13:24:42.103 Channel connected: clientid 10 serveraddr localhost/127.0.0.1:13000
O 2013-08-19 13:24:42.103 Channel connected: clientid 10 serveraddr localhost/127.0.0.1:13010
O 2013-08-19 13:24:42.115 Connecting to server 2: clientid 10 serveraddr localhost/127.0.0.1:13020
O 2013-08-19 13:24:42.117 Channel connected: clientid 10 serveraddr localhost/127.0.0.1:13020
O 2013-08-19 13:24:42.181 Init sequence number: clientid 10 seqno 0
logsrv.retrvint      100
logsrv.reqpause      0
logsrv.output        0
Warm up...
Get serious!
Test run with 1 clients, -1 secs
    1 cnt      173  time   996969  avg     5762  min     3903  max    22791
```

```
    2 cnt      194  time   998743  avg     5148  min     3454  max    16538
    3 cnt      202  time   998547  avg     4943  min     2772  max    17208
    4 cnt      226  time  1000498  avg     4426  min     2571  max    11785
    5 cnt      245  time  1000321  avg     4082  min     2599  max     9704
    6 cnt      252  time  1001656  avg     3974  min     2548  max     8839
                                              …
                                              …
                                              …
  236 cnt      365  time  1010789  avg     2769  min      907  max    96434
  237 cnt      335  time   967749  avg     2888  min      947  max   135802
  238 cnt      351  time  1016591  avg     2896  min     1038  max   105060
  239 cnt      358  time  1017825  avg     2843  min     1027  max   104344
  240 cnt      391  time   951402  avg     2433  min     1001  max    94057
```

Listing 12 - Snippet of client output

From the beginning up to the second 240, the client has sent and received correctly the data.

Each line has the following form:

By summing all storage requests sent to the replicas, we reach 93999 log records stored and retrieved correctly. Note that a retrieval of the log is done every 100 storage request, ensuring that the systems is still able to return the log In this case, it has been done 9399 times. At the 94000th storage request, a new retrieval will be triggered.

3.1.3.4.1.1 Tampering the log storage and detection

At this point, we are ready to tamper the log storage of one of the two active replicas.

To do this, we will invoke a specific command from the CheapBFT demo console:

```
$ ./run_logsrv.bash inderr delline 2 replica 1
```

That deletes line 2 of Replica1. This deletion will force the system to activate the passive replica in order to retrieve the correct data and continue providing the correct result to the client

During the validation execution, this error has been introduced while saving the 94103rd log record, in fact we can see the Replica0 log:

```
O 2013-08-19 13:28:43.051 Client msg already handled: TOMMessage - sender 10 body 74 seq 94101 ro false
hash false contentlen 61; fromclient false, lastseq 94101
O 2013-08-19 13:28:43.061 Checkpoint accepted: 94100 inmajority true
O 2013-08-19 13:28:43.176 New batch of pending requests: size 1 remlistsize 0
O 2013-08-19 13:28:43.186 New batch of pending requests: size 0 remlistsize 0
O 2013-08-19 13:28:43.213 Client msg already handled: TOMMessage - sender 10 body 74 seq 94102 ro false
hash false contentlen 61; fromclient false, lastseq 94102
O 2013-08-19 13:28:43.216 New batch of pending requests: size 1 remlistsize 0
O 2013-08-19 13:28:43.216 Client msg already handled: TOMMessage - sender 10 body 74 seq 94103 ro false
hash false contentlen 61; fromclient false, lastseq 94103
!        2013-08-19        13:28:43.248        Checkpoints        don't        match:        localhash
a152c45d6e548dfd_c6d9b435a741f3b2_be11408562ada98d_f04b89df7d246113 received CheckpointMessage - sender
1    body    145    prot    0    <MC    -    procid    1    ac    95988    uc    11294    cm    ALL    hmac
b8598aff13faf9eb_2898a8cdf5f3ef7e_e64812371178fbe3_47607aba29a9b61d>       consid       94103       hash
4defed2a6c484d56_fcad01842664a305_1e8b80d6d03748a7_910005dc8232f4e1
O 2013-08-19 13:28:43.248 New batch of pending requests: size 1 remlistsize 0
```

```
O 2013-08-19 13:28:43.249 Client msg already handled: TOMMessage - sender 10 body 74 seq 94104 ro false
hash false contentlen 61; fromclient false, lastseq 94104
~ 2013-08-19 13:28:43.261 Switch protocol from CheapTiny to MinBFT
O 2013-08-19 13:28:43.267 New batch of pending requests: size 1 remlistsize 0
O 2013-08-19 13:28:43.281 Client msg already handled: TOMMessage - sender 10 body 74 seq 94105 ro false
hash false contentlen 61; fromclient false, lastseq 94105
O 2013-08-19 13:28:43.303 New batch of pending requests: size 1 remlistsize 0
```

Listing 13 - snippet of Replica0 outcome

The bold text above shows that Replica0 (that is acting as master replica) has detected the fault and has switched from CheapTiny to MinBFT protocol. This switch has caused the passive replica to wake up and start working actively. This can be seen after second 241 in Figure 83.

The client has received responses from all three replicas and detected that one value (from Replica1) is corrupted while performing the next retrieve request:

```
  238 cnt      351  time  1016591  avg     2896  min     1038  max   105060
  239 cnt      358  time  1017825  avg     2843  min     1027  max   104344
  240 cnt      391  time   951402  avg     2433  min     1001  max    94057
  241 cnt      200  time  1026648  avg     5133  min     1144  max   126583
O 2013-08-19 13:28:43.751 Replies don't match: clientid 10 replies TOMMessage - sender 0 body 3439449
seq 94200 ro false hash false contentlen 3439436 <-> TOMMessage - sender 1 body 3439416 seq 94200 ro
false hash false contentlen 3439403
O 2013-08-19 13:28:43.883 Replies don't match: clientid 10 replies TOMMessage - sender 0 body 3439449
seq 94200 ro false hash false contentlen 3439436 <-> TOMMessage - sender 1 body 3439416 seq 94200 ro
false hash false contentlen 3439403
O 2013-08-19 13:28:44.274 Replies don't match: clientid 10 replies TOMMessage - sender 1 body 3443079
seq 94300 ro false hash false contentlen 3443066 <-> TOMMessage - sender 2 body 3443112 seq 94300 ro
false hash false contentlen 3443099
O 2013-08-19 13:28:44.289 Replies don't match: clientid 10 replies TOMMessage - sender 0 body 3443112
seq 94300 ro false hash false contentlen 3443099 <-> TOMMessage - sender 1 body 3443079 seq 94300 ro
false hash false contentlen 3443066
O 2013-08-19 13:28:44.670 Replies don't match: clientid 10 replies TOMMessage - sender 0 body 3446775
seq 94400 ro false hash false contentlen 3446762 <-> TOMMessage - sender 1 body 3446742 seq 94400 ro
false hash false contentlen 3446729
  242 cnt      200  time   930256  avg     4651  min     1309  max   229124
O 2013-08-19 13:28:44.707 Replies don't match: clientid 10 replies TOMMessage - sender 0 body 3446775
seq 94400 ro false hash false contentlen 3446762 <-> TOMMessage - sender 1 body 3446742 seq 94400 ro
false hash false contentlen 3446729
O 2013-08-19 13:28:45.093 Replies don't match: clientid 10 replies TOMMessage - sender 1 body 3450405
seq 94500 ro false hash false contentlen 3450392 <-> TOMMessage - sender 2 body 3450438 seq 94500 ro
false hash false contentlen 3450425
```

Listing 14 - snippet of Replica0 output

At the end of time 241, the client has been sent 94199 storage requests. The corruption has been done during store request number 94013 between second 240 (at 93999 store request) and second 241. In fact, the client knows, starting from the 9420th retrieve request, that Replica1 has been corrupted.

It can be seen in the bold text above that during retrieve request 9430, the responses from the three replicas have the following sizes:

	Replica0	Replica1	Replica2
body	3443112	3443079	3443112
contentlen	3443099	3443066	3443099

Table 15- Replica replies dimension after tampering detection

As it can be seen from the above table, the difference between the two replicas is of 33 Bytes, that matches exactly a single log line dimension.

As it can be seen in Listing 14, at second 242 the client has managed to store correctly other 200 log records.
As it can be seen in Figure 84, it is also possible to see how the server replicas have been able to process all the requests. Inevitably, the request right after the second 241 required almost double the time since the CheapBFT protocol had to manage to wake up the passive replica and manage the protocol switch to MinBFT.

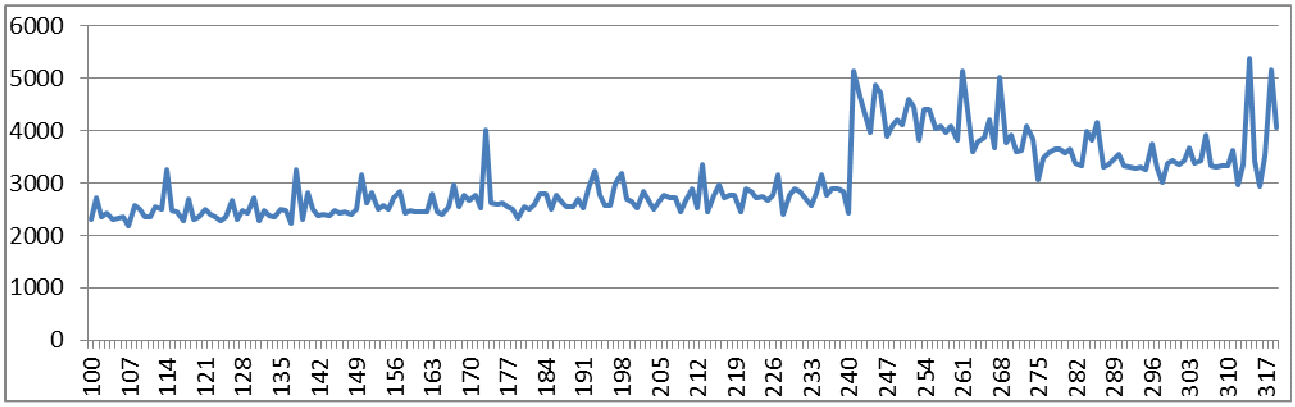


Figure 84 - Average time per each store request handled by the CheapBFT protocol

3.1.3.5 Conclusion

By examining CheapBFT's behavior and by executing the validation activity as described, we can assess that the CheapBFT subsystem work as expected, allowing Byzantine fault-tolerance with $f+1$ active replicas by switching to $2f+1$ active replicas once a tampering has been detected.

Requirements' assessment

LREQ1 – Availability and integrity of personal data - The Cheap-BFT subsystem improves the availability by providing fault tolerance, i.e. it can mask arbitrary faults in the cloud infrastructure. This also includes arbitrary alterations in the transmitted data, ensuring the integrity of the exchanged information.

AHSECREQ2 - Integrity of stored and transmitted data – Integrity of application state is checked against other replicas when externalized, i.e. transmitted to the client. Modifications can be detected and masked, thus ensuring integrity.

AHSECREQ3 - Integrity of the application – The integrity of the application instance is ensured by state machine replication. If the application misbehaves on one of the replicas in a way that is externally visible, the CheapBFT subsystem can detect and mask this fault.

AHSECREQ4 - Availability of stored and transmitted data – As the CheapBFT subsystem is based on replication, it does not only tolerate manipulation of data, but also the complete outages of single replicas.

AHSECREQ5 - Availability of the application – Since the replicas can run any application that can be modeled as a deterministic state machine, most software used in practice could be ported to use CheapBFT for improved availability. Data storage is actually just one of the simpler applications that can be built on top of the CheapBFT protocol.

AHSECREQ8 - Data source authentication – CheapBFT is based on the assumption that the clients and servers mutually authenticate each other

The outcome of the validation activity is POSITIVE.

3.1.4 DepSky Validation Activity

In the following chapter are described the validation activities for DepSky subcomponents.

Validation activities have been slightly changed as described in D3.3.3, since we discovered, while performing the validation, some checks were not really effective for validation purposes. The final validation activities of DepSky consist In the ones described below.

Activity ID	**DepSky_1**
Activity type	Proof of Concept
Activity description	The Home Healthcare appliance is deployed and running onto the Trustworthy OpenStack TClouds prototype. The PHR database VM has installed the DepSky drivers 1- From the Home Healthcare administrator interface define all the setup information to connect to the different cloud providers 2- Perform a snapshot of the PHR data by zipping the dump of the database 3- Send the files on the different cloud by using the DepSky driver 4- Check in the commodity clouds that the file is saved correctly 5- Clean fs cache and force a synchronization with remote clouds 6- Check whether the file has been downloaded and compare it with the original file. 7- Remove all the files from one commodity cloud replica 8- From the healthcare administrator console perform a restore of an old backup 9- Check that the backup data is consistent as the original backup 10- Remove all the files from another commodity cloud replica 11- From the healthcare administrator console perform a restore of an old backup 12- Check that the backup data is not available anymore 13- Perform the same steps as 3…12 by tampering a remote replica file instead of removing it 14- Perform the same steps as 3…12 by either tampering and removing a remote replica file.
Acceptance Criteria	The Activity is passed if: • At point 4 original file and remote file are the same • At point 6 original file and remote file are the same • At point 12 remote file cannot be retrieved.
References Documents:	(Deliverable 2.2.1, 2010) (Deliverable D2.4.2, 2012) (Depsky, 2011) (C2FS)
Requirements satisfied	LREQ1, LREQ2, AHSECREQ1, AHSECREQ2

Table 16 - DepSky_1 validation activity

From this activity has been added the resiliency checks: we want to assess that the subcomponent is able to retrieve data even if one of the four replica is not available anymore. DepSky works with 3f+1 replica where f represents the number of faults that is able to manage. In our case we used 4 replica, thus f=1.

Activity ID	DepSky_2
Activity type	Performance test
Activity description	The Home Healthcare appliance is deployed and running onto the Trustworthy OpenStack TClouds prototype. The PHR database VM has installed the DepSky drivers 1- From the Home Healthcare administrator interface define all the setup information to connect to the different cloud providers 2- Define an incremental file span from 1MB up to 10 MB per file 3- Perform a snapshot of the PHR databases by zipping and spanning the file 4- Send the files on the different cloud by using the DepSky driver 5- Perform a restore of every file. Check the quantity of data that has been sent/retrieved compared with the dimension of the original file.
Acceptance Criteria	The activity is passed if c2fs stored data does not increase over 100%.
References Documents:	(Deliverable 2.2.1, 2010) (Deliverable D2.4.2, 2012) (Depsky, 2011) (C2FS)
Requirements satisfied:	LREQ1, LREQ2, AHSECREQ1, AHSECREQ2

Table 17 - DepSky_2 validation activity

This validation activity has been modified from a comparison with a commodity cloud with a check of overhead produced. The reason of this switch resides in the fact that velocity comparison does not work properly when the bandwidth of internet connection is easy to fluctuate significantly, as in the case of the San Raffaele facilities, where bandwidth may vary from time to time. This creates an unfear and non-effective way to validate the subcomponent.

3.1.4.1 DepSky features

The Depsky cloud-of-clouds object storage service uses object storage services from diverse cloud providers (e.g., Amazon S3, Rackspace Files) to build a dependable object storage service.

The core of the solution is a set of read/write protocols based on the use of Byzantine quorum replication [2] requiring 3f+1 clouds to tolerate up to f unavailable/compromised clouds. This proto-col addresses the mentioned requirements in the following way:

1) The system tolerates arbitrary (a.k.a. Byzantine) faults in order to cope with all possible behavior of a fraction of providers;
2) The replication protocol operates on an unreliable network in which messages can be lost and delayed and do not require participation of the full set of employed clouds, but only of a sub-set of them (a quorum [1]), on any step of the protocol execution;
3) The protocols are completely client-based, in the sense that no specific code is required in the cloud. DepSky assume the clouds provide storage service with standard (RESTful) operations for managing objects and containers (put, get, list, etc.). Moreover, the set of storage clouds do not interact among themselves, but only with the clients;

3.1.4.2 Validation Scenario

The scenario we have depicted for this validation activity make direct use of TPaaS, the Healthcare Trustworthy Platform scenario. Below is depicted the high level deployment architecture:

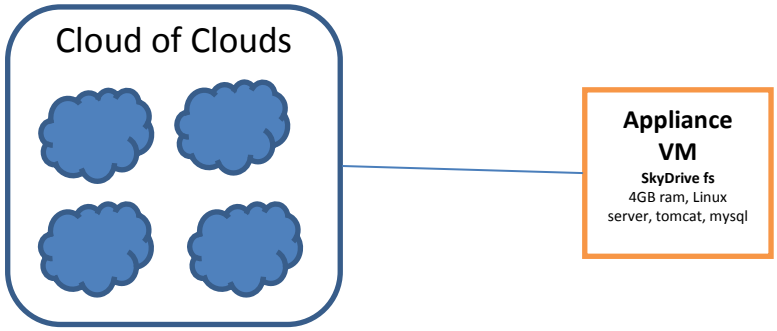


Figure 85 - Deployment scenario: The healthcare Applicance VM is connected with all the commodity clouds throught Cloud of Cloud sub-component (DepSky)

The Healthcare Appliance VM uses C2FS (DepSky) file system drivers in order to connect to the different commodity clouds to store and retrieve the files that the Healthcare Appliance Produces. The Healthcare Platform has a backup system for PHR data that takes all the PHR information of the users and creates a backup file to be sent into DepSky FS.

3.1.4.3 Validation Setup

DepSky comes packaged in a zip file. By unzipping it is necessary just to modify a config file and the system is ready to run. The config file is necessary to allow DepSky to access to the remote commodity cloud space. For this Validation activity we decided to use 4 different grant access of an Amazon EC2 disk.

```
#AMAZON
driver.type=AMAZON-S3
driver.id=cloud1
accessKey=AKIAJBHT7PDKS7JOY7VA
secretKey=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
location=EU_Ireland
canonicalId=nothing

driver.type=AMAZON-S3
driver.id=cloud2
accessKey=AKIAIT5RXDBJ3ZKTZGXQ
secretKey=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
location=EU_Ireland
canonicalId=nothing

driver.type=AMAZON-S3
driver.id=cloud3
accessKey=AKIAIMJ7GHXZDSXWQL5A
secretKey=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
location=EU_Ireland
canonicalId=nothing

driver.type=AMAZON-S3
driver.id=cloud4
accessKey=AKIAIVCKG47IK4GSRP6A
secretKey=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
location=EU_Ireland
```

```
canonicalId=nothing
```

Table 18 - DepSky accounts.properties file

Now the filesystem is ready to start. By issuing the following command:

```
$./runC2FS
```

Depsky is able to connect to the remote commodity cloud and mount a specific folder that the component uses in order to store/retrieve files (`/c2fs_mountPoint`).

3.1.4.4 Validation Execution

The execution is done from the Home Healthcare web portal, once connected as portal administrator. From the UI is possible to create a backup of the PHR database:

Figure 86 - Backup feature at appliance level (Healthcare Platform, admin area)

In our example dump file named “1321b3d5-b26e-4afa-a579-5216fc477a00.zip” has been created. The file is automatically sent to the c2fs mounted directory and DepSky driver recognizes it and starts the synchronization with the remote clouds:

```
liferay@LiferayDevelopNew:~/C2FS$ sudo su
root@LiferayDevelopNew:/home/liferay/C2FS# ls
bin          c2fs_mountPoint  Coding  jni  outputErr.txt  percent     runC2FS.sh
build.conf   cache_4          config  lib  output.txt     README.txt  script.sh
root@LiferayDevelopNew:/home/liferay/C2FS# ./runC2FS.sh
root@LiferayDevelopNew:/home/liferay/C2FS# ls
bin          c2fs_mountPoint  Coding  jni  outputErr.txt  percent     runC2FS.sh
build.conf   cache_4          config  lib  output.txt     README.txt  script.sh
root@LiferayDevelopNew:/home/liferay/C2FS# cd c2fs_mountPoint/
root@LiferayDevelopNew:/home/liferay/C2FS/c2fs_mountPoint# ls
1321b3d5-b26e-4afa-a579-5216fc477a00.zip
root@LiferayDevelopNew:/home/liferay/C2FS/c2fs_mountPoint#
```

Figure 87 - check file presence in local mounted c2fs path

To make a double check we inspected directly into the remote storage and we can see that there is a new file in each of the four bucket designed for DepSky.

Figure 88 - Data stored in bucket1 by DepSky subcomponent

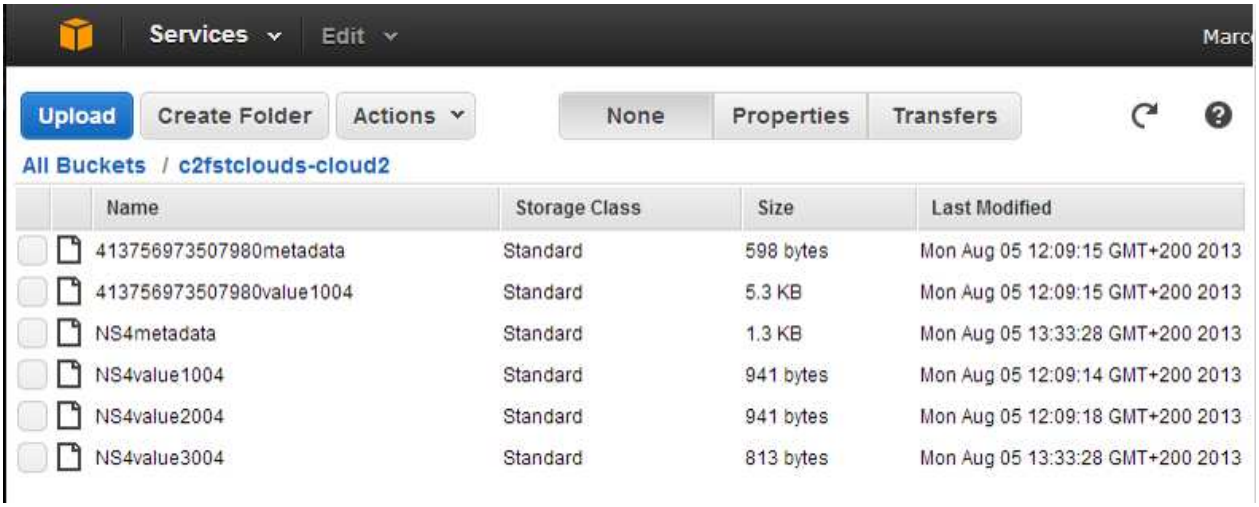


Figure 89 - Data stored in bucket2 by DepSky subcomponent

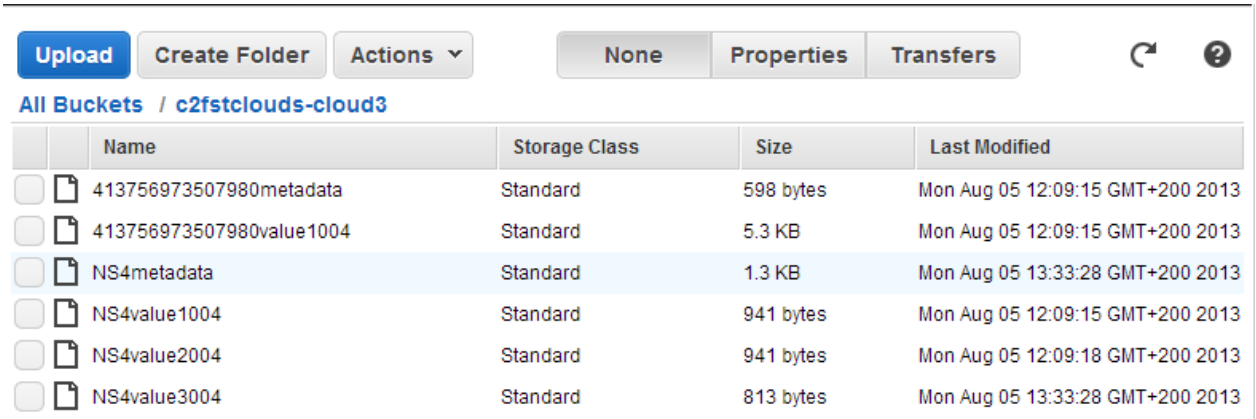


Figure 90 - Data stored in bucket3 by DepSky subcomponent

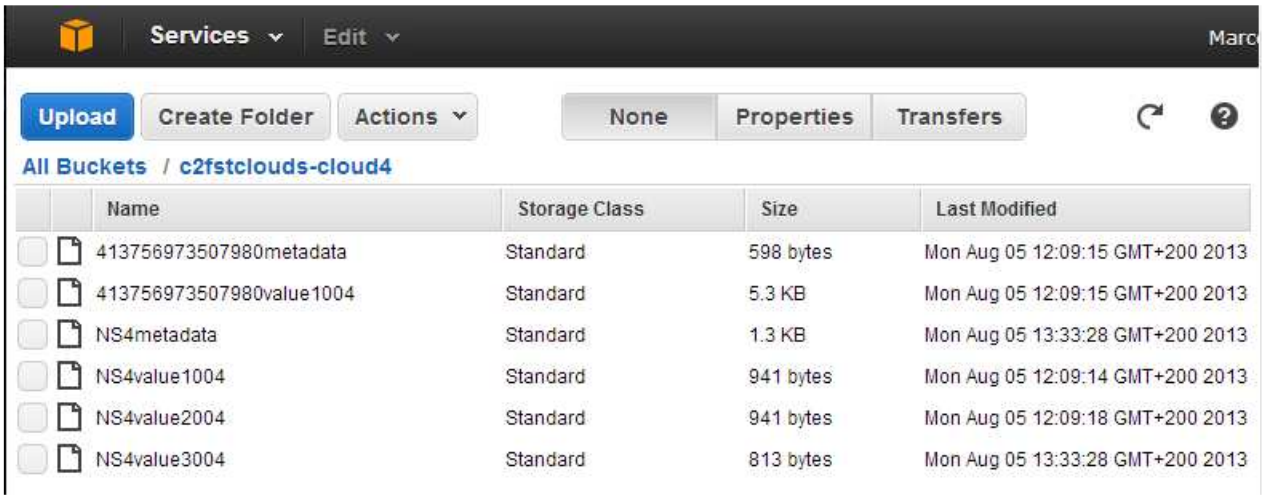


Figure 91 - Data stored in bucket4 by DepSky subcomponent

The file content has been saved as 41375697357980value1004 and there are four different files in the four buckets, each one containing a piece of the original zip file. DepSky has a verbose log that can be inspected as well:

```
 :::: WRITE( /1321b3d5-b26e-4afa-a579-5216fc477a00.zip, isWritepage: false, offset: 8265 )

 :::: GETXATTRSIZE(/1321b3d5-b26e-4afa-a579-5216fc477a00.zip, name: security.capability )

 :::: WRITE( /1321b3d5-b26e-4afa-a579-5216fc477a00.zip, isWritepage: false, offset: 9584 )

 :::: GETXATTRSIZE(/1321b3d5-b26e-4afa-a579-5216fc477a00.zip, name: security.capability )

 :::: WRITE( /1321b3d5-b26e-4afa-a579-5216fc477a00.zip, isWritepage: false, offset: 9600 )

 :::: FLUSH( /1321b3d5-b26e-4afa-a579-5216fc477a00.zip )

 :::: RELEASE ( /1321b3d5-b26e-4afa-a579-5216fc477a00.zip, flags: 32770 )

 :::: GETDIR( / )
NonSharingDis:
        -> 1321b3d5-b26e-4afa-a579-5216fc477a00.zip

 :::: GETATTR( /1321b3d5-b26e-4afa-a579-5216fc477a00.zip )

 :::: GETATTR( /.statistics.txt )
-> vou enviar: NS4
  -> Start upload at depsky
-> vou enviar: 413756973507980
  -> Start upload at depsky
  -> End upload at depsky
  -> Upload operation took: 2100 milis
commitMetadata: NS4, this.idPATH = NS4
  -> End upload at depsky
  -> Upload operation took: 1819 milis
commitMetadata: 413756973507980, this.idPATH = NS4
-> vou enviar: NS4
  -> Start upload at depsky
  -> End upload at depsky
  -> Upload operation took: 1118 milis
commitMetadata: NS4, this.idPATH = NS4

 :::: GETATTR( / )

 :::: GETATTR( / )

 :::: GETDIR( / )
NonSharingDis:
        -> 1321b3d5-b26e-4afa-a579-5216fc477a00.zip

 :::: GETATTR( /1321b3d5-b26e-4afa-a579-5216fc477a00.zip )

 :::: GETATTR( / )
```

Figure 92 - Inspecting DepSky log file, upload of backup file has done successfully

To make better use of resources DepSky adopt a caching techniques, that is, all the files stored remotely are maintained also locally. In order to validate the resiliency capabilities, however, we need to remove the cache and force DepSky to ask a new synchronization with the remote clouds.

```
root@LiferayDevelopNew:/home/liferay/C2FS# umount c2fs_mountPoint/
root@LiferayDevelopNew:/home/liferay/C2FS# ls
bin         c2fs_mountPoint  Coding  jni  outputErr.txt  percent     runC2FS.sh
build.conf  cache_4          config  lib  output.txt     README.txt  script.sh
root@LiferayDevelopNew:/home/liferay/C2FS# cd cache_4/
root@LiferayDevelopNew:/home/liferay/C2FS/cache_4# ls
data directory  info directory
root@LiferayDevelopNew:/home/liferay/C2FS/cache_4# cd data\ directory/
root@LiferayDevelopNew:/home/liferay/C2FS/cache_4/data directory# rm *
root@LiferayDevelopNew:/home/liferay/C2FS/cache_4/data directory# ls
root@LiferayDevelopNew:/home/liferay/C2FS/cache_4/data directory# cd ..
root@LiferayDevelopNew:/home/liferay/C2FS/cache_4# cd info\ directory/
root@LiferayDevelopNew:/home/liferay/C2FS/cache_4/info directory# sudo rm *
root@LiferayDevelopNew:/home/liferay/C2FS/cache_4/info directory# ls
root@LiferayDevelopNew:/home/liferay/C2FS/cache_4/info directory# cd ..
root@LiferayDevelopNew:/home/liferay/C2FS/cache_4# cd ..
root@LiferayDevelopNew:/home/liferay/C2FS# ./runC2FS.sh
```

Figure 93 - deleting local cache of file saved

We also restarted the client driver itself, in order to emulate a shutdown of the machine

```
Debug = true
Use memory cache = false
Assync model = false
Is non-blocking to cloud = true
NonSharing = true
starting drivers...
All drivers started.
starting drivers...
All drivers started.
starting drivers...
All drivers started.
starting drivers...
All drivers started.
starting drivers...
All drivers started.
  -> Start download at depsky
----------------
----------------
Â»Â»Â» Init called ...
  -> End download at depsky
  -> Download operation took: 653 milis
A NS was found.
DELTA = 500
C2FS mounted.
```

Figure 94 - mounting and sync of DepSky driver with remote buckets

As expected the file has been downloaded and it is ready again into the mounted DepSky's path

```
root@LiferayDevelopNew:/home/liferay/C2FS# ./runC2FS.sh
root@LiferayDevelopNew:/home/liferay/C2FS# cd c2fs_mountPoint/
root@LiferayDevelopNew:/home/liferay/C2FS/c2fs_mountPoint# ls
1321b3d5-b26e-4afa-a579-5216fc477a00.zip
root@LiferayDevelopNew:/home/liferay/C2FS/c2fs_mountPoint#
```

Figure 95 - File is again ready locally after c2fs driver restart

To be sure that the file is integer and the same as the original one we used “diff” program:

```
root@LiferayDevelopNew:/home/liferay/C2FS/c2fs_mountPoint# diff -c 1321b3d5-b26e-4afa-a579-5216fc47
7a00.zip /home/liferay/1321b3d5-b26e-4afa-a579-5216fc477a00.zip
root@LiferayDevelopNew:/home/liferay/C2FS/c2fs_mountPoint#
```

Figure 96 - file difference between remote file and original file

Note that diff does not provide any output if the two files are identical.

3.1.4.4.1.1 Deleting from cloud

We re-did the previous process (remove cache, force download) twice more: the first time removing one remote replica, the second time removing a second replica:

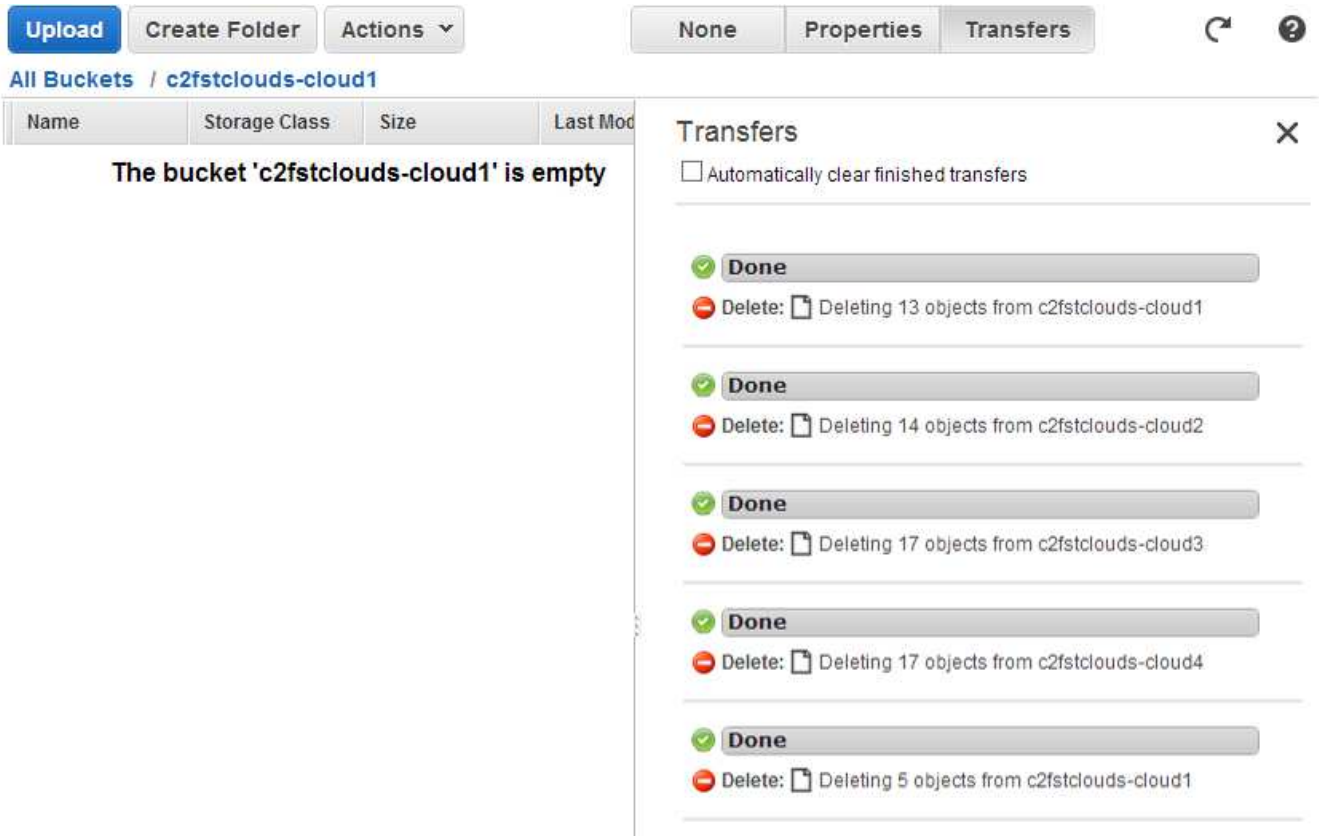


Figure 97 - Deletion of first replica (bucket1). Please note that the last deletion is the lower one. Previous log lines refers to attempts and tests performed before.

```
root@LiferayDevelopNew:/home/liferay/C2FS/cache_4# cd data\ directory/
root@LiferayDevelopNew:/home/liferay/C2FS/cache_4/data directory# rm *
root@LiferayDevelopNew:/home/liferay/C2FS/cache_4/data directory# cd ..
root@LiferayDevelopNew:/home/liferay/C2FS/cache_4# cd info\ directory/
root@LiferayDevelopNew:/home/liferay/C2FS/cache_4/info directory# rm *
root@LiferayDevelopNew:/home/liferay/C2FS/cache_4/info directory# cd ..
root@LiferayDevelopNew:/home/liferay/C2FS/cache_4# cd .
root@LiferayDevelopNew:/home/liferay/C2FS/cache_4# cd ..
root@LiferayDevelopNew:/home/liferay/C2FS# ./runC2FS.sh
root@LiferayDevelopNew:/home/liferay/C2FS#
```

Figure 98 - restart DepSky

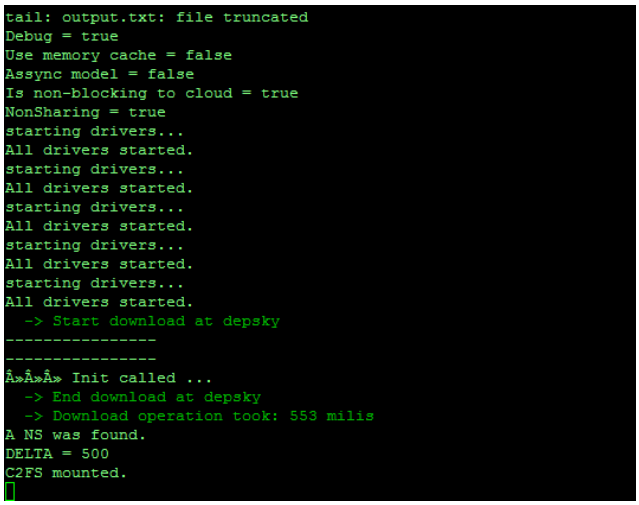


Figure 99 - DepSky has successfully re-synced with the remote replica

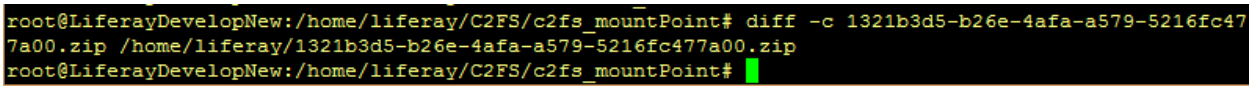


Figure 100 - Local file and restored file form remote have no differences.

As expected even removing one remote piece of file the system is able to reconstruct the file from the other 3 pieces and deliver it correctly to the client.

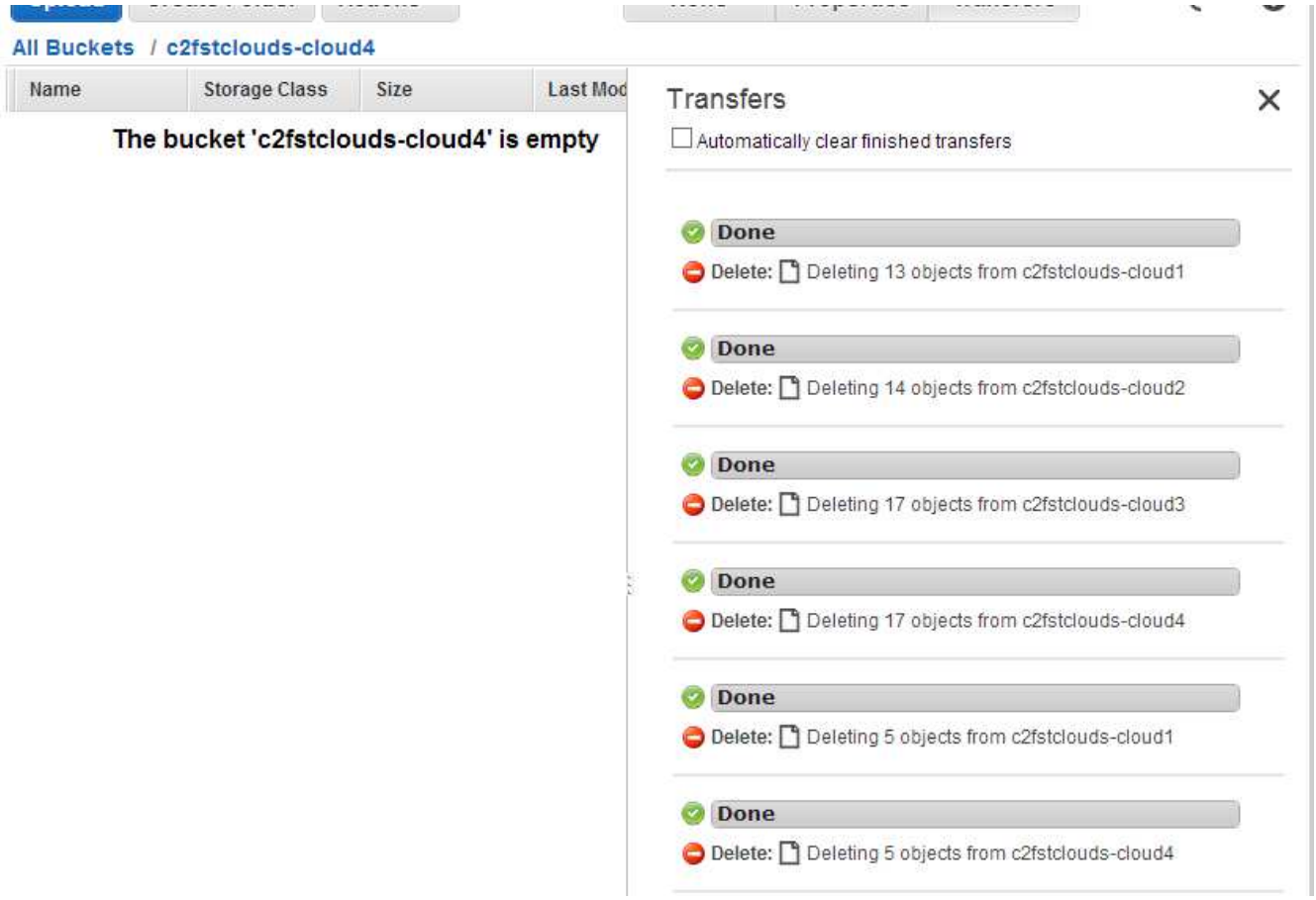


Figure 101 - Deletion of fourth replica (bucket4). Please note that the last deletion is the lower one. Previous log lines refers to attempts and tests performed before.

```
root@LiferayDevelopNew:/home/liferay/C2FS# cd cache_4/
root@LiferayDevelopNew:/home/liferay/C2FS/cache_4# cd data\ directory/
root@LiferayDevelopNew:/home/liferay/C2FS/cache_4/data directory# rm *
root@LiferayDevelopNew:/home/liferay/C2FS/cache_4/data directory# ls
root@LiferayDevelopNew:/home/liferay/C2FS/cache_4/data directory# cd ..
root@LiferayDevelopNew:/home/liferay/C2FS/cache_4# cd info\ directory/
root@LiferayDevelopNew:/home/liferay/C2FS/cache_4/info directory# rm *
root@LiferayDevelopNew:/home/liferay/C2FS/cache_4/info directory# ls
root@LiferayDevelopNew:/home/liferay/C2FS/cache_4/info directory# cd ..
root@LiferayDevelopNew:/home/liferay/C2FS/cache_4# cd ..
root@LiferayDevelopNew:/home/liferay/C2FS# ./runC2FS.sh
root@LiferayDevelopNew:/home/liferay/C2FS# cd c2fs_mountPoint/
root@LiferayDevelopNew:/home/liferay/C2FS/c2fs_mountPoint# ls
root@LiferayDevelopNew:/home/liferay/C2FS/c2fs_mountPoint# ls
root@LiferayDevelopNew:/home/liferay/C2FS/c2fs_mountPoint#
```

Figure 102 - local cache deletion

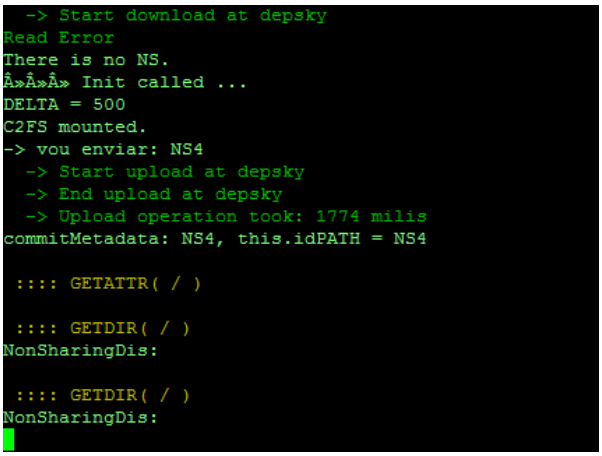


Figure 103 - DepSky restart, no data is sync since remote replica are not enough to reconstruct the original data

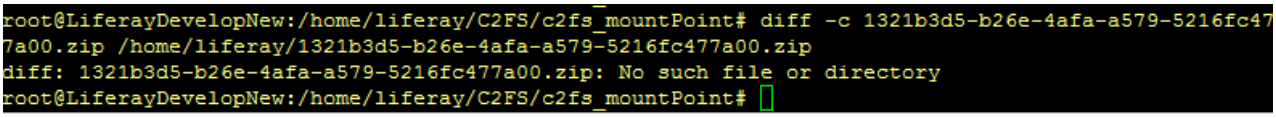


Figure 104 - no files available into DepSky path. Diff fails

As expected at this stage the file wasn't recovered, since the system is setup to work with 3f+1 replicas, where f (number of faulty replica) in this case is equal to 1.

3.1.4.4.1.2 Tampering replica and byzantine attack

We performed also another type of attack which consists in modify a remote replica. DepSky is able to reconstruct the original file also in this case.

We continued stressing the "byzantine" concept and we performed a multiple attach, by modifying a replica file and altering another replica of another file, also including the configuration files that DepSky stores remotely. The results have been positive also in this cases.

We are not going to show all the steps performed since they are totally similar to the steps performed before.

3.1.4.4.1.3 Performance checks

This validation activity has been conducted by spanning a PHR dump file over zip files of different dimension. We started with spanned files of 1MB up to 10MB with 1MB of increment every time. The following graph shows the different dimension of the files uploaded, downloaded when there are all the replica available, downloaded when one replica is missing, and the stored dimension.

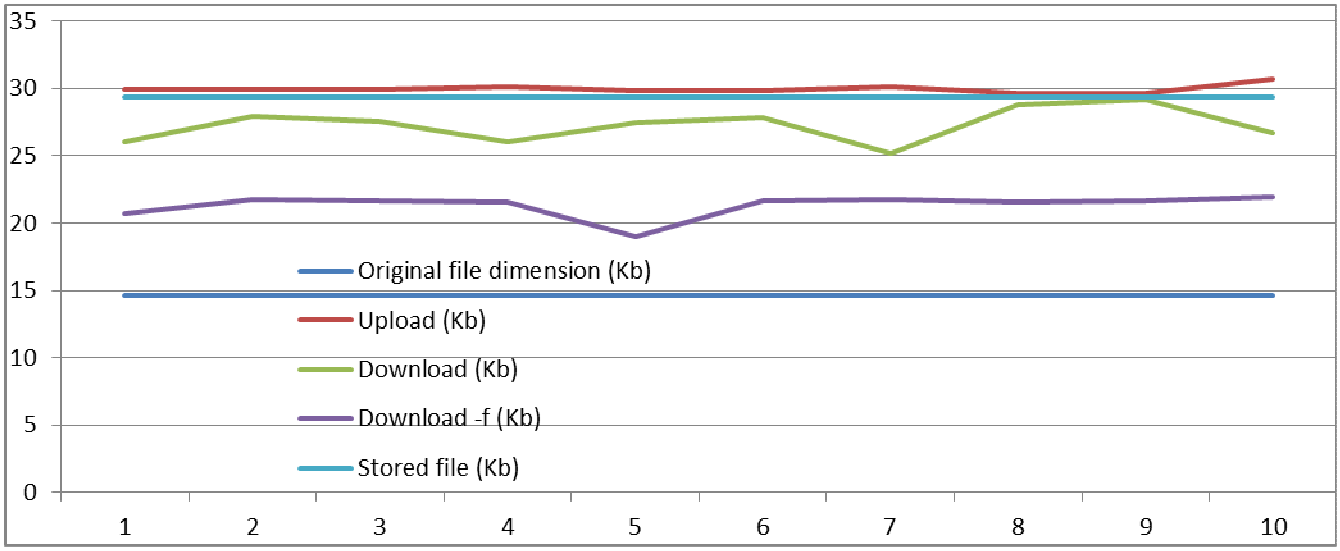


Figure 105 - Dimension of stored file and transmitted data (Scale is in Thousands of Kb – x1000)

What we noticed immediately is the increase of data stored that is doubled. This is due to the fact that data is encrypted and it has redundancy in an Erasure Code fashion. It may be acceptable from a business perspective considering the benefits that this subcomponents allow to gain. Moreover, it can be noticed that transmitted data while downloading is lower than the actual stored data. This is due to the ability of the subcomponent to process received data more efficiently and avoid to download all the different pieces of data.

Moreover, in the case one replica is not available, downloaded data is just around 30% more than the original data size since the sources from which the data is gathered are less.

3.1.4.5 Conclusion

As seen in the previous chapters, DepSky subcomponent has interesting capabilities since we can send encrypted data of pieces of files into standard, untrusted, commodity clouds having no worries that the data can be read or reconstructed since it bases itself on the easy and obvious business assumption that different legal companies, owning different cloud systems will never share their data, thus data cannot be reassemble and deciphered back.
Moreover, Healthcare PHR data (that is not sensitive to legal issues) can be stored anywhere in the world. This subcomponent allows the Healthcare Platform to save its data to other clouds, reducing costs (that may derive of TClouds replication costs) by using free tiers from other cloud hosting providers.

Requirements' assessment
LREQ1 - Confidentiality of personal data – & LREQ4 - Unlinkability and Intervenability - & AHSECREQ1 - Confidentiality of stored and transmitted data – This is ensured by storing only encrypted data on cloud providers. The keys used for encryption are either stored with the client or spread in several providers using secret sharing, ensuring no provider alone has access to the key.
LREQ2 - Availability and Integrity of personal data – & AHSECREQ2 - Integrity of stored and transmitted data – & AHSECREQ4 - Availability of stored and transmitted data – This is ensured by replicating the encrypted stored data in more than one cloud provider and by using novel Byzantine fault-tolerant protocols for reading and writing this data.

The final outcome of DepSky_1 and DepSky_2 Validation Activities is: SUCCESSFULLY PASSED.

3.1.5 LogService Validation Activity

This chapter describes the execution of LogService validation activity. The activity can be found in D3.3.4 and is reported below for sake of clearness.

Activity ID	**LogService_1**
Activity type	Proof of concept test
Activity description	The Home Healthcare appliance is deployed and running onto the Trustworthy OpenStack TClouds prototype. The LogService is up and running as well. The Home Healthcare appliance will perform activities in order to stress the LogService and proof its capabilities 1- A TPaaS user performs store and retrieve activities through a third party application into the Healthcare Platform. The platform logs all the entries that have been generated directly to the remote LogService Perform a request of verification of the logging session a. Perform a request of verification of the logging session using the ID provided at the previous step it is requested a verification of a specific session 2- Perform a download of the verified log as dump (Dump1) 3- Compromise the log storage by simulating an intrusion at application level 4- Request another verification of the logging session 5- Perform a download of the verified log as dump (Dump2) 6- Compare the results (Diff(Dump1,Dump2))
Acceptance Criteria	At point 7, after the verification, the system should warn the user that someone tried to compromise the log and a malicious action has been involved. Outcome: TRUE if verification fails. FALSE otherwise
Reference documents:	(TClouds factsheet - Log as a Service, 2013) (Deliverable D2.4.2, 2012) (Logging handlers) (Deliverable D2.1.2, 2012)
Requirements satisfied:	LREQ1, LREQ3, LREQ4, LREQ5, AHSECREQ6, AHSECREQ7, ASSECREQ1

Table 19 - LogService Validation Activity

We have further refined the validation activity by logging the platform's user activity by means of an external third party mobile demo application that sends and receives PHR and EHR data of the users of the Healthcare Platform.

3.1.5.1 LogService features

Logging is one of the more important administration tools of a complex IT system such as a cloud. The objective of such process is to track the events that happen in the system. Since the logs may be used to rebuild the past history of a system (e.g. after-the-facts analysis in forensics activity) the logging process is frequently victim of cyber-attacks. In order to consider logs as valid event/action evidence, it is necessary to provide procedures to attest their security in terms of integrity and authenticity. The LogService is a cloud oriented logging service that has been designed in order to support different secure logging schemes.

3.1.5.2 Validation scenario

The scenario we setup is depicted in the image below.

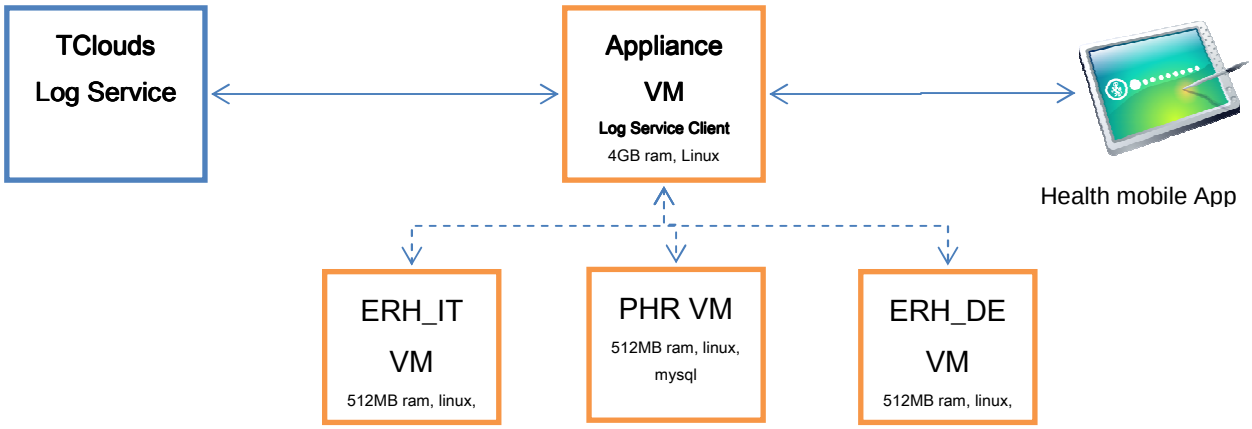


Figure 106 - LogService validation scenario

A third party demo mobile application has been used to send / retrieve Health data through the Healthcare Platform. The Appliance VM is the one that intercepts the request and properly redirect the logs to TClouds LogService.

3.1.5.3 Validation setup

In order to make the validation activity to work there are no particular set-up to be done since the log features are built-in into TPaaS Healthcare platform by means of client side code that communicated properly with the TClouds LogService.

3.1.5.4 Validation execution

The first step we performed is to use the mobile app to send the data (Figure 107).

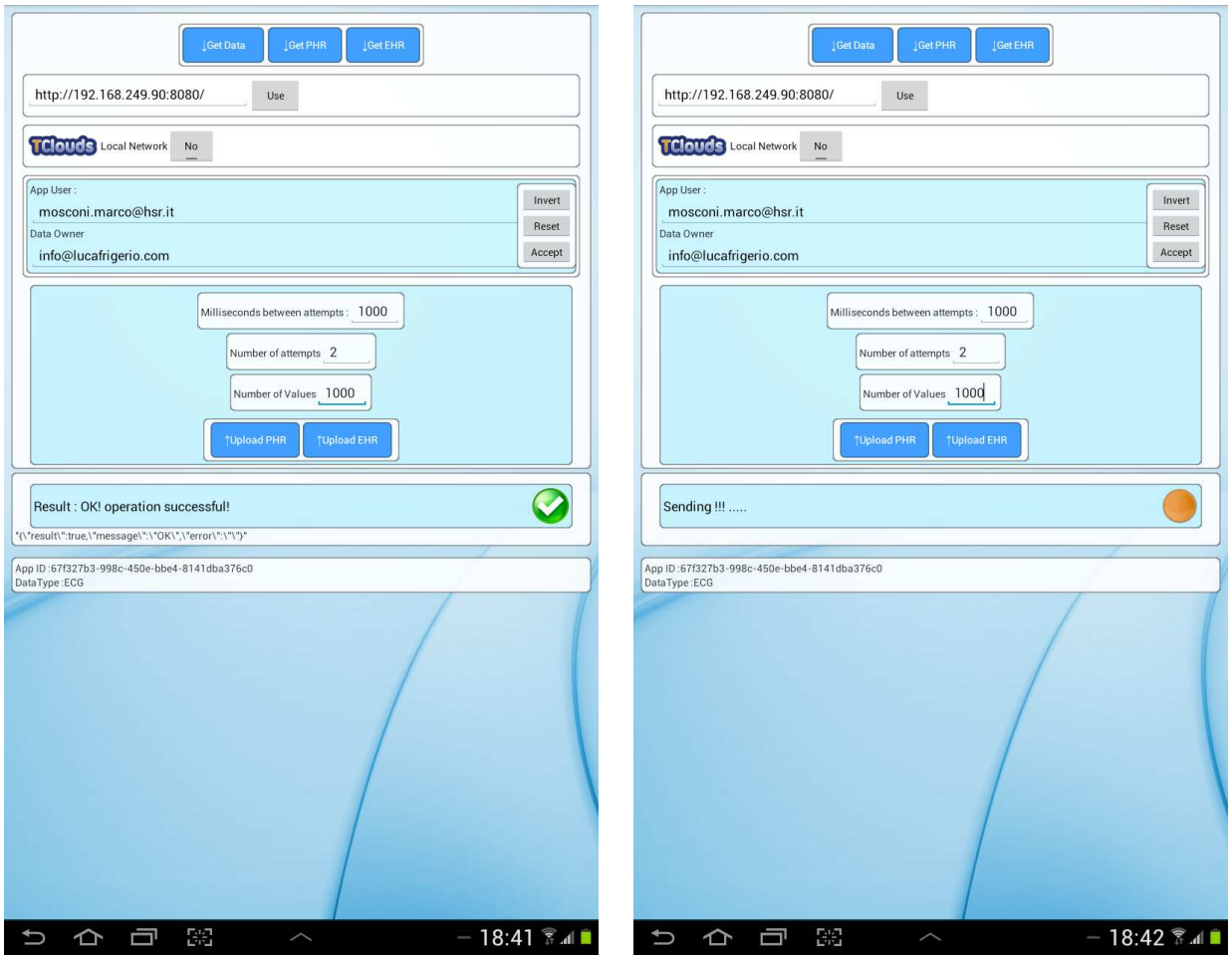


Figure 107 - Third party demo app for TPaaS (Idle state (on the left), Sending state (on the right))

We selected to store two times 1000 random PHR data of an user (info@luca.com). This allows having enough log entry to feed log session and populate the LogService database. We can see the populated LogService by accessing to the LogService Dashboard (Figure 108)

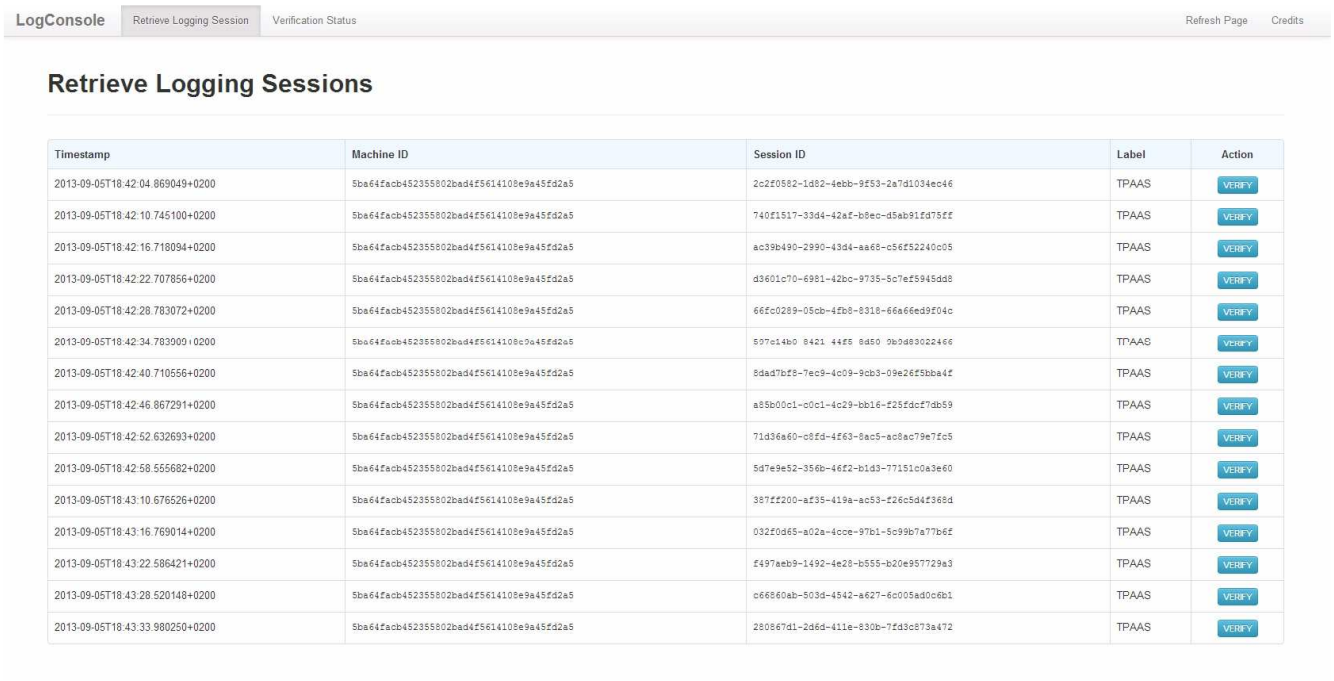


Figure 108 - Listing of log session into the logService

At this point, in which the log is still integer, we can perform a session verification. We chose session id: ac39b490-2990-43d4-aa68-c56f52240c05. The verification process leads to a successful verification.

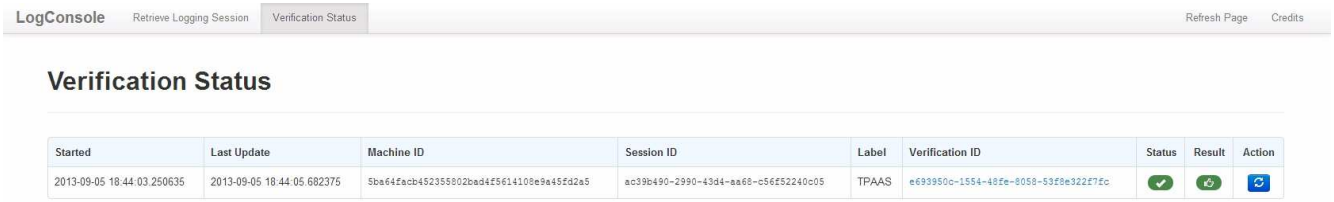


Figure 109 - Verification status of session id ac39b490-2990-43d4-aa68-c56f52240c05. Please note the Successful result

We can also see its details:

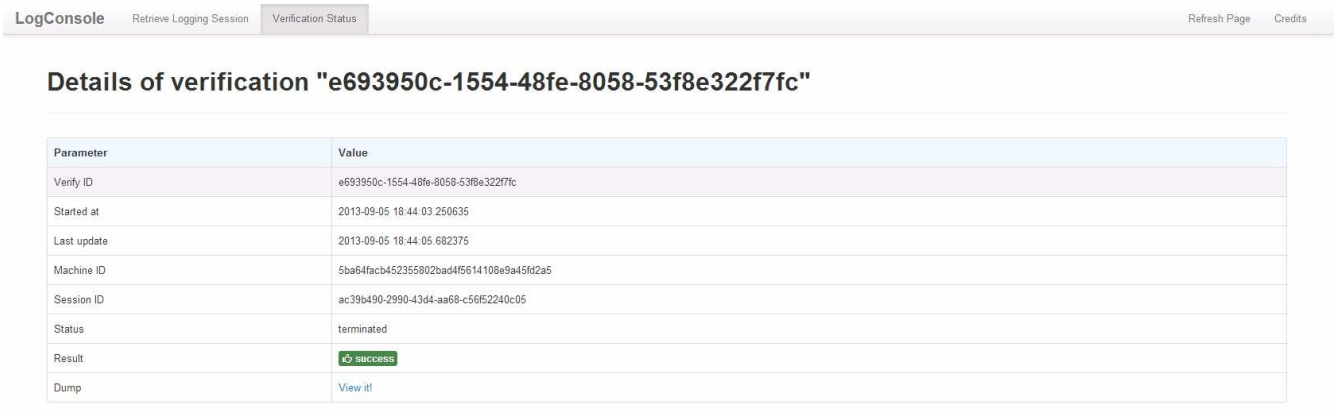


Figure 110 - details of session ac39b490-2990-43d4-aa68-c56f52240c05. Please note the successful result

3.1.5.4.1.1 Tampering the log database

Now is time to simulate an attack to the log file. This attack is performed by simply removing a line from the log file corresponding to the session that we have just verified.

By accessing as administrator into the LogService VM we can inspect the modification that has been done by using “diff” command over the original log file (that we previously saved) and the attacked log file:

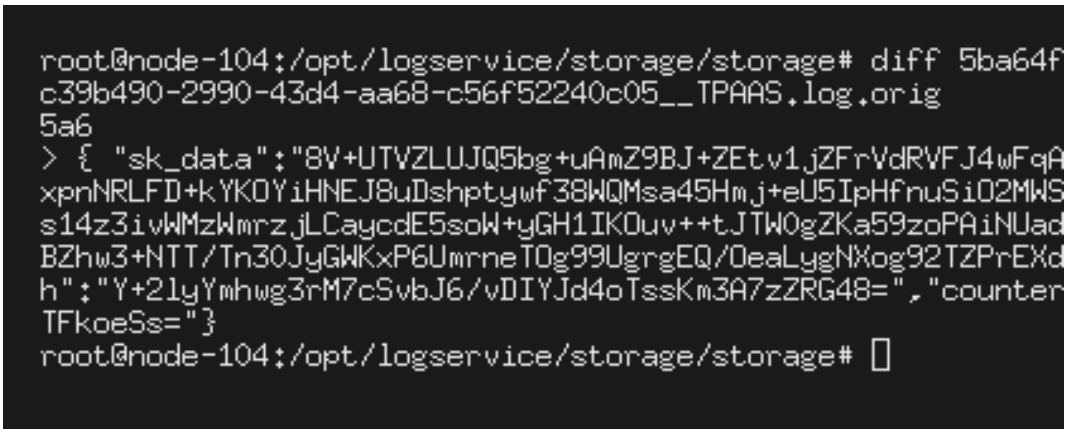


Figure 111 - Diff outcome between original log file and attached log file

Diff commands says that line 6 is the line that has the first difference in fact, by looking the two log files (the original and the attacked) we can see that line 6 is the one that is missing:

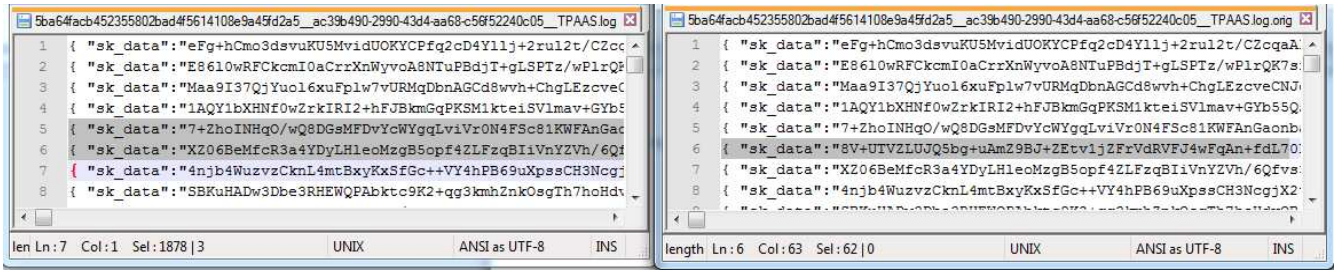


Figure 112 - Detail of Log file difference. Please note the line #6 that is missing on the attacked log file on the left

We can also see the number of the log entry that is missing:

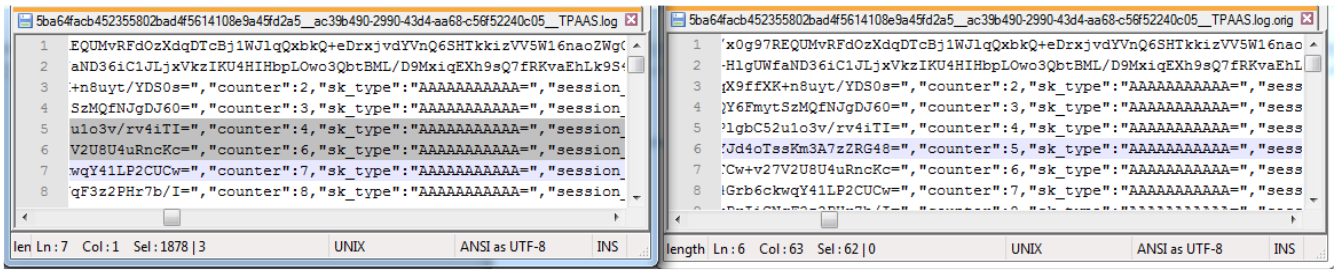


Figure 113 - Detail of log file difference. Please note the missing log entry with "counter"=5 in the attacked file on the left

Which results to be the log entry number 5.

At this point, if we access to the LogService dashboard we can perform again a verification of the same session, with a negative result:

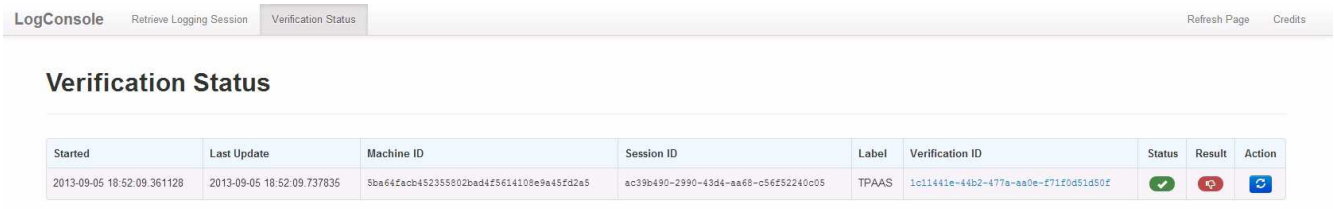


Figure 114 - Verification status of session ac39b490-2990-43d4-aa68-c56f52240c05. Please note the failure outcome

As expected the verification process has failed, these are the details:

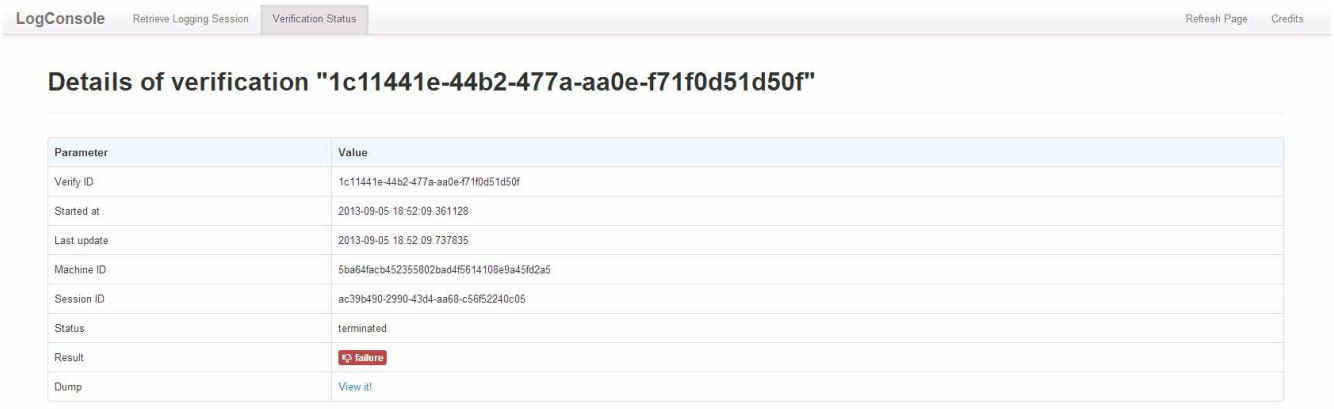


Figure 115 - detail of verification process for session ac39b490-2990-43d4-aa68-c56f52240c05. Please note the failure notification

And its dump shows all the log entries that are valid within the session just verified:

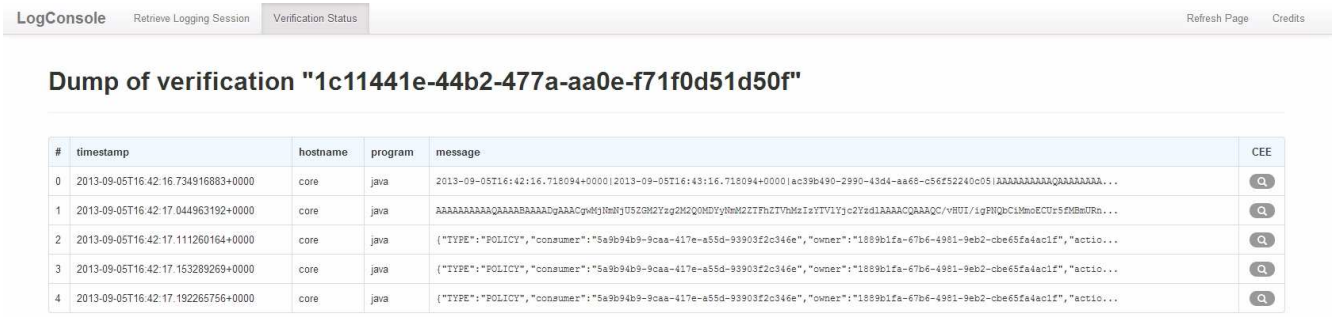


Figure 116 - Dump of the verification process. As expected only the first 4 log entries are shown since the following cannot be trusted anymore

The picture above shows, indeed, that line 4 it the very last valid line, after that the log is not any more reliable.

3.1.5.5 Conclusion

The nature of TClouds LogService, to be an “as-a-service” subcomponent, allows to use all the power of the LogService directly within the cloud customer VM (in this case the Healthcare Platform). This allows the cloud customer to leverage on the intrinsic trustworthiness of TClouds technology to assess any activity that has been done within the VM (of course, activity that has been properly logged). The Healthcare Platform can thus propose professional services and provide guarantees on the user activity of the Healthcare platform.

Requirements’ assessment

LREQ1 - Confidentiality of personal data – This subsystem allows to detect attackers intrusions that caused the leakage of users’ personal data.

LREQ3 - Control of location and responsible provider – This subsystem can be used to log transfers of data between different Cloud data-center sites.

LREQ4 – Un-linkability and Intervenability – This subsystem implements techniques to prevent logs be associated to a real user.

LREQ5 - Transparency for the customer – This subsystem can be used, together with the TPaaS platform, to reliably track operations performed by the Healthcare users on the platform and logs produced by users’ activities.

AHSECREQ6 - Non repudiation – This subsystem ensures that an attacker cannot deny to have performed a specific action by guaranteeing, integrity of logs and the proof that the application is trusted to properly record the actions.

AHSECREQ7 - Accountability – This subsystem ensures that an attacker cannot deny to have granted users’ privileges without permission by guaranteeing, integrity of logs and the proof that the third party application is trusted to properly perform these actions.

By conducting the LogService validation activity we have assessed its efficacy and proved its concepts. The LogService_1 validation activity has been SUCCESSFULLY PASSED.

3.1.6 Tailored Memcached Validation activity

In this chapter we show the execution of the validation activity of the Tailored Memcached subsystem.

This activity is not present in deliverable D3.3.3 since it has been decided afterwards to add Memcached features to the Healthcare Scenario. In M29, the Healthcare Platform has included a major update that makes use of ACaaS, which has enormously affected the shape of the platform infrastructure by requiring three geo-located remote databases in order to store the data in a legally compliant fashion.

This change has introduced a higher overhead in the system, calling for a caching solution. The Tailored Memcached subsystem from WP2.1 has been the perfect candidate.

The following are tables explain the planned validation activities in context of the Tailored Memcached subsystem:

Activity ID	**Memcached_1**
Activity type	Metrics
Activity description	The goal of this activity is to measure the attack surface of the Tailored Memcached platform in comparison with the original

	Memcached running on a Linux kernel. The Source Lines of Code (= Program code without comments, SLoC) provides a rough indicator for the complexity and thus probability of security-critical program errors. • Download Linux Kernel Source and Original Memcached • Unpack Sources and remove all non-relevant code (Test-cases, platform specific files not relevant for x86) • Use sloccount tool on source tree to estimate attack surface of Linux + Memcached platform • Collect HaLVM + HaNS + HsMemcached + required Libraries sources • Remove non-relevant code parts • Use sloccount on source tree to estimate attack surface of Tailored Memcached
Acceptance Criteria	The Tailored Memcached implementation requires less than half of the original code and the majority is written in a language ensuring memory safety.

Table 20 - Memcached_1 Validation Activity

Activity ID	**Memcached_2**
Activity type	Resource usage benchmark
Activity description	We compare the memory resources required by the Tailored Memcached and the original Memcached implementation on a Linux platform. Available RAM is one of the main driving factors for the price tag of a virtual machine in an IaaS cloud. With lower memory requirements, customers save money, consolidating machines becomes easier and cloud providers do not need to keep that much resources available. • Install and boot a small Linux installation • Install standard Memcached package • Measure memory usage for this small Linux installation at runtime • Compare above memory usage with usage of Tailored Memcached
Acceptance Criteria	The Tailored Memcached implementation requires less than 10% storage compared to the small Linux installation and the traditional Memcached.

Table 21 - Memcached_2 Validation Activity

3.1.6.1 Memcached features

Within a virtual machines (VMs) in a cloud environment, users typically install commodity operating systems like Linux or Windows and often an application server that hosts another virtual machine (e.g. Java's JVM) for the actual program the cloud user wants to run. This huge stack of software layers is a source of many security issues that can be exploited, sometimes even if the application does not depend on that vulnerable feature in particular. The Tailored Memcached approach tries to minimize the trusted code base for an individual

service by configuring the software stack within a VM to include only the absolutely necessary features.

3.1.6.1.1.1 Activity Memcached_1

This activity bases its validity on the observation that the number of bugs positively correlates with the number of code lines for any software project. By assessing that Trusted Memcached uses less code lines we are actually saying that the probability to contain critcal bugs is reduced, as smaller code bases are also less error prone.

3.1.6.1.2 Validation scenario

This validation activity does not have a proper scenario since the activity consists in using the *sloccount* tool to estimate the number of code line.

3.1.6.1.3 Validation setup

In order to properly estimate code lines, we have prepared “stripped” versions of the Tailored Memcached core and its dependencies and a “stripped” version of a Linux kernel with the standard Memcached implementation. The code has been stripped out from useless parts such as code examples, test cases and documentation. The stripped version of Tailored Memcached system consist of:

- HaLVM: The Haskell Lightweight Virtual Machine, or HaLVM, is a port of the Glasgow Haskell Compiler toolchain to enable developers to write high-level, lightweight virtual machines that can run directly on the Xen hypervisor. For validation purposes, only the runtime portions linked to the final executable were considered. Original version can be found here: https://github.com/GaloisInc/HaLVM
- HaNS: The HaLVM Network Stack. HaNS is a lightweight, pure Haskell network stack that can be used for TCP/IP networking in the context of HaLVM. Original version can be found here: https://github.com/GaloisInc/HaNS
- HsMemcached: The Tailored Memcached subsystem, developed by TUBS.

The standard Memcached system consist of:

- Linux kernel: Since for Memcached_2 validation activity the lightweight Bodhi Linux flavor has been chosen, we took its latest kernel version at the time of writing: v3.8.0-12. Source code can be obtained from https://launchpad.net/ubuntu/+archive/primary/+files/linux_3.8.0-12.21.tar.gz
- Standard Memcached: Source code is available at: https://code.google.com/p/memcached/wiki/DevelopmentRepos

3.1.6.1.4 Validation execution

Starting with Tailored Memcached solution, in the following listings we can see the `sloccount` values of HaLVM, HaNS and HsMemcached:

```
hsr@ubuntu:~/Tailored_memcached$ sloccount HaLVM-stripped
Have a non-directory at the top, so creating directory top_dir
Adding /home/hsr/Tailored_memcached/HaLVM-stripped/LICENSE to top_dir
Adding /home/hsr/Tailored_memcached/HaLVM-stripped/README to top_dir
Creating filelist for cmm
Creating filelist for ghc-xen-sparse
Creating filelist for libIVC
Creating filelist for libm
Creating filelist for libraries
Creating filelist for rts
Creating filelist for xen-ghc
Categorizing files.
Finding a working MD5 command....
Found a working MD5 command.
Computing results.


SLOC    Directory       SLOC-by-Language (Sorted)
59414   xen-ghc         haskell=48078,ansic=8031,sh=3305
41500   rts             ansic=41375,asm=125
30967   libraries       haskell=29498,ansic=1464,sh=5
8144    libm            ansic=7233,asm=911
7881    ghc-xen-sparse  ansic=7226,asm=655
3042    cmm             ansic=3042
554     libIVC          ansic=554
0       top_dir         (none)


Totals grouped by language (dominant language first):
haskell:      77576 (51.20%)
ansic:        68925 (45.49%)
sh:            3310 (2.18%)
asm:           1691 (1.12%)




Total Physical Source Lines of Code (SLOC)                = 151,502
Development Effort Estimate, Person-Years (Person-Months) = 38.95 (467.36)
 (Basic COCOMO model, Person-Months = 2.4 * (KSLOC**1.05))
Schedule Estimate, Years (Months)                         = 2.15 (25.85)
 (Basic COCOMO model, Months = 2.5 * (person-months**0.38))
Estimated Average Number of Developers (Effort/Schedule)  = 18.08
Total Estimated Cost to Develop                           = $ 5,261,153
 (average salary = $56,286/year, overhead = 2.40).
SLOCCount, Copyright (C) 2001-2004 David A. Wheeler
SLOCCount is Open Source Software/Free Software, licensed under the GNU GPL.
SLOCCount comes with ABSOLUTELY NO WARRANTY, and you are welcome to
redistribute it under certain conditions as specified by the GNU GPL license;
see the documentation for details.
Please credit this data as "generated using David A. Wheeler's 'SLOCCount'."
hsr@ubuntu:~/Tailored_memcached$
```

Figure 117 - sloccount outcome for HaLVM, part of tailored memcached

```
hsr@ubuntu:~/Tailored_memcached$ sloccount HaNS-stripped
Have a non-directory at the top, so creating directory top_dir
Adding /home/hsr/Tailored_memcached/HaNS-stripped/Setup.hs to top_dir
Creating filelist for cbits
Adding /home/hsr/Tailored_memcached/HaNS-stripped/hans.cabal to top_dir
Creating filelist for src_Data
Creating filelist for src_Hans
Categorizing files.
Finding a working MD5 command....
Found a working MD5 command.
Computing results.


SLOC    Directory       SLOC-by-Language (Sorted)
5530    src_Hans        haskell=5530
169     src_Data        haskell=169
103     cbits           ansic=103
4       top_dir         haskell=4


Totals grouped by language (dominant language first):
haskell:       5703 (98.23%)
ansic:          103 (1.77%)



Total Physical Source Lines of Code (SLOC)                = 5,806
Development Effort Estimate, Person-Years (Person-Months) = 1.27 (15.22)
 (Basic COCOMO model, Person-Months = 2.4 * (KSLOC**1.05))
Schedule Estimate, Years (Months)                         = 0.59 (7.03)
 (Basic COCOMO model, Months = 2.5 * (person-months**0.38))
Estimated Average Number of Developers (Effort/Schedule)  = 2.16
Total Estimated Cost to Develop                           = $ 171,282
 (average salary = $56,286/year, overhead = 2.40).
SLOCCount, Copyright (C) 2001-2004 David A. Wheeler
SLOCCount is Open Source Software/Free Software, licensed under the GNU GPL.
SLOCCount comes with ABSOLUTELY NO WARRANTY, and you are welcome to
redistribute it under certain conditions as specified by the GNU GPL license;
see the documentation for details.
Please credit this data as "generated using David A. Wheeler's 'SLOCCount'."
hsr@ubuntu:~/Tailored_memcached$
```

Figure 118 - sloccount outcome for HaNS, part of tailored memcached

```
hsr@ubuntu:~/Tailored_memcached$ sloccount hsMemcached
Creating filelist for hsMemcached
Categorizing files.
Finding a working MD5 command....
Found a working MD5 command.
Computing results.


SLOC    Directory       SLOC-by-Language (Sorted)
2449    hsMemcached     haskell=2449


Totals grouped by language (dominant language first):
haskell:       2449 (100.00%)



Total Physical Source Lines of Code (SLOC)                = 2,449
Development Effort Estimate, Person-Years (Person-Months) = 0.51 (6.15)
 (Basic COCOMO model, Person-Months = 2.4 * (KSLOC**1.05))
Schedule Estimate, Years (Months)                         = 0.42 (4.98)
 (Basic COCOMO model, Months = 2.5 * (person-months**0.38))
Estimated Average Number of Developers (Effort/Schedule)  = 1.23
Total Estimated Cost to Develop                           = $ 69,196
 (average salary = $56,286/year, overhead = 2.40).
SLOCCount, Copyright (C) 2001-2004 David A. Wheeler
SLOCCount is Open Source Software/Free Software, licensed under the GNU GPL.
SLOCCount comes with ABSOLUTELY NO WARRANTY, and you are welcome to
redistribute it under certain conditions as specified by the GNU GPL license;
see the documentation for details.
Please credit this data as "generated using David A. Wheeler's 'SLOCCount'."
hsr@ubuntu:~/Tailored_memcached$
```

Figure 119 - sloccount outcome for HsMemcached, part of Tailored memcached

The same tool has been used to estimate the code lines of standard Memcached solution (Linux kernel + standard Memcached source code).

```
Creating filelist for virt
Categorizing files.
Finding a working MD5 command....
Found a working MD5 command.
Computing results.


SLOC    Directory       SLOC-by-Language (Sorted)
5615064 drivers         ansic=5610304,yacc=1688,asm=1475,perl=792,lex=779,
                        sh=26
698974  fs              ansic=698974
533134  sound           ansic=532951,asm=183
493711  net             ansic=493615,awk=96
301728  include         ansic=299977,cpp=1709,asm=42
175482  arch            ansic=157919,asm=17129,awk=374,sh=38,perl=22
120454  kernel          ansic=120149,perl=305
56177   tools           ansic=51029,perl=3272,python=1399,sh=476,asm=1
54529   mm              ansic=54529
44171   security        ansic=44171
42627   crypto          ansic=42627
37307   scripts         ansic=22487,perl=8287,sh=2028,cpp=1820,yacc=1291,
                        lex=947,python=447
28486   lib             ansic=28473,awk=13
14382   block           ansic=14382
5705    ipc             ansic=5705
4661    virt            ansic=4661
2377    init            ansic=2377
1876    firmware        asm=1660,ansic=216
564     usr             ansic=550,asm=14
0       top_dir         (none)


Totals grouped by language (dominant language first):
ansic:      8185096 (99.44%)
asm:          20504 (0.25%)
perl:         12678 (0.15%)
cpp:           3529 (0.04%)
yacc:          2979 (0.04%)
sh:            2568 (0.03%)
python:        1846 (0.02%)
lex:           1726 (0.02%)
awk:            483 (0.01%)




Total Physical Source Lines of Code (SLOC)                = 8,231,409
Development Effort Estimate, Person-Years (Person-Months) = 2,583.91 (31,006.96)
 (Basic COCOMO model, Person-Months = 2.4 * (KSLOC**1.05))
Schedule Estimate, Years (Months)                         = 10.61 (127.26)
 (Basic COCOMO model, Months = 2.5 * (person-months**0.38))
Estimated Average Number of Developers (Effort/Schedule)  = 243.65
Total Estimated Cost to Develop                           = $ 349,051,504
 (average salary = $56,286/year, overhead = 2.40).
SLOCCount, Copyright (C) 2001-2004 David A. Wheeler
SLOCCount is Open Source Software/Free Software, licensed under the GNU GPL.
SLOCCount comes with ABSOLUTELY NO WARRANTY, and you are welcome to
redistribute it under certain conditions as specified by the GNU GPL license;
see the documentation for details.
Please credit this data as "generated using David A. Wheeler's 'SLOCCount'."
hsr@ubuntu:~/Tailored_memcached$
```

Figure 120 - sloccount outcome of Linux Kernel, part of standard memcached

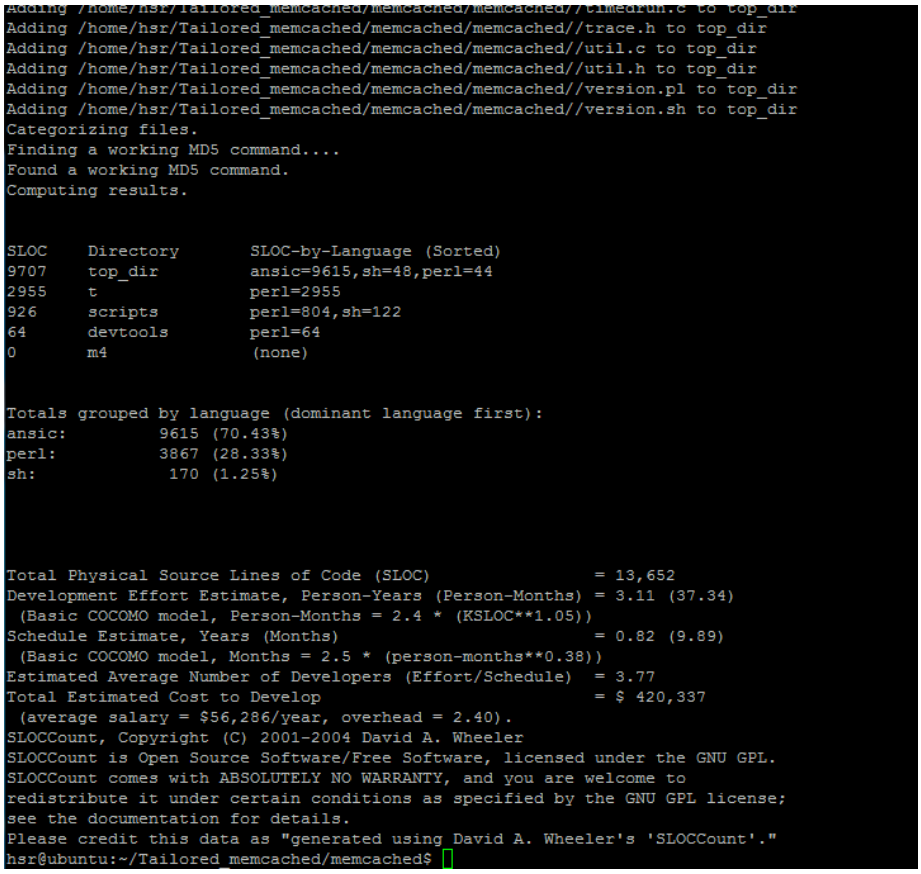


Figure 121- sloccount outcome of Memcached sources, part of standard memcached

The following summarizes all the code line count, divided by programming language

Tailored Memcached									**Std Memcached**					
HaLVM			**HaNS**			**HSMemcached**			**Linux Kernel (sound removed)**			**Std. Memcached**		
Haskell	77576	51.2 %	haskell	5703	98.23 %	haskell	2449	100 %	Ansic	7,65M	99%	ansic	9615	70.43 %
ansic	68925	45.49 %	ansic	103	1.77 %				Asm	20k	0.2 %	Perl	3867	27.33 %
sh	3310	2.18 %							Perl	12678	0.15 %	sh	170	1.25 %
asm	1691	1.12 %							Cpp	3529	0.04 %			
									Yacc	2979	0.04 %			
									Sh	2568	0.03 %			
									Python	1846	0.02 %			
									Lex	1726	0.02 %			
									Awk	483	0.01 %			
summary									**summary**					
Haskell			85728											
Ansic			69028						Ansic		7.65M			
Sh			3310						Sh		2738			

asm	1691		Asm	20k	
			Cpp	3529	
			Yacc	2979	
			Python	1846	
			Lex	1726	
			Awk	483	
			Perl	16545	
TOTAL	**159757**		**TOTAL**	**7699846**	

Table 22 - summarizing table between standard and tailored memecached

As we can see from the table above, the Tailored Memcached subsystem uses far less code to make the system running (around 2% of the code of the reference Memcached solution), and it makes heavy use of Haskell programming language, which has a purely functional programming paradigm. This means that functions may not have arbitrary side effects in functions and believed to result in less error prone code. This slim code usage of Tailored Memcached solution gives, assuming a similar fault rate, 98% less probability to have bugs, thus increases the overall TClouds security capabilities[2].

3.1.6.1.4.1 Activity Memcached_2

This validation activity aims to prove Tailored Memcached's reduced memory requirements. The pay-per-use model on public IaaS clouds basically means that the more resource the customer uses, the more the money he has to pay. A tailored service should be able to reduce the required overhead for the runtime system and thus make it more cost effective.

3.1.6.1.5 Validation scenario

The benchmark scenario with the two machines used for this validation activity is depicted below.

[2] This assumes that is the same programmer (or a group of programmers with a similar skill level) wrote the two code bases. Please refer to the conclusion chapter for a more punctual comparison.

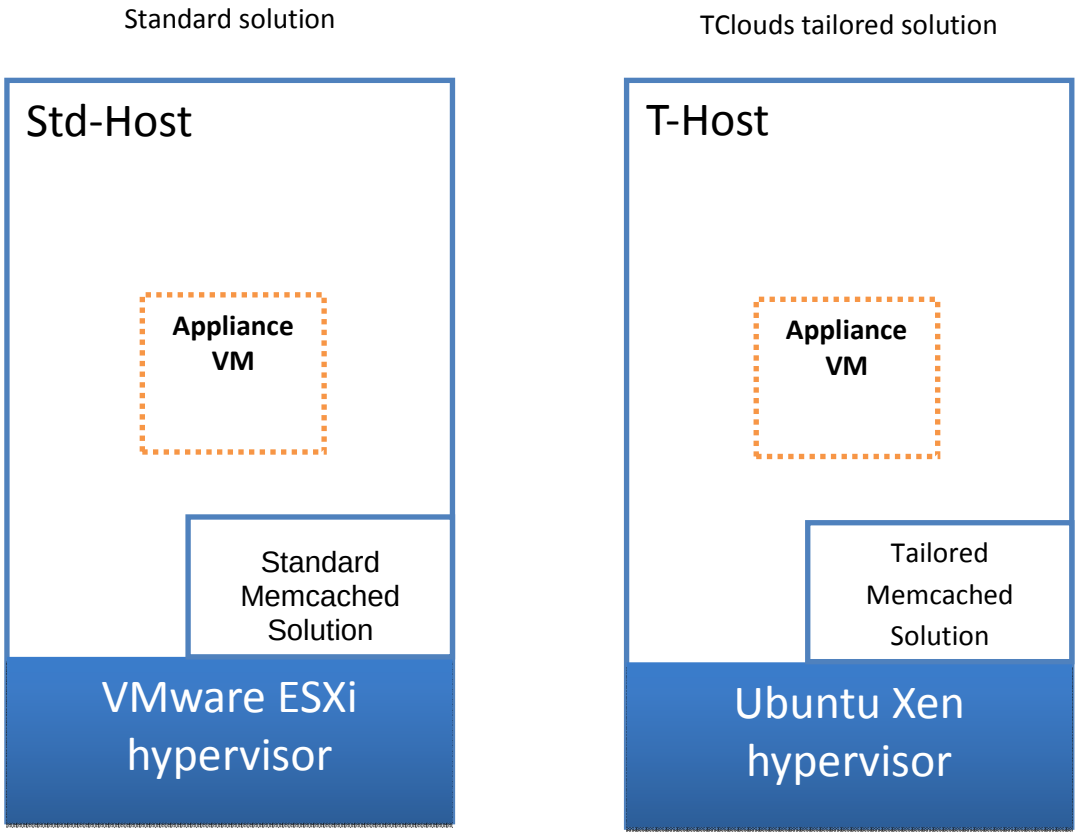


Figure 122 - Memcached validation scenario

In preparation for this scenario we have used a TClouds-like host environment (a Xen hypervisor) and placed the Tailored Memcached VM on top. For comparison with a classic cloud infrastructure, we have also set up a VMware ESXi host, in which we placed a minimalistic, Debian-based Linux with the standard Memcached program installed.

Standard solution setup:

- Host: VMware ESXi server
- Standard Memcached VM:
  - Turnkey Linux (http://www.turnkeylnux.org/)
  - 512 MB RAM
  - 1 virtual CPU
  - No virtual hard disk
  - Standard Memcached: `apt-get install memcached`

Tailored Memcached setup:

- Host: Ubuntu Server with Xen hypervisor
- Tailored Memcached VM:
- HaLVM + HaNS
- HsMemcached
- 512 MB RAM
- 1 virtual CPU
- No virtual hard disk

In order to have comparable results, we decided to use an optimized-Linux distribution (in our scenario: Turnkey Linux). TurnkeyLinux core OS is a Debian based linux distribution that

has been highly optimized in order to work efficiently in servers environment. From it all the unnecessary packages have been removed in order to maintain memory consumption as lower as possible.

3.1.6.1.6 Validation execution

We first deployed the Tailored Memcached on top of the Xen hypervisor. Then we queried the Xen hypervisor for the idle resource consumption of the Tailored Memcached VM. A screenshot of the `xl top` output is depicted below:

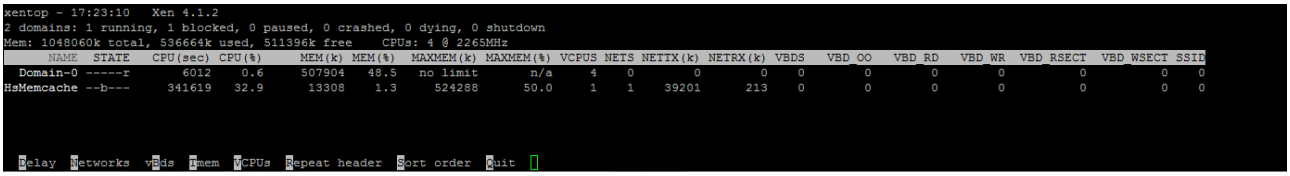


Figure 123 - Memcached memory consumption

We also performed a longer resource monitoring (around 2 minutes) and this is the result:

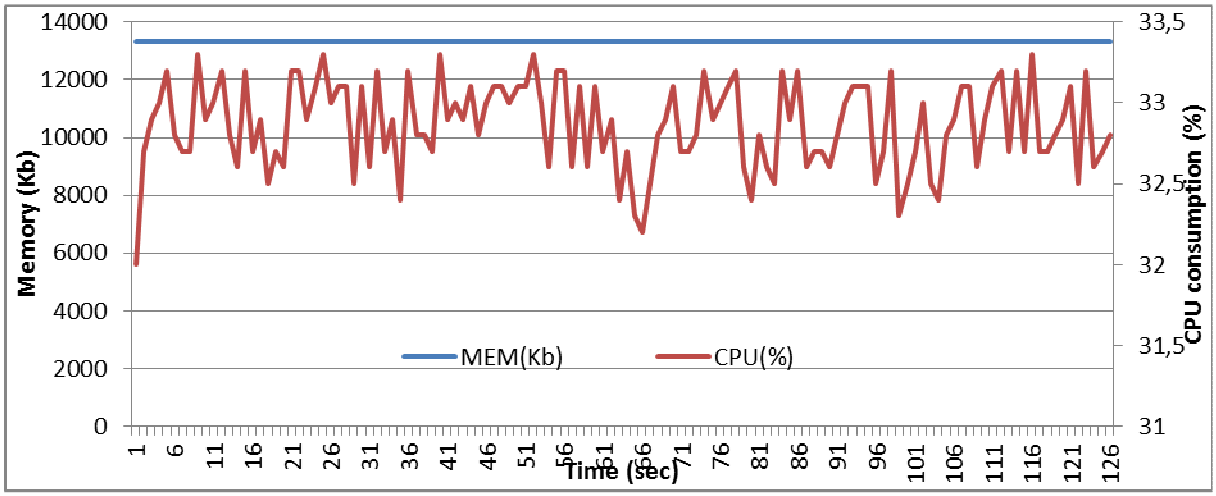


Figure 124 - Resource consumption for Tailored Memcached

Memory consumption of HsMemcached, in idle period, is stuck at around 13 MB of memory. Consider that the system is running the whole Operating System with Tailored Memcached active.

We performed the read at idle time since the extra workload is imputable to the client appliance usage.

For the standard Memcached solution, we deployed its VM into a classic host. We have chosen VMware ESXi as a virtualization layer.

We have conducted a measurement of memory resource usage over a longer time period (around 2 minutes) and this is the result:

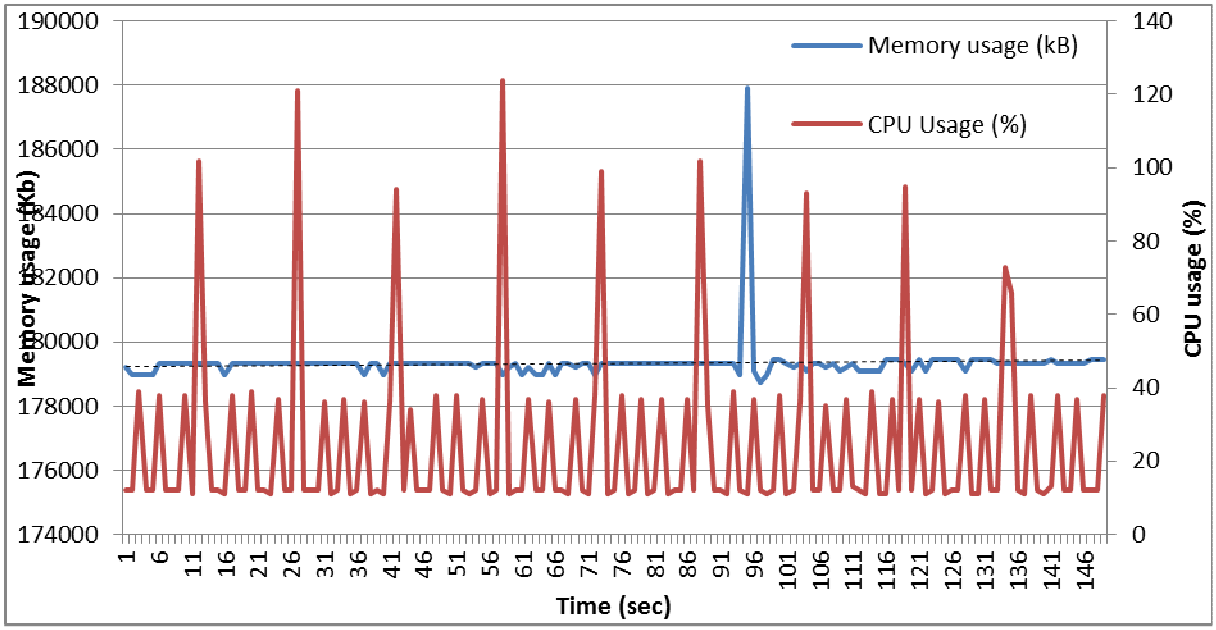


Figure 125 - resource consumption for Standard memcached

Here we can see the memory resource usage is much higher than in the Tailored Memcached. A standard Linux installation requires more than 10 times the memory of TClouds' Tailored Memcached (almost 180 MB instead of 13 MB). The dashed line indicates the linear average memory usage.

3.1.6.1.7 Conclusion

Having Memcached capabilities directly managed by the cloud infrastructure has many advantages either from the cloud owner perspective (that can dedicate specific resources for Memcached VMs) and for the cloud customer (that can leverage Memcached features to the cloud, reduce its VM complexity and reduce the potential security holes within the customer VM).

Watching Memcached_1 validation activity, the TClouds Tailored Memcached subsystem performs better compared with a standard installation of Memcached.

Even though the assumption, that a function with less line of code has lower probability to contain bugs might be controversial, we can assess that the usage only of 2% of code lines and the involvement only of the necessary code (in contrast with a normal Linux kernel that includes the whole stack of components) may reduce significantly the probability for critical bugs.

Requirements' assessment

AHSECREQ3 - Integrity of the application – By developing the subsystem in a type-safe language even at the operating system level and minimizing the running code base, the entire service becomes very hard to attack. This increases confidence in the integrity of the system.

ASSECREQ6 - High performance & Scalable – As the amount of program code is reduced and the operating system becomes part of the application, many time consuming intermediate steps can be skipped. Also taking special considerations for cloud computing environments into account, this should result in better performance than traditional software stacks can achieve.

ASSECREQ2 - Trustworthy Infrastructure – The improved type-safety and compile-time checks reduce the attack surface of the memcached storage. This prevents intrusions into cloud infrastructures providing a key/value service using our implementation, thus improving the trustworthiness of the infrastructure.

We can conclude that Memcached_1 Validation activity has successfully 100% PASSED.

In regard to the Memcached_2 validation activity, we have obtained a similar result. Tailored Memcached has outperformed a standard installation in the sense that it uses less memory resources (around 76% less resources compared with a standard installation). However, we noticed a higher CPU consumption of Tailored Memcached (around 30% of the resources assigned to the VM).

Considering all the results obtained in the validation activity, the Tailored Memcached subsystem has also PASSED the Memcached_2 validation activity 100%.

3.1.7 SAVE validation activity

As described in D3.3.3 SAVE subcomponent was not going to be used by the Healthcare Scenario. However, with some extra effort of integration among the infrastructure and the platform we managed to properly use SAVE. The validation activity is described below

Activity ID	**SAVE_1**
Activity type	Proof of concept test
Activity description	1- Complementary for the ACaaS_1 activity: Automated inspection of the VM deployment to verify that ERH_IT and Appliance is running on Host 1_IT and that ERH_DE and PHR is running on Host 2_DE. 2- Deploy a VM in the wrong host and perform SAVE validation
Acceptance Criteria	SAVE needs to successfully verify the deployment of the VMs on the desired hosts in the specific geo location. Deployment of a VM on the wrong host needs to be detected, as it constitutes a policy violation.
Reference Documents:	D2.3.1, Cha. 8 D2.3.2, Cha. 3 and 4

Table 23- SAVE_1 validation activity description

3.1.7.1 SAVE features

The aim of this work is to automate information-flow analysis for large-scale heterogeneous

virtualized infrastructures. We aim at reducing the analysis complexity for human administrators to the specification of a few well-designed trust assumptions and leave the extrapolation of these assumptions and analysis of information flow behavior to the tools.

We propose an information flow analysis tool for virtualized infrastructures. The tool is capable of discovering and unifying the actual configurations of different virtualization systems (Xen, VMware, KVM, and IBM's PowerVM) and running a static information flow analysis based on explicitly specified trust rules. Our analysis tool models virtualized

infrastructures faithfully, independent of their vendor, and is efficient in terms of absence of false negatives as well as adjustable false positive rates.

When you run SAVE it pass through 4 main steps:

- **Discovery**: The goal of the discovery phase is to retrieve sufficient information about the configuration of the virtualized infrastructure
- **Transformation into a Graph Model**: we translate the discovered platform-specific configuration into a unified graph representation of the virtualization infrastructure, the Realization Model.
- **Coloring through Graph traversal**: the graph traversal phase obtains a realization model and a set of information source vertices with their designated colors as input
- **The traversal rules**: the graph coloring algorithm requires a set of traversal rules that model information flows, isolation properties, and trust assumptions
- **Detecting undesired information flows:** The goal of the description phase is to produce meaningful outputs for system administrators. For detecting undesired information flows, save colors a set of information sources that mark types of critical information that must not leak.

3.1.7.2 Validation scenario

SAVE program is a vast and complex system able to perform many activities, it also has UI features. However, for our validation scenario we decided to stop SAVE process at the “discovery” phase. This will allow us to have the overview of the healthcare VMs deployment location. To have an overview of the deployment scenario of the VMs, please refer to Figure 48.

3.1.7.3 Validation setup

SAVE is a client-side component, we need to connect via ssh directly to the cloud hosts in order to perform commands that SAVE uses for the discovery phase. We decided to connect via ssh by using private/public key pairing. Thus, we just added the healthcare public key into the TClouds hosts.

SAVE has some config files that needs to be configured as well.

- Save.ini - consists in the hosts addresses and ssh configuration
- PolicyDeploymentExists.gpr – consists in the graph of the nodes that SAVE discovery process should find into TClouds and assess against it.

The positive case we want to address is the original deployment scenario (EHR_IT and Appliance into the Italian Host, PHE and EHR_DE into the German host). It can be schematized by the picture below:

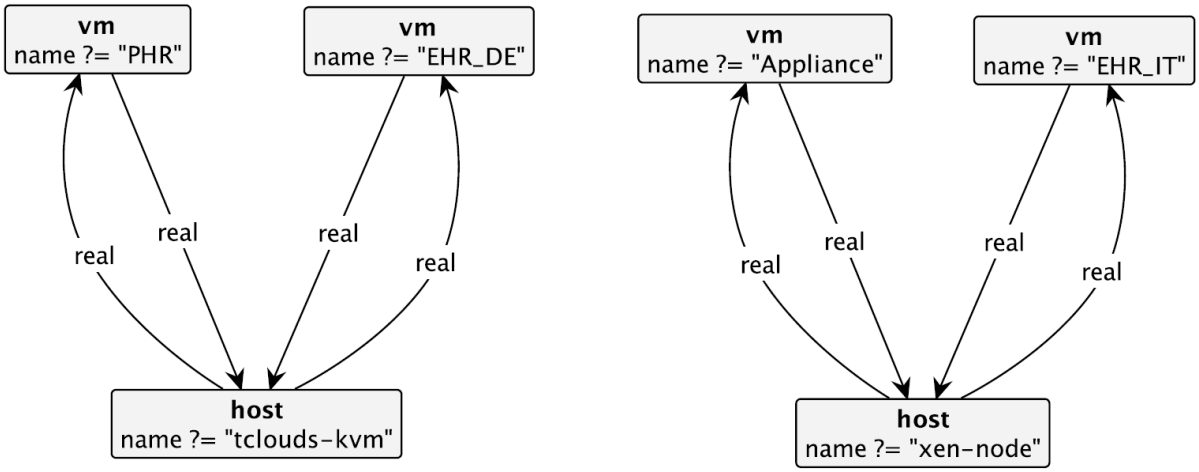


Figure 126 - SAVE validation schema. This schema has been generated by inspecting "PolicyDeploymentExists.gpr" file

3.1.7.4 Validation execution

The validation is straightforward. It is necessary start SAVE client jar in order to perform the discovery:

```
java -jar /root/SAVE/save-cmd-0.1.jar
```

We started with the original deployment scenario, EHR_IT and Appliance into the Italian Host, PHE and EHR_DE into the German host, with all the machines running, this is the outcome:

```
root@core ~/SAVE# java -jar save-cmd-0.1.jar
Hello, Save Cmd
19:06:15.111 INFO  com.ibm.zurich.save.cmd.CmdUtil$     - username not set, using default (empty)
19:06:15.114 INFO  com.ibm.zurich.save.cmd.CmdUtil$     - password not set, using default (empty)
marco@134.147.62.162's password:
19:06:20.578 INFO  com.ibm.zurich.save.cmd.CmdUtil$     - username not set, using default (empty)
19:06:20.579 INFO  com.ibm.zurich.save.cmd.CmdUtil$     - password not set, using default (empty)
marco@134.147.217.66's password:
19:06:24.348 INFO  com.ibm.zurich.save.cmd.Cmd$         - Probes: Extract data...
19:06:25.325 INFO  com.ibm.zurich.save.actor.ProbeActor - extract finished
19:06:26.125 INFO  c.i.z.save.actor.LibvirtTransActor   - translation done
19:06:26.125 INFO  c.i.zurich.save.model.real.RealModel - merging models...
19:06:35.347 INFO  com.ibm.zurich.save.actor.ProbeActor - extract finished
19:06:35.348 INFO  com.ibm.zurich.save.cmd.Cmd$         - Probes finished
19:06:43.184 INFO  c.i.z.save.actor.LibvirtTransActor   - translation done
19:06:43.184 INFO  c.i.zurich.save.model.real.RealModel - merging models...
19:06:43.187 INFO  c.i.z.s.p.openstack.OpenstackProbe   - extract
19:06:46.673 INFO  com.ibm.zurich.save.actor.ProbeActor - extract finished
19:06:46.681 INFO  c.i.zurich.save.model.real.RealModel - merging models...
19:06:46.681 INFO  c.i.z.save.actor.OpenstackTransActor - translation done
19:06:46.682 INFO  com.ibm.zurich.save.cmd.Cmd$         - Translations finished
19:06:46.682 INFO  com.ibm.zurich.save.cmd.Cmd$         - Going into differential mode for model
19:06:47.224 INFO  c.i.z.save.analysis.GrooveAnalysis   - Converting RealModel to Groove graph..
19:06:47.269 INFO  c.i.z.save.analysis.GrooveAnalysis   - Running Groove exploration..
19:06:47.857 INFO  c.i.z.save.analysis.GrooveAnalysis   - Groove analysis done!
All SAVE!
19:06:47.888 INFO  com.ibm.zurich.save.actor.ProbeActor - disconnecting probe
19:06:47.888 INFO  com.ibm.zurich.save.actor.ProbeActor - disconnecting probe
19:06:47.888 INFO  com.ibm.zurich.save.actor.ProbeActor - disconnecting probe
good bye
root@core ~/SAVE#
```

Figure 127 - SAVE result with correct deployment

As expected the outcome has been successful.

Now we removed one VM (we chose the PHR one). And we run again SAVE validation this is the new deployment:

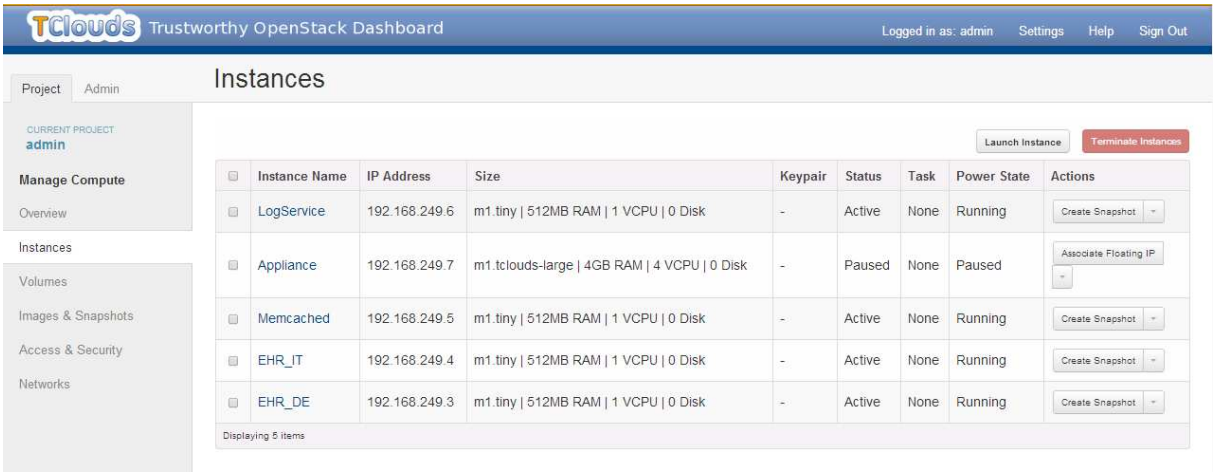


Figure 128 - wrong deployment scenario, PHR has been deleted

And this is SAVE output:

```
root@core ~/SAVE# java -jar save-cmd-0.1.jar
Hello, Save Cmd
19:11:54.159 INFO  com.ibm.zurich.save.cmd.CmdUtil$     - username not set, using default (empty)
19:11:54.163 INFO  com.ibm.zurich.save.cmd.CmdUtil$     - password not set, using default (empty)
marco@134.147.62.162's password:
19:11:58.636 INFO  com.ibm.zurich.save.cmd.CmdUtil$     - username not set, using default (empty)
19:11:58.636 INFO  com.ibm.zurich.save.cmd.CmdUtil$     - password not set, using default (empty)
marco@134.147.217.66's password:
19:12:02.102 INFO  com.ibm.zurich.save.cmd.Cmd$         - Probes: Extract data...
19:12:02.876 INFO  com.ibm.zurich.save.actor.ProbeActor - extract finished
19:12:03.083 INFO  com.ibm.zurich.save.actor.ProbeActor - extract finished
19:12:03.086 INFO  com.ibm.zurich.save.cmd.Cmd$         - Probes finished
19:12:03.184 INFO  c.i.z.save.actor.LibvirtTransActor   - translation done
19:12:03.184 INFO  c.i.zurich.save.model.real.RealModel - merging models...
19:12:03.836 INFO  c.i.z.save.actor.LibvirtTransActor   - translation done
19:12:03.836 INFO  c.i.zurich.save.model.real.RealModel - merging models...
19:12:03.840 INFO  c.i.z.s.p.openstack.OpenstackProbe   - extract
19:12:07.189 INFO  com.ibm.zurich.save.actor.ProbeActor - extract finished
19:12:07.196 INFO  c.i.zurich.save.model.real.RealModel - merging models...
19:12:07.196 INFO  c.i.z.save.actor.OpenstackTransActor - translation done
19:12:07.197 INFO  com.ibm.zurich.save.cmd.Cmd$         - Translations finished
19:12:07.197 INFO  com.ibm.zurich.save.cmd.Cmd$         - Going into differential mode for model
19:12:07.816 INFO  c.i.z.save.analysis.GrooveAnalysis   - Converting RealModel to Groove graph..
19:12:07.870 INFO  c.i.z.save.analysis.GrooveAnalysis   - Running Groove exploration..
19:12:08.390 INFO  c.i.z.save.analysis.GrooveAnalysis   - Groove analysis done!
Policy Violation of 'PolicyDeploymentExists'
19:12:08.421 INFO  com.ibm.zurich.save.actor.ProbeActor - disconnecting probe
19:12:08.421 INFO  com.ibm.zurich.save.actor.ProbeActor - disconnecting probe
19:12:08.421 INFO  com.ibm.zurich.save.actor.ProbeActor - disconnecting probe
good bye
root@core ~/SAVE#
```

Figure 129 - SAVE validation against deployment without PHR

Now we re-deployed the PHR VM but in the wrong host (the Italian one). In order to do so, we placed the wrong location requirements that AcaaS uses to deploy the VM. To show the deployment location we can refer to the following picture:

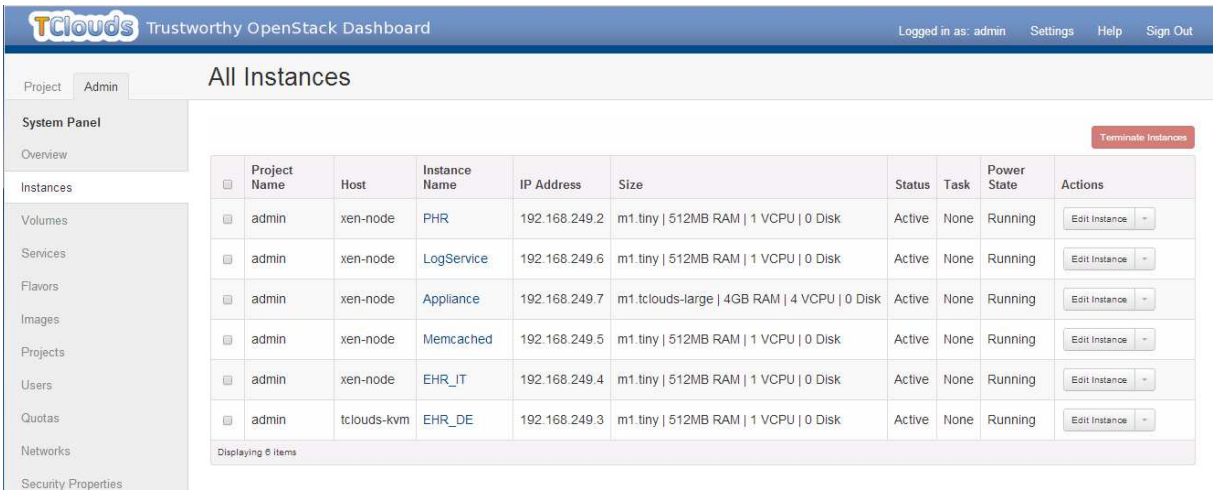


Figure 130 - Wrong Deployment. Please not that this view comes from the admin tab.

From the picture above we evince that the deployment's location of PHR VM is the wrong one (Xen-node corresponds to the Italian node, while tclouds-kvm is the German node).

Also this time SAVE output fails as expected:

```
root@core ~/SAVE# java -jar save-cmd-0.1.jar
Hello, Save Cmd
19:25:02.204 INFO  com.ibm.zurich.save.cmd.CmdUtil$     - username not set, using default (empty)
19:25:02.207 INFO  com.ibm.zurich.save.cmd.CmdUtil$     - password not set, using default (empty)
marco@134.147.62.162's password:
19:25:09.597 INFO  com.ibm.zurich.save.cmd.CmdUtil$     - username not set, using default (empty)
19:25:09.598 INFO  com.ibm.zurich.save.cmd.CmdUtil$     - password not set, using default (empty)
marco@134.147.217.66's password:
19:25:13.390 INFO  com.ibm.zurich.save.cmd.Cmd$         - Probes: Extract data...
19:25:14.426 INFO  com.ibm.zurich.save.actor.ProbeActor - extract finished
19:25:15.359 INFO  c.i.z.save.actor.LibvirtTransActor   - translation done
19:25:15.359 INFO  c.i.zurich.save.model.real.RealModel - merging models...
19:25:21.800 INFO  com.ibm.zurich.save.actor.ProbeActor - extract finished
19:25:21.801 INFO  com.ibm.zurich.save.cmd.Cmd$         - Probes finished
19:25:25.700 INFO  c.i.z.save.actor.LibvirtTransActor   - translation done
19:25:25.700 INFO  c.i.zurich.save.model.real.RealModel - merging models...
19:25:25.703 INFO  c.i.z.s.p.openstack.OpenstackProbe   - extract
19:25:29.013 INFO  com.ibm.zurich.save.actor.ProbeActor - extract finished
19:25:29.024 INFO  c.i.zurich.save.model.real.RealModel - merging models...
19:25:29.024 INFO  c.i.z.save.actor.OpenstackTransActor - translation done
19:25:29.025 INFO  com.ibm.zurich.save.cmd.Cmd$         - Translations finished
19:25:29.025 INFO  com.ibm.zurich.save.cmd.Cmd$         - Going into differential mode for model
19:25:29.588 INFO  c.i.z.save.analysis.GrooveAnalysis   - Converting RealModel to Groove graph..
19:25:29.627 INFO  c.i.z.save.analysis.GrooveAnalysis   - Running Groove exploration..
19:25:30.041 INFO  c.i.z.save.analysis.GrooveAnalysis   - Groove analysis done!
Policy Violation of 'PolicyDeploymentExists'
19:25:30.070 INFO  com.ibm.zurich.save.actor.ProbeActor - disconnecting probe
19:25:30.071 INFO  com.ibm.zurich.save.actor.ProbeActor - disconnecting probe
19:25:30.070 INFO  com.ibm.zurich.save.actor.ProbeActor - disconnecting probe
good bye
root@core ~/SAVE#
```

Figure 131 - output of SAVE against a wrong deployment scenario

3.1.7.5 Conclusion

SAVE validation showed that this subcomponent has SUCCESSFULLY PASSED SAVE_1 validation activity

Requirement's assessment

LREQ5 – Transparency for the customer – The cloud provider can show the verification results of the VM's deployment and misconfiguration of the cloud provider. If used within the

Healthcare VMs or from external sources, it also prevents against a malicious cloud provider that manipulates the verification results.

3.2 Activities for Smart Lighting System scenario

Chapter Authors:

Alexander Bürger, Alysson Bessani, Marcel Santos, Paulo Santos, Marco Abitabile

The full set of validation activities rely on a distributed environment (Figure 132), comprised of:

- One BFT-SMaRt replica node running at Amazon commodity cloud
- One BFT-SMaRt replica node running at Azure commodity cloud
- Two Trusted Infrastructure clouds, each with:
  - One BFT-SMaRt replica node
  - One Smart Lighting Application node publicly accessible (online one active at one of the trusted clouds at a time)
  - One Smart Lighting Application node with restricted access (online one active at one of the trusted clouds at a time)
  - One standard Memcached node

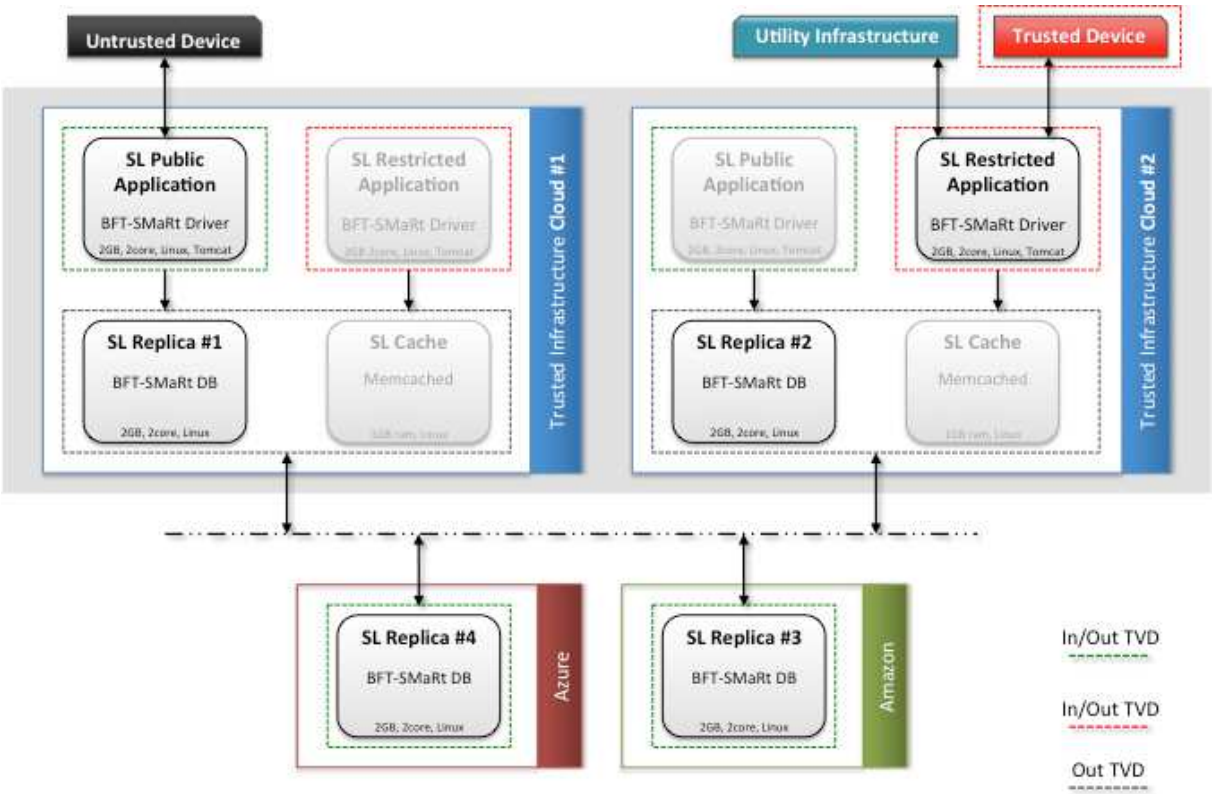


Figure 132 - Smart Lighting System validation deployment layout.

The purpose of this deployment layout is to provide redundancy at the application frontend, and high resiliency at persistency level.

In practice, for operational effectiveness during the validation procedures, both trusted clouds are hosted at the same cloud provider, on the same host, with only one public IP (134.147.232.38) with a simple port forwarding to reach internal nodes.

The deployment scenario described in the figure above is the same for all the validation activities that will be shown hereafter. The actual deployment scenario of SLS make use of BFT-SMaRt, Trusted Server, Trusted Channel and Trusted Object Manager, that are all the subcomponents that SLS uses, that is, the deployment scenario represents the final TClouds integrated system on which is deployed the final version of SLS.

3.2.1 Description of subcomponents

For the sake of ease of understanding, In this chapters will be briefly described the subcomponents that has been used by SLS and are part of the validation activities. All this subcomponents define the TClouds Trusted Infrastructure that offer a trust model that matches the needs of Smart Lighting System application.

As described also in D.1.3.3, customers of SLS applications constitute a limited quantity of users that are tightly related with the energy supply company. Differently form the healthcare scenario (that is a mass related application), SLS administrator ties a well-defined relation with its customers that is often constituted by an agreement among the parties and a signature of a contract of furniture of a specific service (in SLS case, the management of public lighting)

Given this, the trust model for SLS needs an higher level of “trustiness” since final user of SLS requires extremely high confidence that the underlying cloud Infrastructure is accessible only by a predefined set of users.

TClouds Trusted Infrastructure is able, in fact, to allow only specific hardware machines, that contains a specific hardware signature (given by the TPM module) that makes impossible from non-authorized to have access into the system

Besides an economic impact, public light management, has also impact on public safety, as such it demands a very high level of availability and resiliency. TClouds BFT-SMaRt, addresses this by supplying a highly resilient persistency solution in the form of a state machine replication Relational Database Management System.

3.2.1.1 BFT-SMaRt

The BFT-SMaRt cloud-of-clouds object storage service uses object storage services from diverse cloud providers (e.g., Amazon S3, Rackspace…) to build a dependable object storage service.
The core of the solution is a set of read/write protocols based on the use of Byzantine quorum replication [2] requiring 3f+1 clouds to tolerate up to f unavailable/compromised clouds. This proto-col addresses the mentioned requirements in the following way:

4) The system tolerates arbitrary (a.k.a. Byzantine) faults in order to cope with all possible behavior of a fraction of providers;
5) The replication protocol operates on an unreliable network in which messages can be lost and delayed and do not require participation of the full set of employed clouds, but only of a sub-set of them (a quorum [1]), on any step of the protocol execution;
6) The protocols are completely client-based, in the sense that no specific code is required in the cloud. DepSky assume the clouds provide storage service with standard (RESTful) operations for managing objects and containers (put, get, list,

etc.). Moreover, the set of storage clouds do not interact among themselves, but only with the clients;

State Machine Replication (SMR) is a classical fault tolerance technique in which a set of service replicas can be consistently updated in such a way that the crash of a subset of them does not prevent the service to be provided.

Byzantine fault tolerant (BFT) SMR leverages the fault tolerance to support arbitrary faults. These faults can be due to corruption in data, bugs in software and even intrusions.

BFT-SMaRt (see Chapter 2 of D2.2.4 for details) is a Byzantine Fault Tolerant State Machine Replication library. It uses 3f+1 replicas to tolerate up to f Byzantine faults.

We deployed BFT-SMaRt in a configuration using two nodes inside the Trusted Infrastructure and two commodity clouds (Amazon EC2 and Windows Azure) to tolerate crash and Byzantine faults. This creates a Byzantine fault-tolerant cloud-of-clouds setup.

In the validation activities performed BFT-SMaRt has been used jointly with a database server, SteelDB (see Chapter 6 of D2.2.4 for details) creating a replicated database server distributed on different commodity clouds. From now on we will refer to this system as SteelDB and BFT-SMaRt interchangeably.

3.2.1.2 Trusted Infrastructure (Trusted Server/Channel/Object manager)

The TClouds Trusted Infrastructure is an aggregation of several TClouds subcomponents. These subcomponents constitutes the foundation to build the trust model needed by SLS scenario. It consists of: the Trusted Server, the Trusted Channel and the Trusted Object Manager.

In the TrustedInfrastructure Cloud, a central management component, called TrustedObjects Manager (TOM), manages a set of TrustedServers (TS) which run a security kernel, which in turn run the virtual machines (VM) of the users. A virtual machine consists of the operating system (OS) and applications (App).

TS as well as the TOM, are equipped with a hardware security module (HSM). When started, the HSM is employed for secure boot, ensuring the integrity of the software (in particular of the security kernel). Moreover, the hard drives are encrypted by a key that is stored within the HSM. Via this sealing, the local hard drives can only be decrypted in case the HSM has crosschecked the integrity of the component. Hence only an un-tampered security kernel can be booted and can access the decrypted data. The security kernel enforces the security policy and the isolation.

The TOM is in charge of deploying configuration data (including key material and security policies) and VMs on the TrustedServers. Security services within the security kernel handle the configuration and ensure that the security policies are properly enforced. Encrypted communication of TOM and TrustedServer is via the Trusted Management Channel (TMC) which ensures the integrity using remote attestation before transmitting any data. All administrative tasks on the Trusted-Server are performed via the TMC, there is no other management channel for an administrator (like an ssh-shell).

3.2.2 Integration validation activities

The *Integration* validation activities intent to validate the Smart Lighting System as a whole having the selected TClouds security components integrated.

3.2.2.1 Integration_1

Activity ID	**Integration_1**
Activity type	Benchmarking
Activity description	Evaluate the infrastructure trustworthiness to prevent intrusions. 1- Confirm access to SL hosts employs state of the art secure mechanisms (ex. secure protocols; certificates)
Acceptance Criteria	Step 1 is successful

Table 24 - Integration_1 validation activity

The infrastructure hosting *Cloud#1* and *Cloud#2* (Figure 132), only provides a single access IP to outside, from which port forwarding rules provides access to a specific port of each node inside. Thus, being these ports the only way to access the nodes, from outside.

Each node provides solely a HTTPS (port 443) service. It's then up to the node itself to employ their access policy (Integration_2).

During the validation operations, SSH (port 22) was also open and configured to allow a secure remote access to the nodes.

Therefore, it's successfully validated the infrastructure trustworthiness by only forwarding access to the internal nodes that supply an external service.

3.2.2.2 Integration_2

Activity ID	**Integration_2**
Activity type	Benchmarking
Activity description	Evaluate the *persistence engine* trustworthiness to prevent intrusions. 1- Confirm access to the *persistence engine* employs state of the art secure mechanisms (ex. secure protocols; certificates)
Acceptance Criteria	Step 1 is successful

Table 25 - Integration_2 validation activity

This validation criterion was satisfied due to the use of three standard mechanisms for implementing secure distributed protocols.

The first mechanism is to have different username/password pairs (from now on called credentials) to access each SteelDB replica. When a JDBC connection is established, a

client needs to provide its credential for accessing all the replicas. This ensures that a compromised replica cannot use its credentials to open connections with other replicas.

The second mechanism is internal to BFT-SMaRt and is used to ensure the integrity of communications between the replicas and the replicas and clients. This mechanism comprises the use of Message Authentication Codes (MACs) based on the SHA-1 algorithm. More specifically, every pair of processes in BFT-SMaRt has a pair of shared keys that is used to create MACs for communicating in each direction. The use of this mechanism ensures that any modification on the messages will be detected and the message will be discarded at its destination.

The third and last mechanism is the use of IPSec (in Tunneled mode) between every replica on the public clouds and the VMs hosted in the *TrustedInfrastructure*. IPSec is used for ensuring that all communication between the trusted and untrusted part of the distributed system is confidential.

3.2.2.2.1.1 Validation setup

In Integration_3 we show that it is impossible to access H2 directly from outside of the trusted infrastructure. We also show in that validation activity that messages not authenticated are discarded from BFT-SMaRt. To validate this activity we setup IPSec in the public clouds. The process to setup IPSec took a lot of effort. We describe next the steps taken.

Using the Amazon EC2 Management Console, we created a virtual private cloud (VPC). In the VPC view we chose VPC Creation Wizard → VPC with a Single public Subnet Only**.**

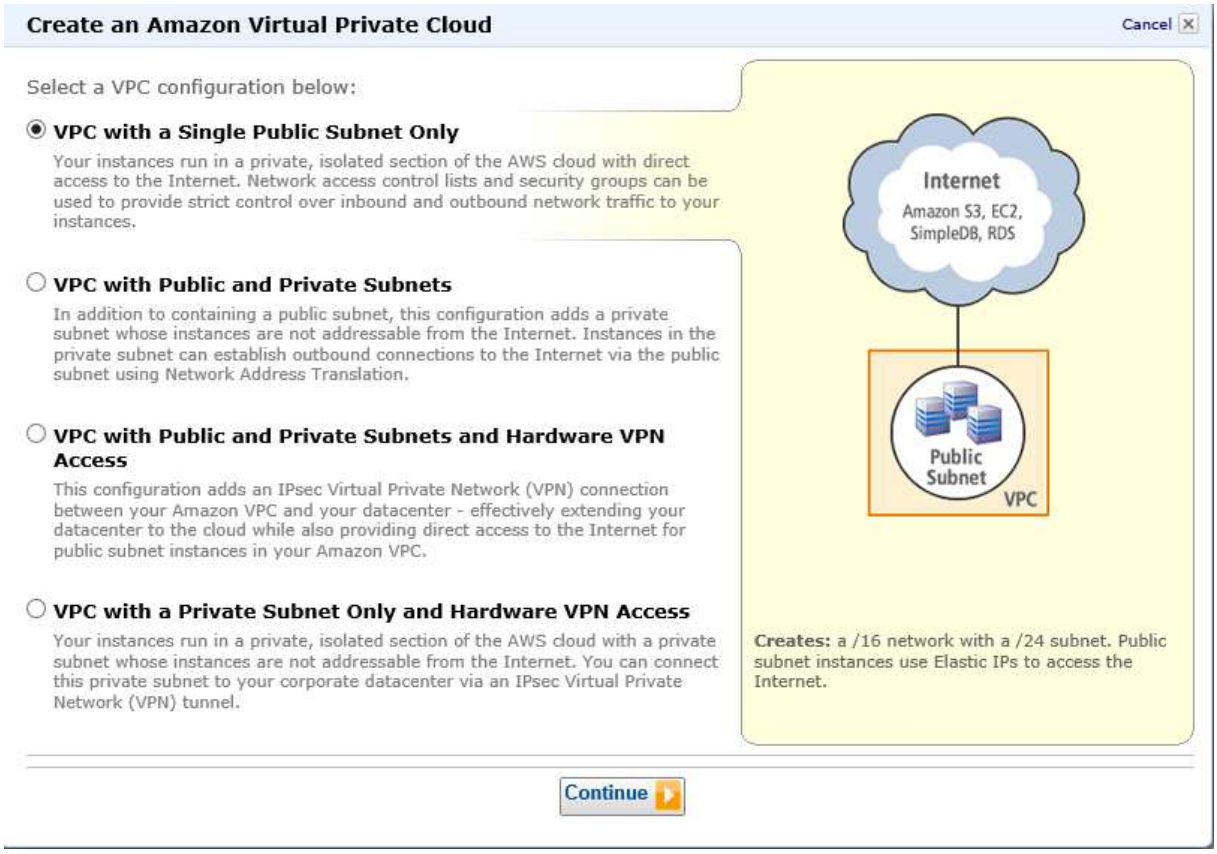


Figure 133 - Creation of Amazon virtual private cloud. Step1

Next, the address space (e.g. 10.0.0.0/16).

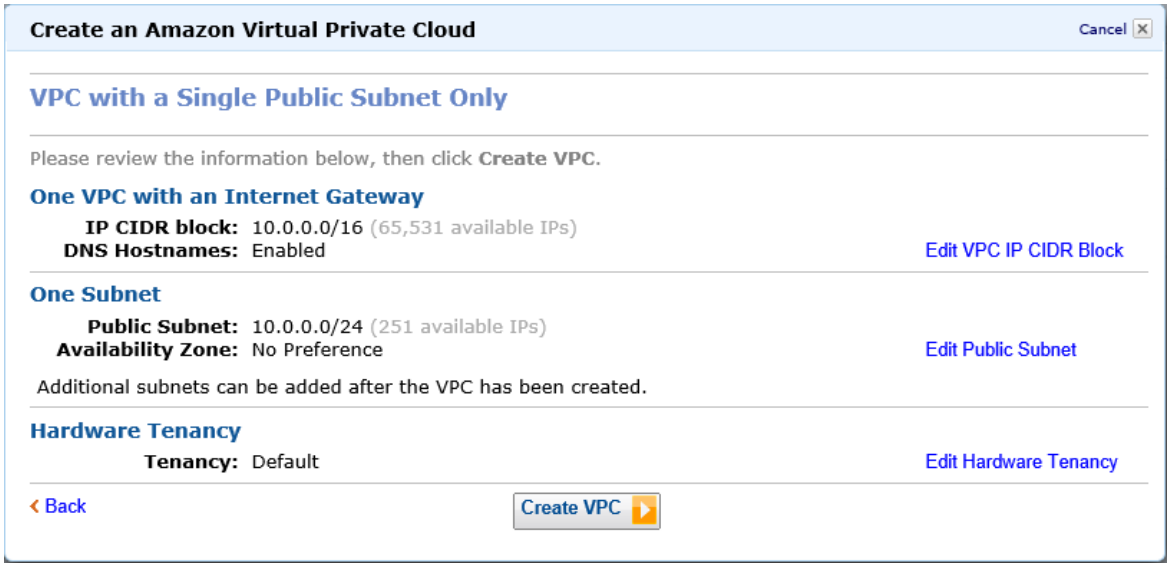


Figure 134 - Creation of Amazon virtual private cloud. Step2

Being VPC created, next we deployed an instance on Amazon EC2 used to host IPSec library. We then launched an Ubuntu 13.04 into the VPC Subnet just created:

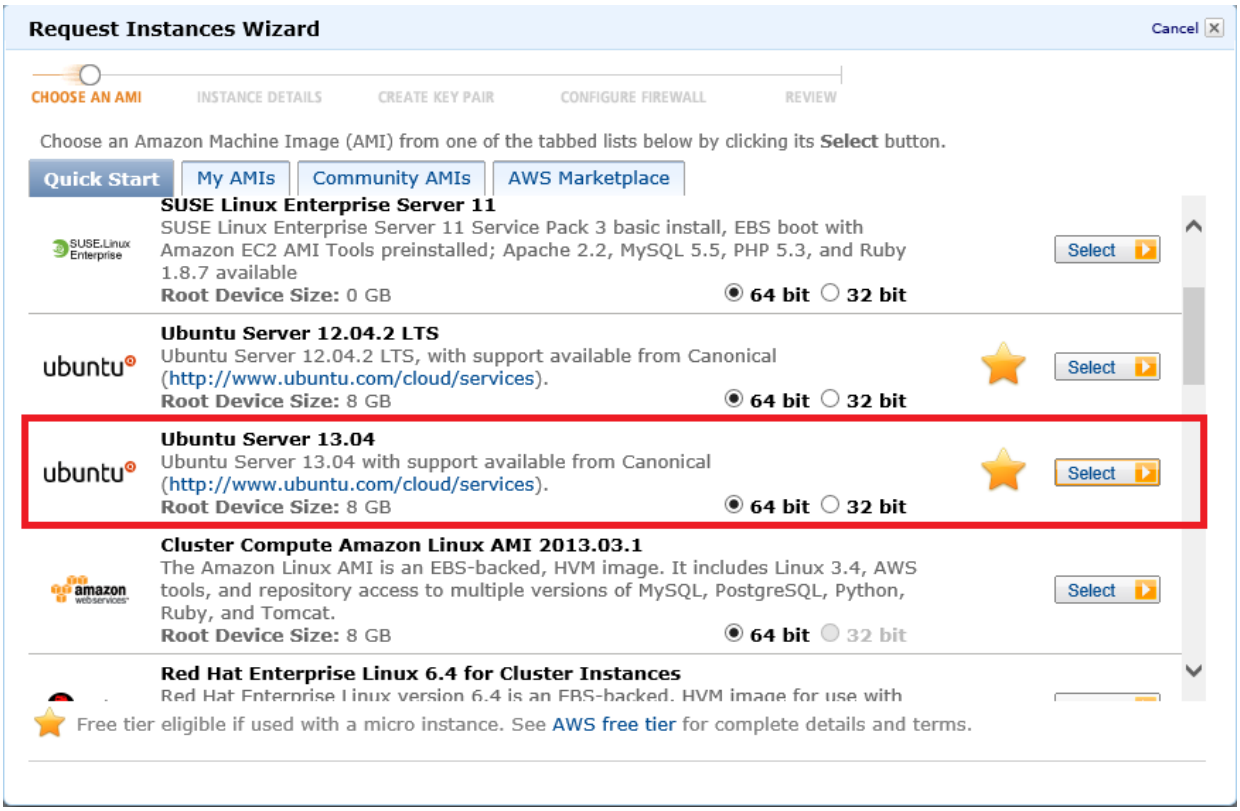


Figure 135 - launching Ubuntu instance into the private cloud created. Step1

We specified the VPC subnet previously deployed. I also chose a small instance instead of a micro instance.

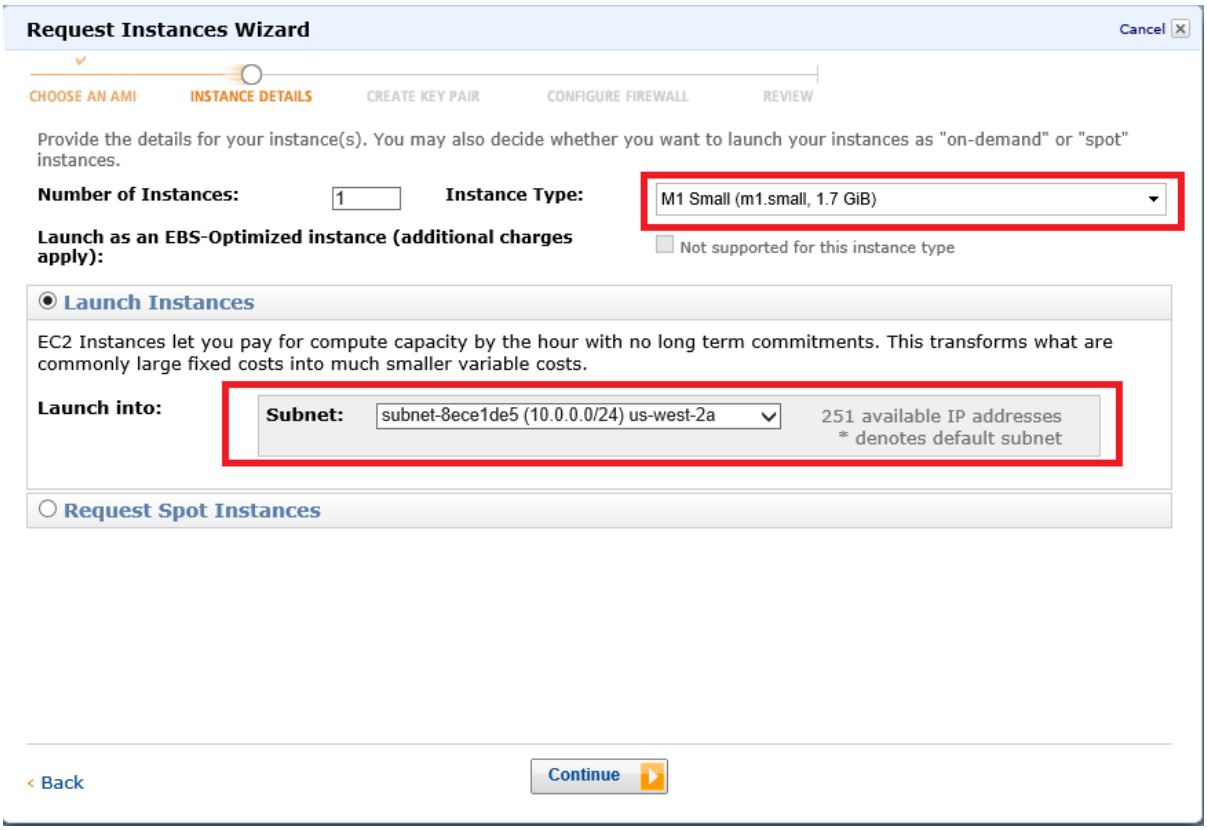


Figure 136 - launching Ubuntu instance into the private cloud created. Step2

With the instance launched, we associated a static IP to the machine. We opened the EC2 view and allocated a new Elastic IP. Next, we associated it with the previously created machine:

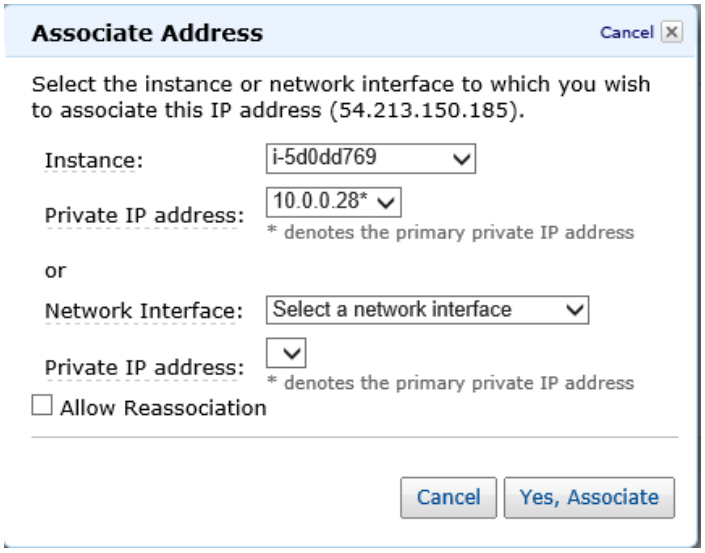


Figure 137 - launching Ubuntu instance into the private cloud created. Step3

This was the IP you will use to access the machine using ssh.

Before configuring the IPSec in EC2, we needed to configure the other side of the network, in Windows Azure. Going to Windows Azure Side, we created a Virtual Network to connect with

the Amazon EC2 VPC. To establish the IPSec tunnel we created gateway IP address and an authentication key from that side of the network.

In the Windows Azure Management Console, chose the options **+New -> Virtual Network -> Custom Create** in the bottom left of the screen:

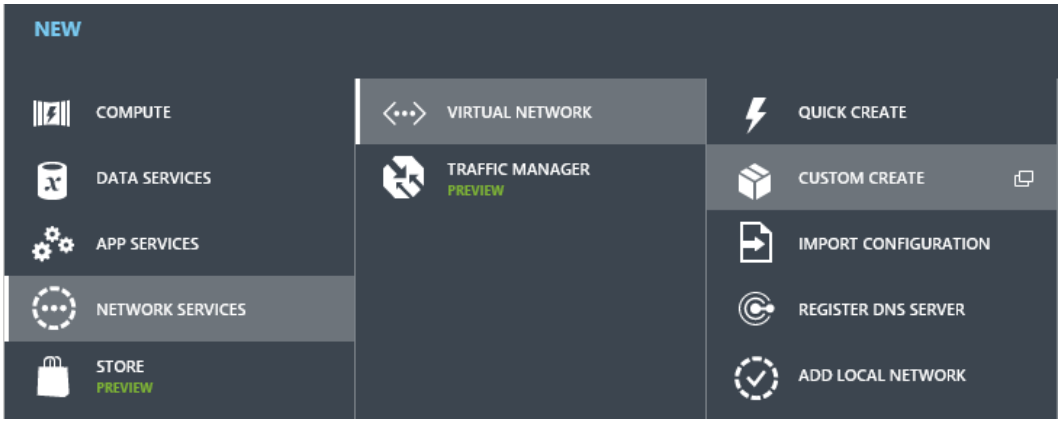


Figure 138 - creation of private network in Windows Azure. Step1

We specified the Region and VNET/Afinity Group Name:

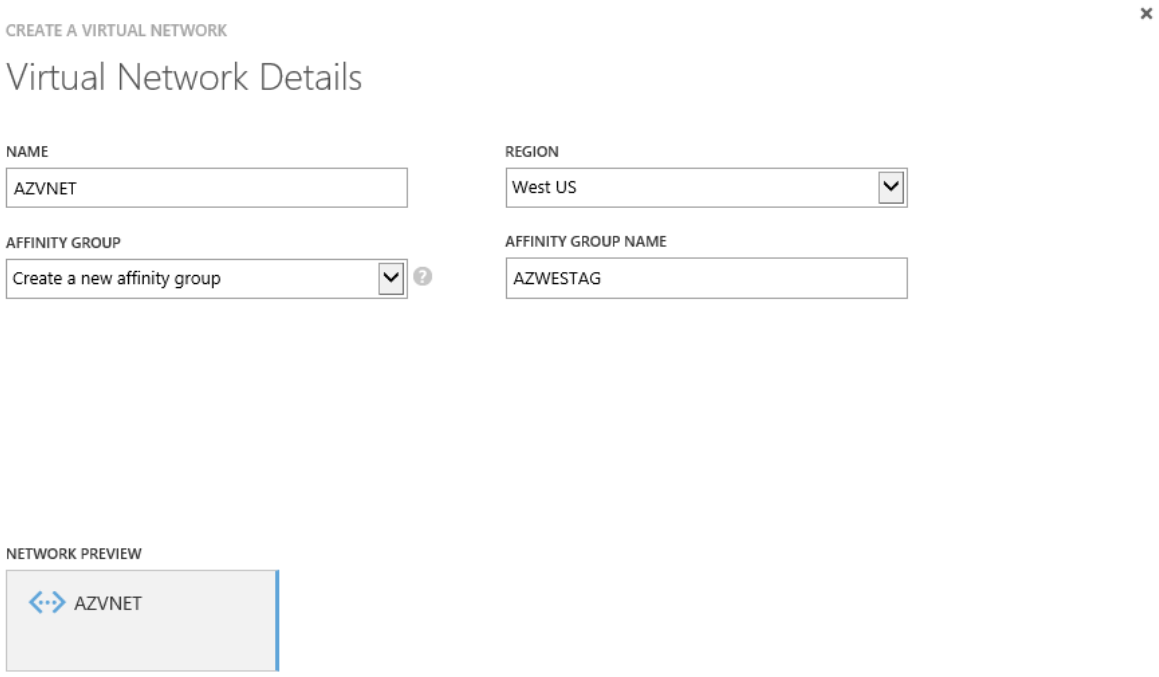


Figure 139 - creation of private network in Windows Azure. Step2

We chose side-to-side VPN

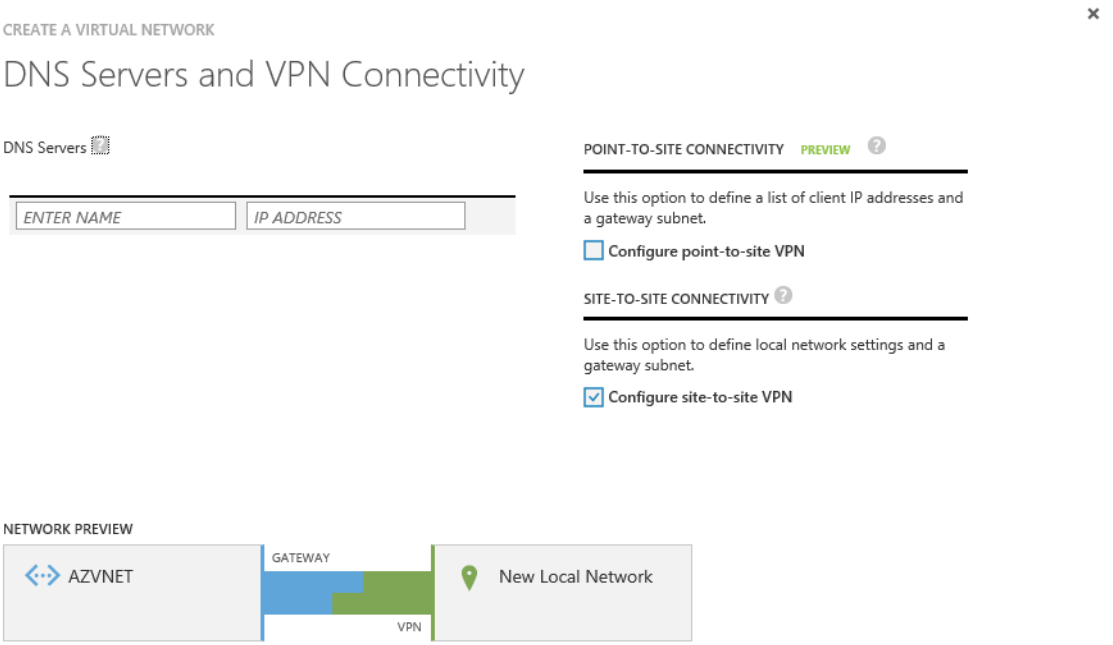


Figure 140 - creation of private network in Windows Azure. Step3

We defined the onsite network properties. The parameters were the Amazon VPC Address Space and Elastic IP of the previously created instance in Amazon EC2.

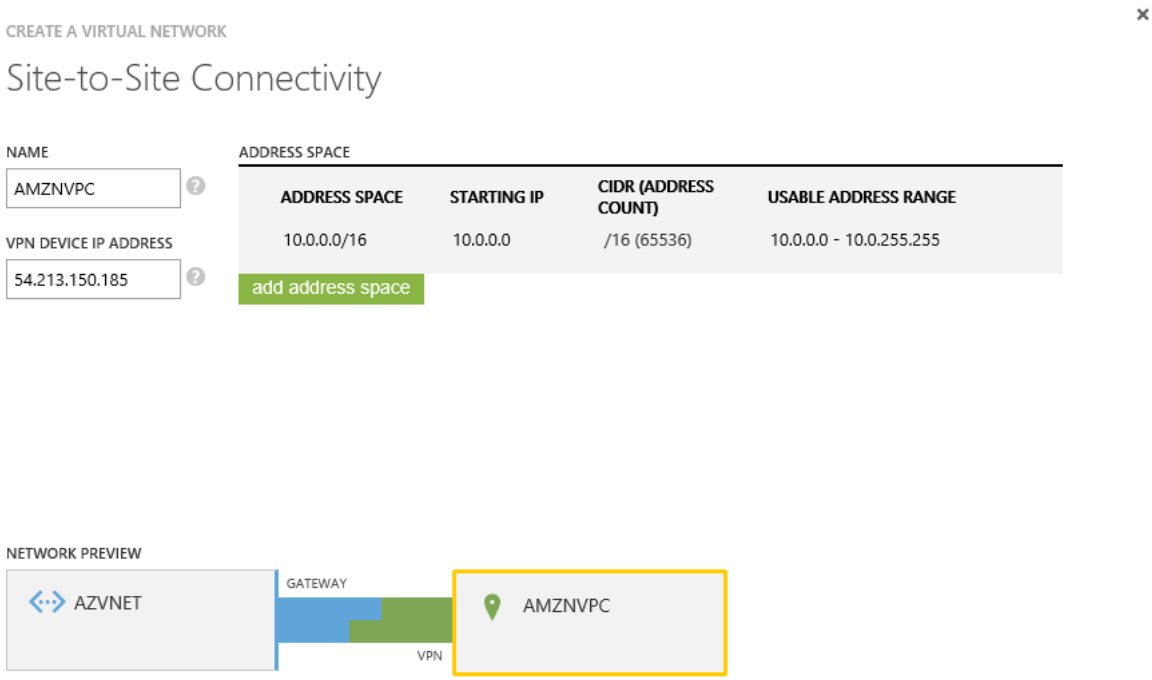


Figure 141 - creation of private network in Windows Azure. Step4

We then defined the Windows Azure address space. We also created a Gateway subnet and finished the creation of the Virtual Network.

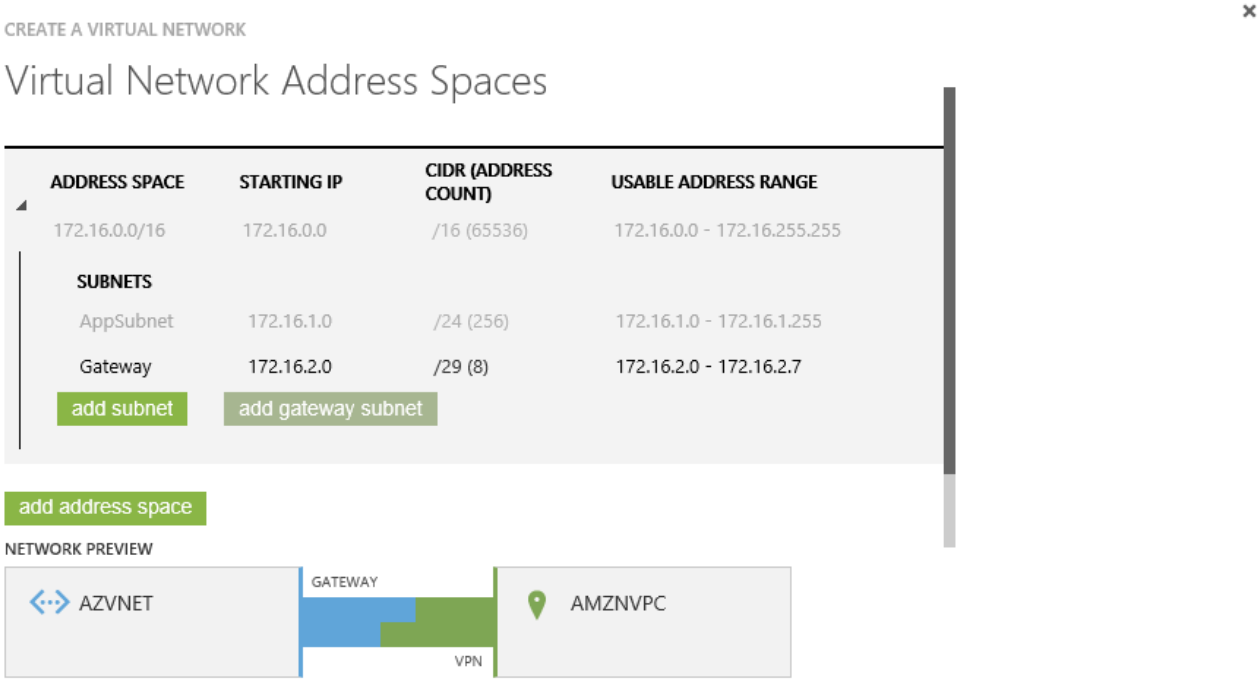


Figure 142 - creation of private network in Windows Azure. Step5

The next step was the creation of the Gateway to the Virtual Network just created. To do this, we opened the Virtual Network and clicked: **create gateway -> and select static routing.**

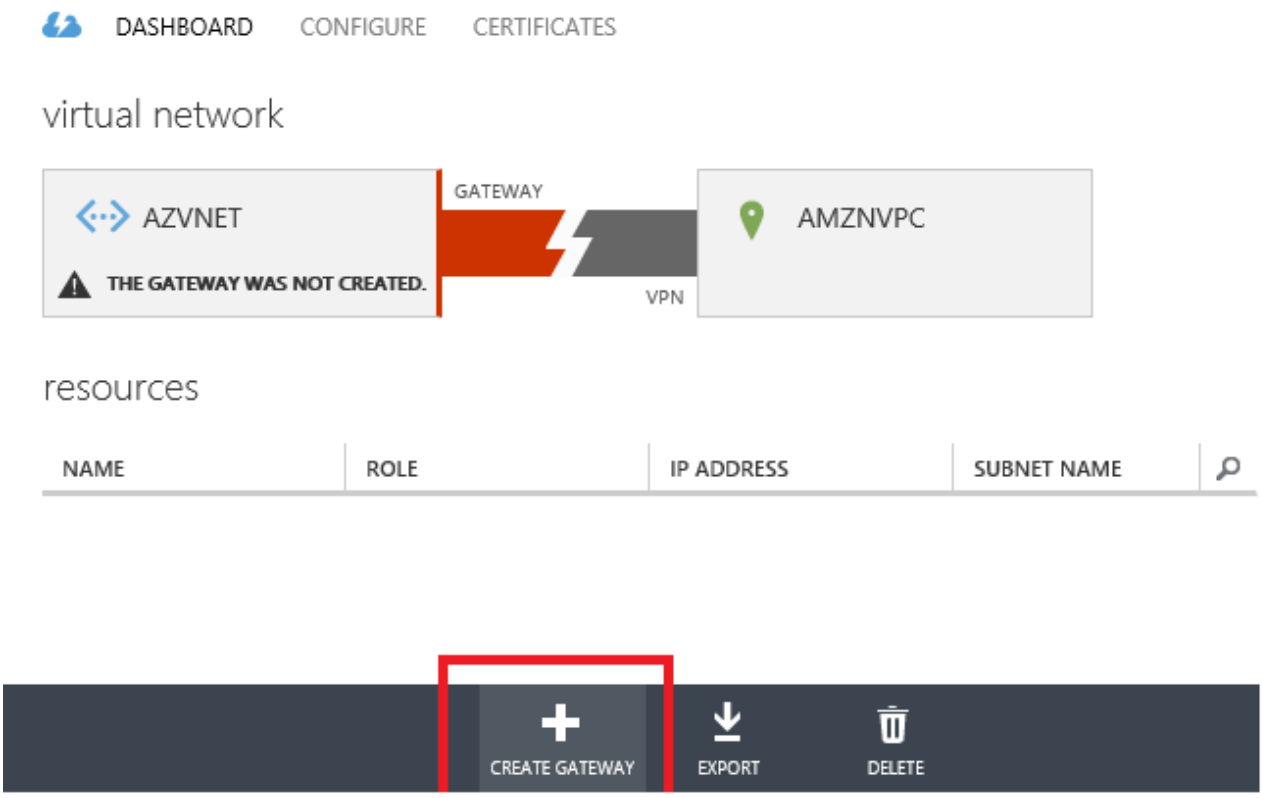


Figure 143 - creation of private network in Windows Azure. Step6

Once Gateway was created we got its IP address and the access key to configure IPSec in the Amazon side.

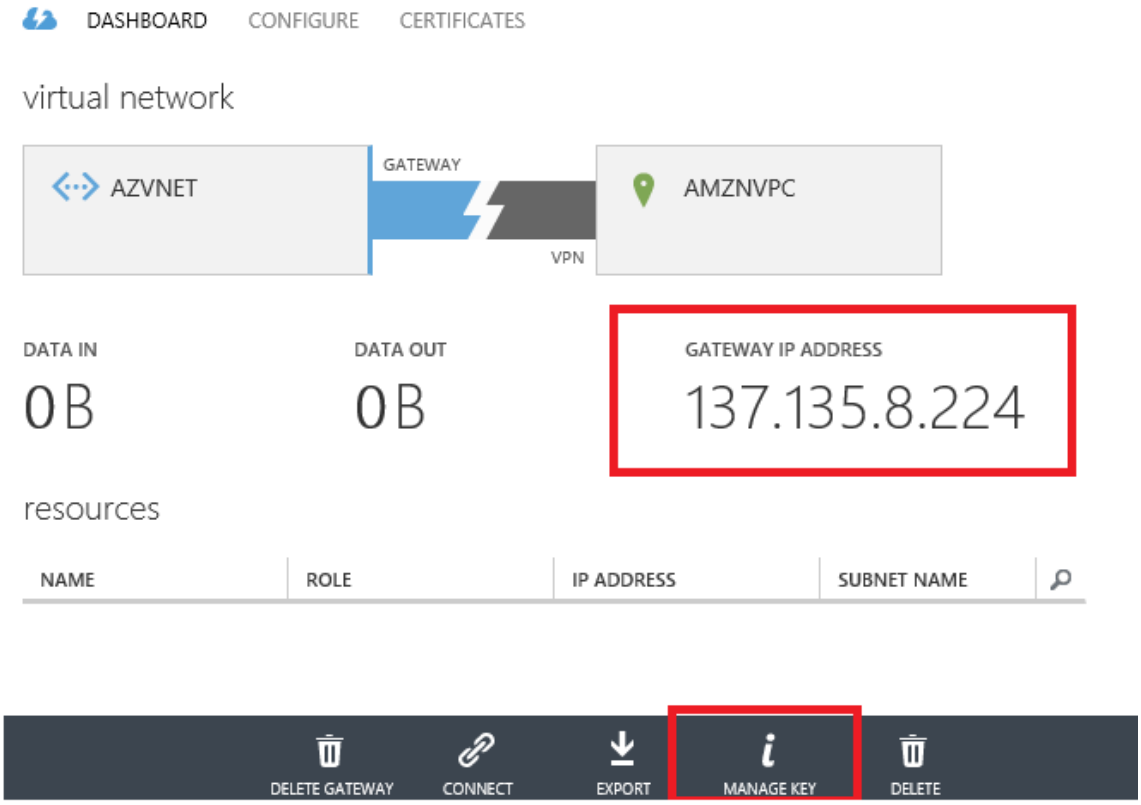


Figure 144 - creation of private network in Windows Azure. Step7

The next step was configure IPSec in the Amazon EC2 we created previously. We used OpenSwan to make IPSec configuration and use easier. To install it in Amazon EC2, we had to run:

```
sudo apt-get install openswan
```

We selected NO for installing a certificate since we used key based authentication.

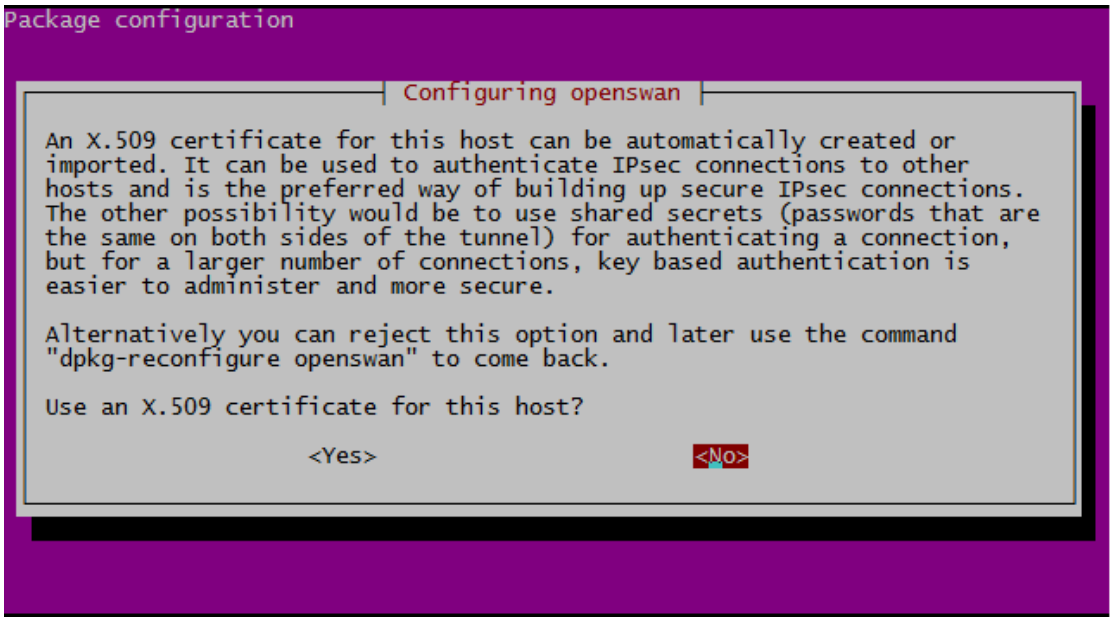


Figure 145 - IPSec creation in Amazon

After openswan was installed, the next step was editing the configuration files. First we configured `/etc/ipsec.conf` file. We replaced the content in this file with the following:

```
config setup
      protostack=netkey
      nat_traversal=yes
      virtual_private=%v4:10.0.0.0/16
      oe=off
include /etc/ipsec.d/*.conf
```

Next, we created the file `/etc/ipsec.d/amznazure.conf` and filled it with the following information:

```
conn amznazure
   authby=secret
   auto=start
   type=tunnel
   left=10.0.0.28
   leftsubnet=10.0.0.0/16
   leftnexthop=%defaultroute
   right=[WINDOWS AZURE GATEWAY IP]
   rightsubnet=172.16.0.0/16
   ike=aes128-sha1-modp1024
   esp=aes128-sha1
   pfs=no
```

Notes about the fields on this configuration files:

- left= is the local IP address of the Open Swan Server
- leftsubnet= is the local address space of the servers in the VPC
- right= is the IP Address of the Windows Azure VNET Gateway **(replace with your own)**
- rightsubnet= is the address space of the Windows Azure Virtual Network

After that, we specified the authentication key. We added the following line to the end of /etc/ipsec.secrets (without the brackets []):

```
10.0.0.28 [WINDOWS AZURE GATEWAY IP] : PSK "[WINDOWS AZURE GATEWAY KEY]"
```

Then, we enabled the IPv4 forwarding to the OpenSwan VM. This was done in the file /etc/sysctl.conf by uncomment the line:

```
net.ipv4.ip_forward=1
```

Next, we applied the changed network setting:

```
sudo sysctl -p /etc/sysctl.conf
```

The next step was to disable the source / destination checking on the Open Swan server. To do this, we went to the Amazon EC2 console, selected the instance created, clicked on **Actions -> Change Source/Dest check** and confirmed the action.

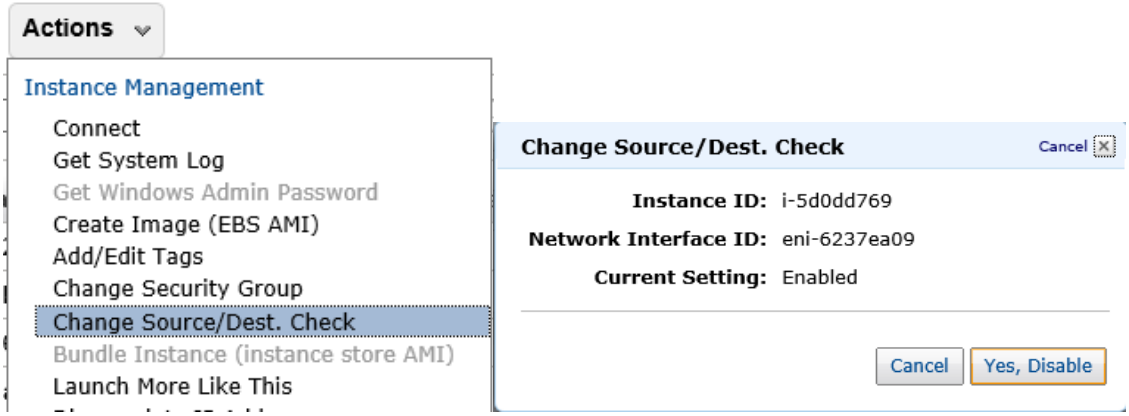


Figure 146 Figure 146 - disabling source/destination checking on OpenSwan server

Other step we needed to do was allow traffic from Windows Azure to the Amazon EC2 instance. To do this we changed the security group assigned to the OpenSwan server. In the Amazon management console we selected **Security Groups -> [your instance's security group]**, and added two custom UDP inbound rules – one for 500 and one for 4500 using the Windows Azure GW IP with /32 as the CIDR.

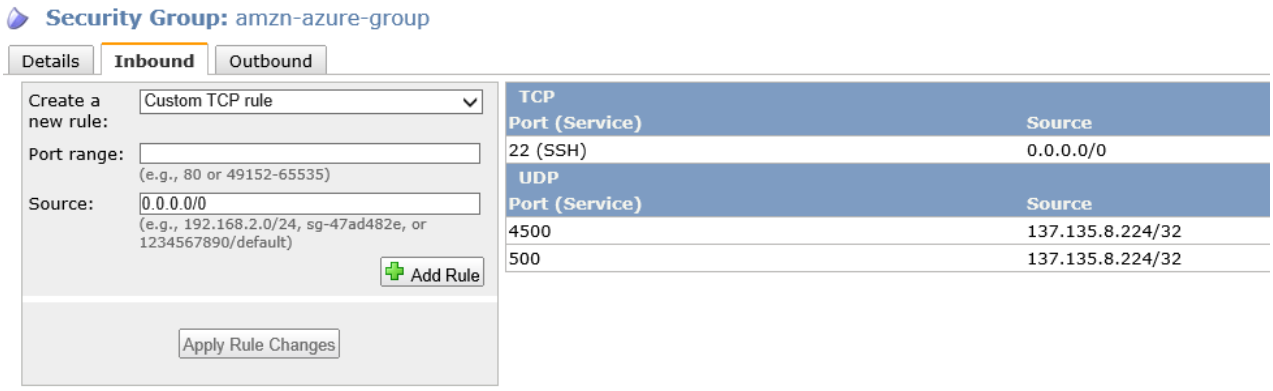


Figure 147 - allow traffic from Windows azure to Amazon

Finally, we restarted openswan to apply all changes.

```
sudo service ipsec restart
```

At that point Windows Azure Virtual Network was connected to the Amazon AWS VPC.

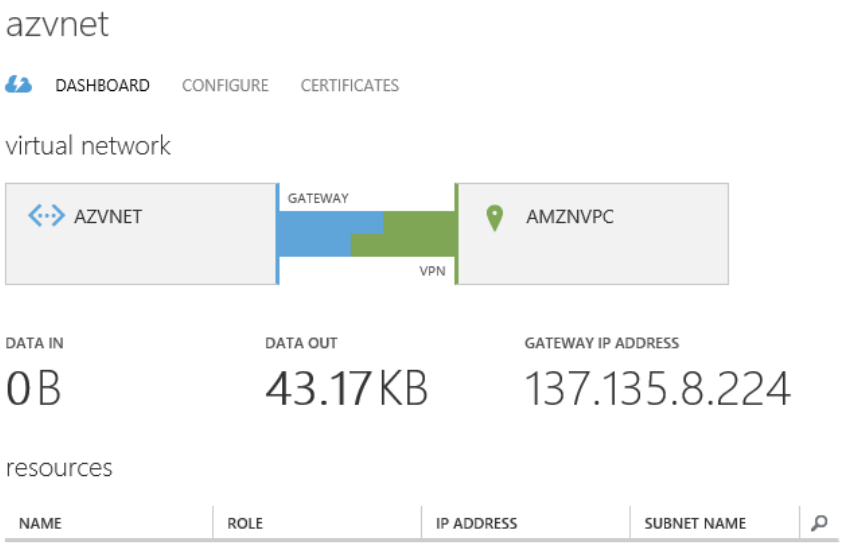


Figure 148 - network connection status

As the final configuration step, we added a new route to the routing table of the VPC we created. To do this, we went to Amazon VPC view and selected Route Tables. After that, we selected the VPC and added a new route to the 172.16.0.0/16 (Windows Azure Network) and that routes traffic through the instance ID of the Open Swan Server.

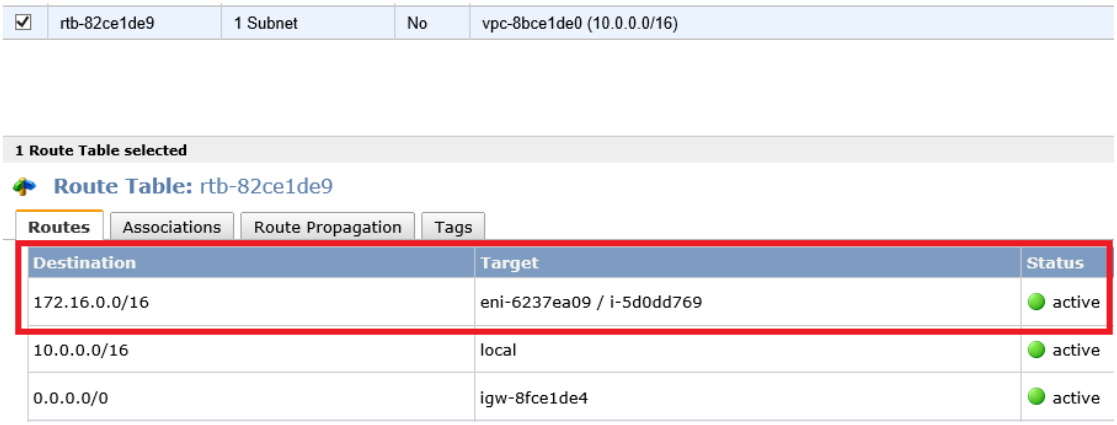


Figure 149 - New route added

To connect with the instance on EC2, we deployed an instance in the Windows Azure Virtual Network. To do that we chose the options **+New -> Compute -> Virtual Machine -> From Gallery.**

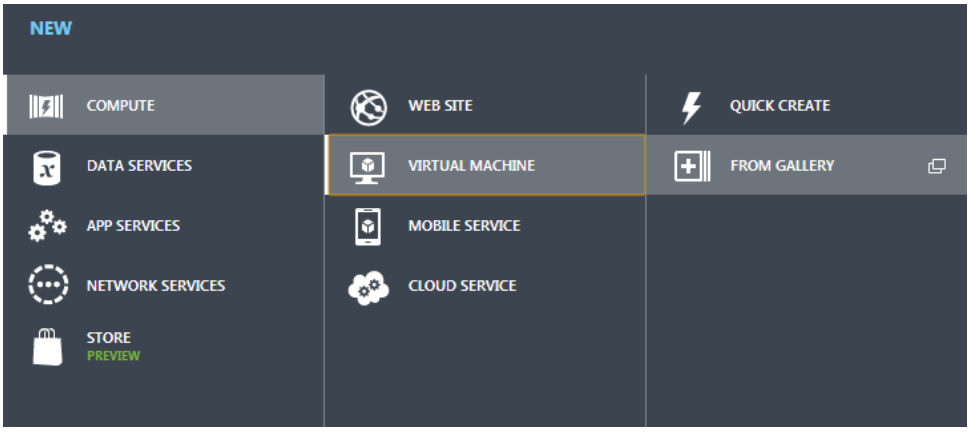


Figure 150 - instantiation of new VM into Azure. Step1

We chose the VM settings and the virtual network created to this new machine:

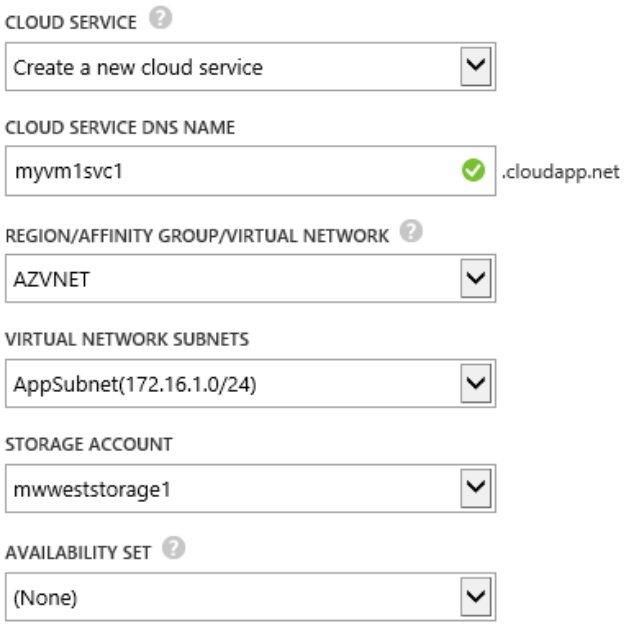


Figure 151 - instantiation of new VM into Azure. Step2

In this step we needed a Microsoft Certificate. Instructions to get a certificate was found in: http://www.windowsazure.com/en-us/manage/linux/how-to-guides/ssh-into-linux/ .

3.2.2.2.1.2 execution

To prove that the links between replicas are secure we used the command nc in both sides of the link. In one side we created a listener:

```
nc -l <port>
```

To create the message in the sender side we used:

```
nc <ip> <port>
```

We sent some text to the side that was listening in the defined port. To prove that the link is actually secure, we tried to send text from one side of the link to the other and obtained the same text, with the all configuration files correctly configured. Then, we corrupted the /etc/ipsec.secrets file (changing the authentication key) and ran nc again in both sides of the link. Now, we were not be able to obtain the text sent from one side to the other. It was important to run the restart command after corrupting the authentication key.

To install it in Windows Azure and Amazon EC2 (assuming that we deployed Debian based VMs), we need to run:

```
sudo apt-get install ipsec-tools
```

In the TI VMs (based on CentOS Linux) we need to manually install it. To do that, first we need to download ipsec-tools sources from its homepage (http://ipsec-tools.sourceforge.net/).

After that we should perform the following commands:

```
tar jxf ipsec-tools-x.y.z.tar.bz2
cd ipsec-tools-x.y.z
./configure
make
make install
```

Being `ipsec-tools` installed in all replicas, the next step is to configure the links between them. This links are configured in `/etc/ipsec-tools.conf`.

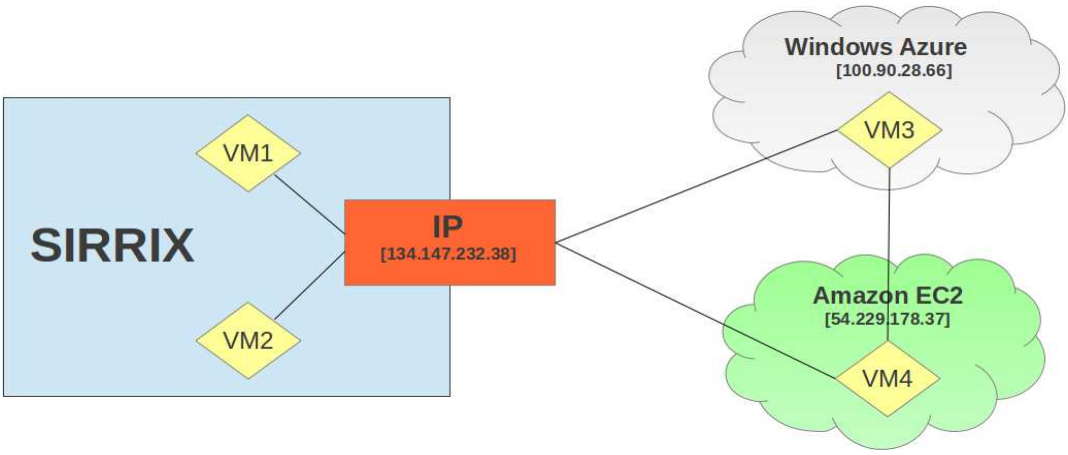


Figure 152 - final network configuration

The links are configured as shown in figure. Since TI VMs have the same IP, we only need to make secure links between the public clouds and this IP, which represents the border of the *TrustedInfrastructure*.

The configuration for the machines based on the figure need to be defined in their `/etc/ipsec-tools.conf` files are as follows:

```
                # ===== Configuration for WINDOWS AZURE ===== #

# Flush the SAD and SPD
flush;
spdflush;

# = ESP SAs using 192 bit long keys (168 + 24 parity) =
# node with EC2
add 137.117.212.174 54.229.178.37 esp 0x202 -E 3des-cbc
        0xbc54540582040508 9d502c378dbf69b1b4abcbff49684bf0;
add 54.229.178.37 137.117.212.174 esp 0x302 -E 3des-cbc
        0x9fec0ae7293b361795e107eca453848f41c32789e9c9c997;
```

```
# node with TI
add 137.117.212.174 134.147.232.38 esp 0x502 -E 3des-cbc
        0xb0cf3f2a80069ced4c094e40c3c9b21edc6244b2d52f5401;
add 134.147.232.38 137.117.212.174 esp 0x402 -E 3des-cbc
        0x41d5799b646e257db3218cd52b9d96b1a6fcc63e844caade;

# = Security policies =
#node with EC2
spdadd 137.117.212.174 54.229.178.37 any -P out ipsec
           esp/transport//require;

spdadd 54.229.178.37 137.117.212.174 any -P in ipsec
           esp/transport//require;

# node with Sirrix
spdadd 137.117.212.174 134.147.232.38 any -P out ipsec
           esp/transport//require;

spdadd 134.147.232.38 137.117.212.174 any -P in ipsec
           esp/transport//require;

                  # ===== Configuration for Amazon EC2 ===== #

# Flush the SAD and SPD
flush;
spdflush;

# = ESP SAs using 192 bit long keys (168 + 24 parity) =
# node with Azure
add 54.229.178.37 137.117.212.174 esp 0x201 -E 3des-cbc
        0x9fec0ae7293b361795e107eca453848f41c32789e9c9c997;
add 137.117.212.174 54.229.178.37 esp 0x301 -E 3des-cbc
        0xbc54540582040508 9d502c378dbf69b1b4abcbff49684bf0;

# node with Sirrix
add 54.229.178.37 134.147.232.38 esp 0x302 -E 3des-cbc
        0x6f39acbd21de4ef1322cc3d02d2d3a2c321703a19ed3486e;
add 134.147.232.38 54.229.178.37 esp 0x202 -E 3des-cbc
        0xc82a680389e651cfabc2d08484f0a25b02da8d3c9ac97670;

# Security policies
#node with Azure
spdadd 54.229.178.37 137.117.212.174 any -P out ipsec
           esp/transport//require;

spdadd 137.117.212.174 54.229.178.37 any -P in ipsec
           esp/transport//require;

# node with Sirrix
spdadd 54.229.178.37 134.147.232.38 any -P out ipsec
           esp/transport//require;

spdadd 134.147.232.38 54.229.178.37 any -P in ipsec
           esp/transport//require;

                  # ===== Configuration for Sirrix ===== #

# Flush the SAD and SPD
flush;
```

```
spdflush;

# = ESP SAs using 192 bit long keys (168 + 24 parity) =
# node with Azure
add 134.147.232.38 137.117.212.174 esp 0x201 -E 3des-cbc
        0x41d5799b646e257db3218cd52b9d96b1a6fcc63e844caade;
add 137.117.212.174 134.147.232.38 esp 0x301 -E 3des-cbc
        0xb0cf3f2a80069ced4c094e40c3c9b21edc6244b2d52f5401;

# node with EC2
add 134.147.232.38 54.229.178.37 esp 0x202 -E 3des-cbc
        0xc82a680389e651cfabc2d08484f0a25b02da8d3c9ac97670;
add 54.229.178.37 134.147.232.38 esp 0x302 -E 3des-cbc
        0x6f39acbd21de4ef1322cc3d02d2d3a2c321703a19ed3486e;

# = Security policies =
# node with Azure
spdadd 134.147.232.38 137.117.212.174 any -P out ipsec
           esp/transport//require;

spdadd 137.117.212.174 134.147.232.38 any -P in ipsec
           esp/transport//require;

# node with EC2
spdadd 134.147.232.38 54.229.178.37 any -P out ipsec
           esp/transport//require;

spdadd 54.229.178.37 134.147.232.38 any -P in ipsec
           esp/transport//require;
```

3.2.2.2.2 Conclusion

This validation criterion was satisfied due to the use of two standard mechanisms for implementing and deploying secure and distributed protocols.

The first mechanism is internal to BFT-SMaRt and is used to ensure the integrity of communications between the replicas and the replicas and clients. This mechanism comprises the use of Message Authentication Codes (MACs) based on the SHA-1 algorithm. More specifically, every pair of processes in BFT-SMaRt has a pair of shared keys that is used to create MACs for communicating in each direction (as will be show in Integration 3). The use of this mechanism ensures that any modification on the messages will be detected and the message will be discarded at its destination.

The second and last mechanism is the use of IPSec (in ESP mode) between every replica on the public clouds and the VMs hosted in the *TrustedInfrastructure* (TI). The idea here is to use standard technology for ensuring all communication between the trusted and untrusted part of the distributed system is confidential. The configuration of IPSec on the public clouds and in the *TrustedInfrastructure* is based on the ipsec-tools software package.

3.2.2.3 Integration_3

Activity ID	**Integration_3**
Activity type	Benchmarking
Activity description	Evaluate the *persistence engine* confidentiality. 1- Confirm data stored within the *persistence engine*, is only readable by authorized sessions.
Acceptance Criteria	Step 1 is successful

Table 26 - Integration:3 validation activity

There are three possible attack vectors for accessing the data stored in SteelDB without fully compromising one of the replicas, as show in Figure 1 and explained bellow:

- **Vector 1:** The adversary can make a login in the SteelDB and access the stored data as a normal authorized user.
- **Vector 2:** The adversary can fool BFT-SMaRt to create a request that will make the replica execute a command in its database and return a value.
- **Vector 3:** The adversary can connect directly to the backend database to execute SQL command and thus get some info about what is stored in the database.

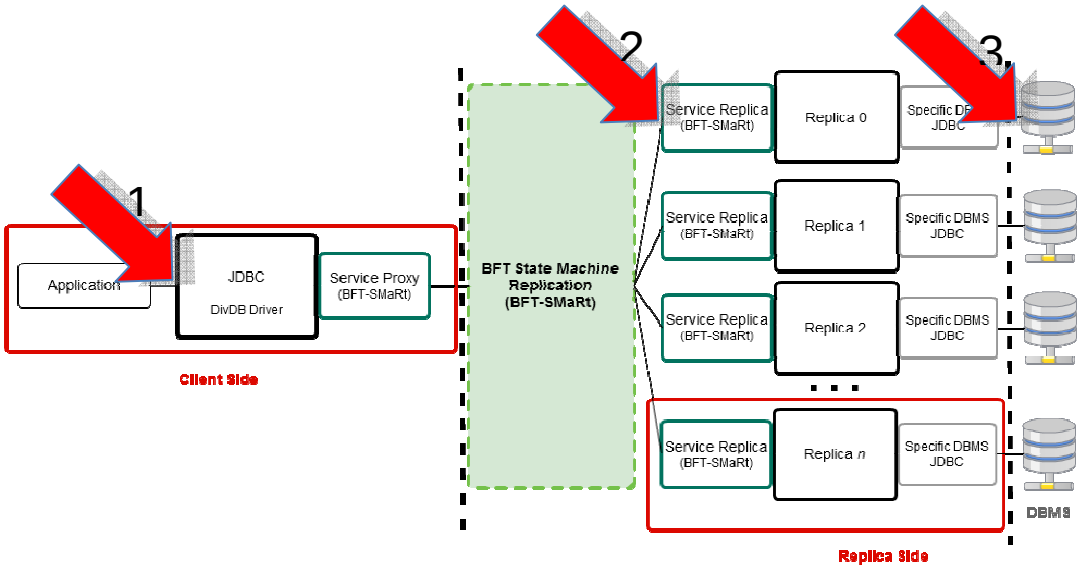


Figure 153 - Possible vectors of attack for SteelDB

In the following we validate that attacks based on any of these three vectors will not be successful in our current deployment.

3.2.2.3.1 Validation activity execution

Vector 1. For testing Vector 1, we used a SteelDB console to try access the replicated database from another machine.

```
mhsantos@mhsantos-laptop: ~/eclipse/workspace/SteelDBLeader
mhsantos@mhsantos-laptop:~/eclipse/workspace/SteelDBLeader$ ./runClient.sh 10000
Connecting to replica 0 at /134.147.232.38:3001
Impossible to connect to 0
Connecting to replica 1 at /134.147.232.38:3011
Impossible to connect to replica.
Impossible to connect to 1
Connecting to replica 2 at tclouds-medium.cloudapp.net/137.135.218.195:11000
Impossible to connect to replica.
Impossible to connect to 2
Connecting to replica 3 at ec2-54-229-177-243.eu-west-1.compute.amazonaws.com/54.229.177.243:11000
Impossible to connect to replica.
Impossible to connect to replica.
Impossible to connect to 3
Exception in thread "main" java.lang.RuntimeException: Impossible to connect to servers!
        at bftsmart.communication.client.netty.NettyClientServerCommunicationSystemClientSide.send(NettyClientServerCommunicationSyst
emClientSide.java:356)
        at bftsmart.tom.TOMSender.TOMulticast(TOMSender.java:149)
        at bftsmart.tom.ServiceProxy.invoke(ServiceProxy.java:188)
        at bftsmart.tom.ServiceProxy.invokeOrdered(ServiceProxy.java:135)
        at lasige.divdb.comm.MessageHandler.send(MessageHandler.java:96)
        at lasige.divdb.jdbc.BFTDriver.connect(BFTDriver.java:72)
        at java.sql.DriverManager.getConnection(DriverManager.java:571)
        at java.sql.DriverManager.getConnection(DriverManager.java:215)
        at lasige.divdb.demo.console.DBConsole.main(DBConsole.java:19)
Impossible to connect to replica.
Impossible to connect to replica.
mhsantos@mhsantos-laptop:~/eclipse/workspace/SteelDBLeader$
```

Figure 154 - Refused connection to SteelDB

As expected, the test failed. The reason it failed is that to login in SteelDB requires the client to connect to all replicas of the system. However, the *TrustedInfrastructure* VPNs prevent any attempt to connect to SL Replica #1 and SL Replica #2 replicas from outside the trusted TVD, and the public cloud machines (Azure and EC2) are configured with IPSec communication, accepting thus packets only from machines inside the trusted infrastructure (see Integration 2).

When the console is executed from inside the trusted infrastructure, the connection is successful, as displayed below.

```
tomcat@localhost:~/validation_fcul
[tomcat@localhost validation_fcul]$ ./runClient.sh 5000
Connecting to replica 0 at /192.168.21.2:11000
Channel connected
Connecting to replica 1 at /192.168.22.2:11000
Channel connected
Connecting to replica 2 at tclouds-medium.cloudapp.net/137.135.218.195:11000
Channel connected
Connecting to replica 3 at ec2-54-229-177-243.eu-west-1.compute.amazonaws.com/54.229.177.243:11000
Channel connected
SELECT * FROM profile;
Query executed. Results: 2
[tomcat@localhost validation_fcul]$
```

Figure 155 - Connection accepted when client is inside the trusted infrastructure

Vector 2. For testing Vector 2 we investigated what happens when we try to send a message to a SteelDB replica to try to make it run a command in the database backend. We tried to connect on the BFT-SMaRt ports using telnet and, as expected, the client couldn't connect.

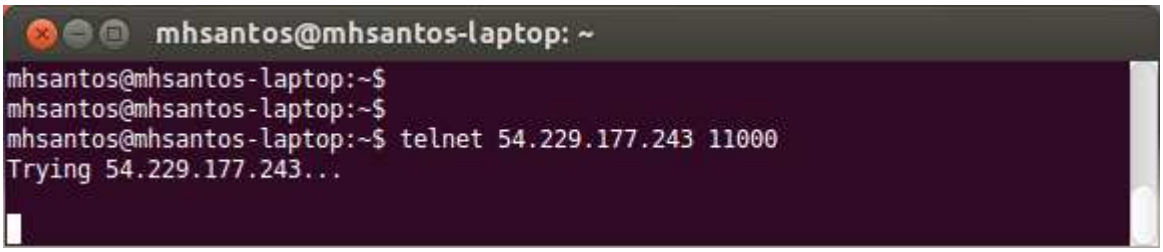


Figure 156 - Telnet connection not succeeded

Just as with vector 1, this did not work for several reasons. First, the replica VM is configured with IPSec for only accepting packets from the machines defined in the deployment. Second, even if a connection could be established, the BFT-SMaRt protocol requires another layer of security, in which only replicas sharing a key can communicate.

Vector 3. For testing Vector 3 we used H2 database client to try to connect directly to the H2 database deployed in the windows azure replica. Again, the test did not work.

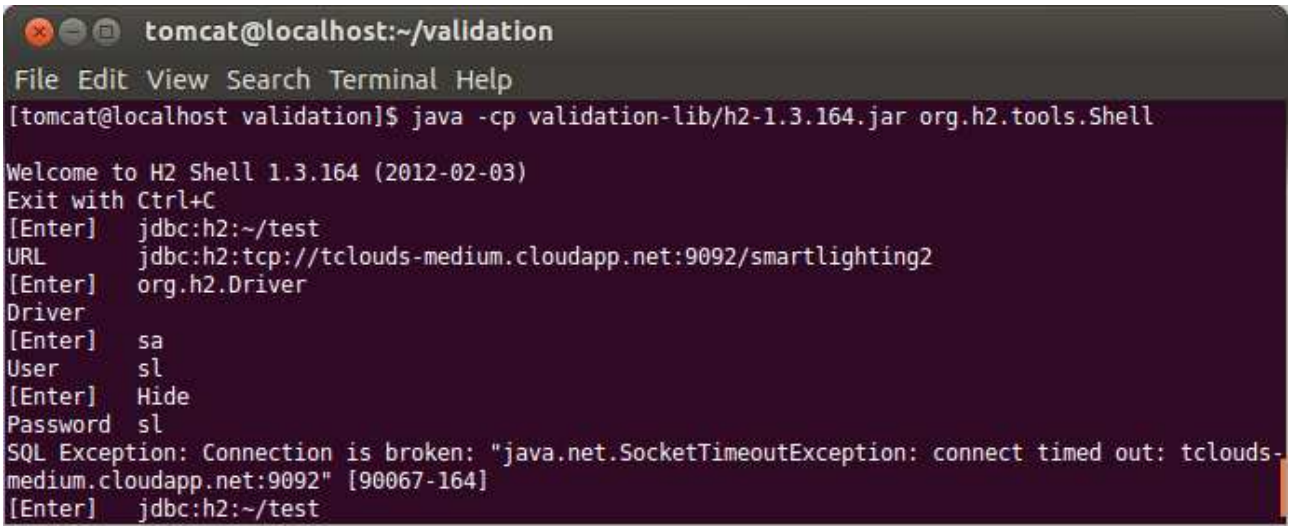


Figure 157 - Client couldn't access H2 database from outside the trusted infrastructure

The connection did not work for two reasons. First, the replica only accepts connections from previously configured IPSec peers. Second, all replicas have a firewall configured to accept packets only for the BFT-SMaRt ports (for running the distributed protocol), and not others.

3.2.2.3.2 Conclusion

The configuration of the virtual machines on the clouds plus the configuration in Trusted Infrastructure prevents users from getting access to ports used by the replication protocol.

Even in the case of a successful connection, the protocol ignores messages from unauthenticated users.

By employing secure communication channels, and only accept certified clients to connect to the replicas, it has been validated the persistence engine confidentiality.

3.2.2.4 Integration_4 & Integration_6

Integration_4 and Integration_6 validation activities address resiliency, performances and scalability. Since the setup and the components involved are the same, we decided to show their execution together in this chapter to help the reader to have an overview of the overall setup.

To understand better the components involved, the figure below shows which part of the deployment scenario (Figure 132) takes part to this validation activity:

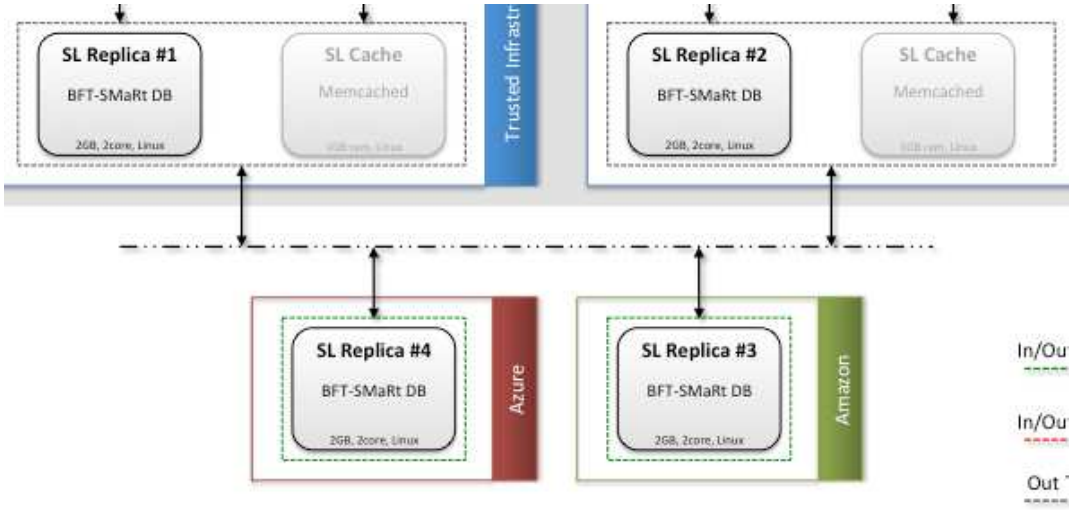


Figure 158 - Components involved in Integration_4 and Integration_6 validation activity

The four Database replica (SL Replica #1, #2, #3, #4) are stressed. Replica are maintained by TClouds BFT-SMaRt subcomponent.

Following are shown the validation activity description:

Activity ID	**Integration_4**
Activity type	Stress Test
Activity description	Evaluate the *persistence engine* resiliency ratio, when the number of faults is within the designed tolerance. 1- Being *f* (tolerated faults) nodes unreachable and 2 SL App nodes, with an automated script execute through SL Business Layer interface, 5, 10 and 20 simultaneous sessions doing: • 100 create actions over Schedules • 100 create actions over Users • 10 successful logins • 10 successful logouts • 10 unsuccessful logins • 100 edit actions over Schedules • 100 retrieval actions over Schedules • 100 edit actions over Users

	• 100 delete actions over Schedules • 100 delete actions over Users • 100 state reports generated • 100 auditing reports generated 2- Being *f* nodes compromised (reachable but with un-synched data) and 2 SL App nodes, with an automated script execute through SL Business Layer interface, 5, 10 and 20 simultaneous sessions doing: • 100 create actions over Schedules • 100 create actions over Users • 10 successful logins • 10 successful logouts • 10 unsuccessful logins • 100 edit actions over Schedules • 100 retrieval actions over Schedules • 100 edit actions over Users • 100 delete actions over Schedules • 100 delete actions over Users • 100 state reports generated • 100 auditing reports generated 3- Being all nodes online and 2 SL App nodes, with an automated script execute through SL Business Layer interface, 5, 10 and 20 simultaneous sessions doing (while doing it, disconnect *f* nodes): • 100 create actions over Schedules • 100 create actions over Users • 10 successful logins • 10 successful logouts • 10 unsuccessful logins • 100 edit actions over Schedules • 100 retrieval actions over Schedules • 100 edit actions over Users • 100 delete actions over Schedules • 100 delete actions over Users • 100 state reports generated • 100 auditing reports generated
Acceptance Criteria	The success ratio of all steps is 100%

Table 27 - Integration_4 validation activity

Activity ID	**Integration_6**
Activity type	Stress Test
Activity description	Evaluate the *persistence engine* performance levels and scalability. 1- Being all nodes online and 2 SL App nodes, with an automated script execute through SL Business Layer interface, 5, 10 and 20 simultaneous sessions doing: • 100 create actions over Schedules • 100 create actions over Users • 10 successful logins • 10 successful logouts • 10 unsuccessful logins • 100 edit actions over Schedules • 100 retrieval actions over Schedules • 100 edit actions over Users • 100 delete actions over Schedules • 100 delete actions over Users • 100 state reports generated • 100 auditing reports generated 2- Being *f* (tolerated faults) nodes unreachable and 2 SL App nodes, with an automated script execute through SL Business Layer interface, 5, 10 and 20 simultaneous sessions doing: • 100 create actions over Schedules • 100 create actions over Users • 10 successful logins • 10 successful logouts • 10 unsuccessful logins • 100 edit actions over Schedules • 100 retrieval actions over Schedules • 100 edit actions over Users • 100 delete actions over Schedules • 100 delete actions over Users • 100 state reports generated • 100 auditing reports generated 3- Being *f* nodes compromised (reachable but with un-synched data) and 2 SL App nodes, with an automated script execute through SL Business Layer interface, 5, 10 and 20 simultaneous sessions doing: • 100 create actions over Schedules • 100 create actions over Users • 10 successful logins • 10 successful logouts • 10 unsuccessful logins

<table>
<tr>
<td></td>
<td>

100 edit actions over Schedules
100 retrieval actions over Schedules
100 edit actions over Users
100 delete actions over Schedules
100 delete actions over Users
100 state reports generated
100 auditing reports generated

4- Being all nodes online and 2 SL App nodes, with an automated script execute through SL Business Layer interface, 5, 10 and 20 simultaneous sessions doing (while doing it, disconnect f nodes):

100 create actions over Schedules
100 create actions over Users
10 successful logins
10 successful logouts
10 unsuccessful logins
100 edit actions over Schedules
100 retrieval actions over Schedules
100 edit actions over Users
100 delete actions over Schedules
100 delete actions over Users
100 state reports generated
100 auditing reports generated

</td>
</tr>
<tr>
<td>Acceptance Criteria</td>
<td>
For step#1 the average response time (in milliseconds) shouldn’t overpass 2 times the average response for a direct access to the underlying database as presented in the following table:
<table>
<tr><th>Action \ sessions</th><th>5</th><th>10</th><th>20</th></tr>
<tr><td>100 create actions over Schedules</td><td>20</td><td>59</td><td>67</td></tr>
<tr><td>100 create actions over Users</td><td>8</td><td>18</td><td>25</td></tr>
<tr><td>10 successful logins</td><td>166</td><td>381</td><td>722</td></tr>
<tr><td>10 successful logouts</td><td>6</td><td>14</td><td>21</td></tr>
<tr><td>10 unsuccessful logins</td><td>161</td><td>368</td><td>708</td></tr>
<tr><td>100 edit actions over Schedules</td><td>51</td><td>162</td><td>167</td></tr>
<tr><td>100 retrieval actions over Schedules</td><td>18</td><td>61</td><td>60</td></tr>
<tr><td>100 edit actions over Users</td><td>13</td><td>34</td><td>37</td></tr>
<tr><td>100 delete actions over Schedules</td><td>20</td><td>54</td><td>57</td></tr>
<tr><td>100 delete actions over Users</td><td>7</td><td>16</td><td>19</td></tr>
</table>
</td>
</tr>
</table>

	100 state reports generated	31	68	96
	100 audit reports generated	7	15	24
	For the next steps, since there's no default comparison possible (traditional approach does not support faults), the average response time (when a fault exists) shouldn't overpass **2** times the average response time to when there're no faults (step#1). Meaning, steps 2; 3; and 4 shouldn't reach 2 times more average response time then the average obtained from step 1.			

Table 28- Integration_6 validation activity

3.2.2.4.1 Validation activities' setup

To efficiently execute *Integration_4* and *Integration_6* validation activities, an automated software process was developed. This process (written in Java) uses the BFT-SMaRt client layer to dispatch requests to the SMR and the same service layer as the Smart Lighting System.

Since the goal is to validate the resiliency of BFT-SMaRt working in conjunction with a web application, each validation client creates a number of threads to simulate simultaneous request from user sessions.

Each replica is started in its respective node server and responds to requests from two client servers. The replication nodes configuration is the same as if it were for the Smart Lighting System.

After running each step a file with the result is generated.

Since each action is the same in each step, these actions are hardcoded into the validation process. However some parameters can be defined by passing a properties file:

```
# name of the test suite and consequently the result file name
name=Integration6-step3

# number of concurrent threads to launch in this test
sessions=5

# number of repetitive executions of an action before collecting statistics (warm-up phase)
warmup=500

# number of schedule actions to execute
scheduleBatchSize=100

# number of user actions to execute
userBatchSize=100

# number of login actions to execute
loginBatchSize=10

# number of failed login actions to execute
failLoginBatchSize=10

# number of listing actions to execute
reportsBatchSize=100

# the following properties define a direct connection
# to the underlying database (in this case a H2)
# and the SQL instruction to execute in order to compromise the DB and therefore the node.
# In the absence of these properties no “attack” is executed
```

```
compromise.driver=org.h2.Driver
compromise.node=jdbc:h2:tcp://192.168.24.2:9092/smartlighting3;MVCC=TRUE;USER=sl;PASSWORD=sl
compromise.sql=delete from timetable
```

3.2.2.4.2 Integration_4 Validation activity execution

To execute the activity it is now required to pass a property file with the execution parameters explained previously. The name of each file indicates the step that it refers to, with the following format:

```
<section>-step<step number>-<number of sessions>sessions.properties
```

As an example, `Integration4-step3-20sessions.properties` has the properties to run *Integration_4*, step 3 with 20 simultaneous sessions.

To execute all steps for this activity it's necessary to properly define 9 independent property files.

```
Integration4-step1-10sessions.properties
Integration4-step1-20sessions.properties
Integration4-step1-5sessions.properties
Integration4-step2-10sessions.properties
Integration4-step2-20sessions.properties
Integration4-step2-5sessions.properties
Integration4-step3-10sessions.properties
Integration4-step3-20sessions.properties
Integration4-step3-5sessions.properties
```

Each step runs 12 actions, each action triggering a transaction block with a set of SQL statements similar to:

```
-- Schedule Creation
insert into TIMETABLE (NAME, OPERATIONAL_AREA, VERSION, TIMETABLEUID) values (?, ?, ?, ?)
insert into PERIOD (END, START, TIMETABLE, VERSION, PERIODUID) values (?, ?, ?, ?, ?)
-- x2:
insert into CONTROL (MODE, OFFSET, RANK, PERIOD, SPECIAL_DAY_SERVICE, TARGET_STATE, TIME, VERSION,
CONTROLUID) values (?, ?, ?, ?, ?, ?, ?, ?, ?)


-- User Creation
insert into USER (CLIENT, DEATH_DATE, DELETED, EMAIL, FAILED_LOGINS, LOCKED, LOCKED_DATE, LOGIN, NAME,
OPERATIONAL_AREA, PASSWORD, SALT, VERSION, USERUID) values (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)


-- Login
select user0_.USERUID as USERUID9_, user0_.CLIENT as CLIENT9_, user0_.DEATH_DATE as DEATH3_9_,
user0_.DELETED as DELETED9_, user0_.EMAIL as EMAIL9_, user0_.FAILED_LOGINS as FAILED6_9_, user0_.LOCKED
as LOCKED9_, user0_.LOCKED_DATE as LOCKED8_9_, user0_.LOGIN as LOGIN9_, user0_.NAME as NAME9_,
user0_.OPERATIONAL_AREA as OPERATI11_9_, user0_.PASSWORD as PASSWORD9_, user0_.SALT as SALT9_,
user0_.VERSION as VERSION9_ from USER user0_ where lower(user0_.LOGIN)=? and user0_.DELETED=? limit ?
insert into AUDIT_ACTION (CONTEXT, ACTION_DATE, LOGIN, TEXT, TYPE, USERUID, VERSION, AUDIT_ACTIONUID)
values (?, ?, ?, ?, ?, ?, ?, ?)


-- Logout
```

```
insert into AUDIT_ACTION (CONTEXT, ACTION_DATE, LOGIN, TEXT, TYPE, USERUID, VERSION, AUDIT_ACTIONUID)
values (?, ?, ?, ?, ?, ?, ?, ?)


-- Failed Login

select user0_.USERUID as USERUID9_, user0_.CLIENT as CLIENT9_, user0_.DEATH_DATE as DEATH3_9_,
user0_.DELETED as DELETED9_, user0_.EMAIL as EMAIL9_, user0_.FAILED_LOGINS as FAILED6_9_, user0_.LOCKED
as LOCKED9_, user0_.LOCKED_DATE as LOCKED8_9_, user0_.LOGIN as LOGIN9_, user0_.NAME as NAME9_,
user0_.OPERATIONAL_AREA as OPERATI11_9_, user0_.PASSWORD as PASSWORD9_, user0_.SALT as SALT9_,
user0_.VERSION as VERSION9_ from USER user0_ where lower(user0_.LOGIN)=? and user0_.DELETED=? limit ?

select applicatio0_.PARAMETERUID as PARAMETE1_14_, applicatio0_.NAME as NAME14_, applicatio0_.VALUE as
VALUE14_, applicatio0_.VERSION as VERSION14_ from APPLICATION_SETTING applicatio0_ where
applicatio0_.NAME=? limit ?

update USER set CLIENT=?, DEATH_DATE=?, DELETED=?, EMAIL=?, FAILED_LOGINS=?, LOCKED=?, LOCKED_DATE=?,
LOGIN=?, NAME=?, OPERATIONAL_AREA=?, PASSWORD=?, SALT=?, VERSION=? where USERUID=? and VERSION=?


-- Schedule Modification

select timetable0_.TIMETABLEUID as TIMETABL1_1_0_, timetable0_.NAME as NAME1_0_,
timetable0_.OPERATIONAL_AREA as OPERATIO3_1_0_, timetable0_.VERSION as VERSION1_0_ from TIMETABLE
timetable0_ where timetable0_.TIMETABLEUID=?

update TIMETABLE set NAME=?, OPERATIONAL_AREA=?, VERSION=? where TIMETABLEUID=? and VERSION=?

select profile0_.PROFILEUID as PROFILEUID4_, profile0_.DESCRIPTION as DESCRIPT2_4_, profile0_.NAME as
NAME4_, profile0_.OPERATIONAL_AREA as OPERATIO4_4_, profile0_.TIMETABLE as TIMETABLE4_,
profile0_.VERSION as VERSION4_ from PROFILE profile0_ where profile0_.TIMETABLE=?

select service0_.SERVICEUID as SERVICEUID19_, service0_.NAME as NAME19_, service0_.PROFILE as
PROFILE19_, service0_.TIMETABLE as TIMETABLE19_, service0_.VERSION as VERSION19_ from SERVICE service0_
where service0_.TIMETABLE=?

select dtc0_.DTCUID as DTCUID5_, dtc0_.BIRTH_DATE as BIRTH2_5_, dtc0_.CLIENTUID as CLIENTUID5_,
dtc0_.DEATH_DATE as DEATH4_5_, dtc0_.DELETED as DELETED5_, dtc0_.DISTRICTUID as DISTRICT6_5_,
dtc0_.HASPHOTOCELL as HASPHOTO7_5_, dtc0_.IPADDRESS as IPADDRESS5_, dtc0_.LATITUDE as LATITUDE5_,
dtc0_.LONGITUDE as LONGITUDE5_, dtc0_.MUNICIPALITYUID as MUNICIP11_5_, dtc0_.NAME as NAME5_,
dtc0_.OPERATIONAL_AREAUID as OPERATI13_5_, dtc0_.PROFILEUID as PROFILEUID5_, dtc0_.RTUUID as RTUUID5_,
dtc0_.SECSUBSTATION as SECSUBS16_5_, dtc0_.TIMETABLEUID as TIMETAB17_5_, dtc0_.VERSION as VERSION5_
from DTC dtc0_ where dtc0_.TIMETABLEUID=? and dtc0_.DELETED=?

select period0_.PERIODUID as PERIODUID10_0_, period0_.END as END10_0_, period0_.START as START10_0_,
period0_.TIMETABLE as TIMETABLE10_0_, period0_.VERSION as VERSION10_0_ from PERIOD period0_ where
period0_.PERIODUID=?

update PERIOD set END=?, START=?, TIMETABLE=?, VERSION=? where PERIODUID=? and VERSION=?

select control0_.CONTROLUID as CONTROLUID16_0_, control0_.MODE as MODE16_0_, control0_.OFFSET as
OFFSET16_0_, control0_.RANK as RANK16_0_, control0_.PERIOD as PERIOD16_0_,
control0_.SPECIAL_DAY_SERVICE as SPECIAL6_16_0_, control0_.TARGET_STATE as TARGET7_16_0_,
control0_.TIME as TIME16_0_, control0_.VERSION as VERSION16_0_ from CONTROL control0_ where
control0_.CONTROLUID=?

update CONTROL set MODE=?, OFFSET=?, RANK=?, PERIOD=?, SPECIAL_DAY_SERVICE=?, TARGET_STATE=?, TIME=?,
VERSION=? where CONTROLUID=? and VERSION=?

select control0_.CONTROLUID as CONTROLUID16_0_, control0_.MODE as MODE16_0_, control0_.OFFSET as
OFFSET16_0_, control0_.RANK as RANK16_0_, control0_.PERIOD as PERIOD16_0_,
control0_.SPECIAL_DAY_SERVICE as SPECIAL6_16_0_, control0_.TARGET_STATE as TARGET7_16_0_,
control0_.TIME as TIME16_0_, control0_.VERSION as VERSION16_0_ from CONTROL control0_ where
control0_.CONTROLUID=?

update CONTROL set MODE=?, OFFSET=?, RANK=?, PERIOD=?, SPECIAL_DAY_SERVICE=?, TARGET_STATE=?, TIME=?,
VERSION=? where CONTROLUID=? and VERSION=?


-- Schedule Retrieval

select timetable0_.TIMETABLEUID as TIMETABL1_1_0_, timetable0_.NAME as NAME1_0_,
timetable0_.OPERATIONAL_AREA as OPERATIO3_1_0_, timetable0_.VERSION as VERSION1_0_ from TIMETABLE
timetable0_ where timetable0_.TIMETABLEUID=?

select period0_.PERIODUID as PERIODUID10_, period0_.END as END10_, period0_.START as START10_,
period0_.TIMETABLE as TIMETABLE10_, period0_.VERSION as VERSION10_ from PERIOD period0_ where
period0_.TIMETABLE=?
```

```
select control0_.CONTROLUID as CONTROLUID16_, control0_.MODE as MODE16_, control0_.OFFSET as OFFSET16_,
control0_.RANK as RANK16_, control0_.PERIOD as PERIOD16_, control0_.SPECIAL_DAY_SERVICE as
SPECIAL6_16_, control0_.TARGET_STATE as TARGET7_16_, control0_.TIME as TIME16_, control0_.VERSION as
VERSION16_ from CONTROL control0_ where control0_.PERIOD=?


-- User Modification

select user0_.USERUID as USERUID9_0_, user0_.CLIENT as CLIENT9_0_, user0_.DEATH_DATE as DEATH3_9_0_,
user0_.DELETED as DELETED9_0_, user0_.EMAIL as EMAIL9_0_, user0_.FAILED_LOGINS as FAILED6_9_0_,
user0_.LOCKED as LOCKED9_0_, user0_.LOCKED_DATE as LOCKED8_9_0_, user0_.LOGIN as LOGIN9_0_, user0_.NAME
as NAME9_0_, user0_.OPERATIONAL_AREA as OPERATI11_9_0_, user0_.PASSWORD as PASSWORD9_0_, user0_.SALT as
SALT9_0_, user0_.VERSION as VERSION9_0_ from USER user0_ where user0_.USERUID=?

update USER set CLIENT=?, DEATH_DATE=?, DELETED=?, EMAIL=?, FAILED_LOGINS=?, LOCKED=?, LOCKED_DATE=?,
LOGIN=?, NAME=?, OPERATIONAL_AREA=?, PASSWORD=?, SALT=?, VERSION=? where USERUID=? and VERSION=?


-- Schedule Deletion

delete from CONTROL where CONTROLUID=? and VERSION=?

delete from PERIOD where PERIODUID=? and VERSION=?

delete from TIMETABLE where TIMETABLEUID=? and VERSION=?


-- User Deletion

delete from USER where USERUID=? and VERSION=?


-- State Report

select servicesto0_.SERVICE_TIMERUID as SERVICE1_20_, servicesto0_.BEGIN_DATE as BEGIN2_20_,
servicesto0_.CLIENTEUID as CLIENTEUID20_, servicesto0_.DISTRICTUID as DISTRICT4_20_,
servicesto0_.DTCUID as DTCUID20_, servicesto0_.DTC_NAME as DTC6_20_, servicesto0_.END_DATE as END7_20_,
servicesto0_.MINUTES as MINUTES20_, servicesto0_.MUNICIPALITYUID as MUNICIPA9_20_,
servicesto0_.OPERATIONAL_AREAUID as OPERATI10_20_, servicesto0_.POWER as POWER20_, servicesto0_.RUNNING
as RUNNING20_, servicesto0_.SERVICEUID as SERVICEUID20_, servicesto0_.SERVICE_NAME as SERVICE14_20_,
servicesto0_.STATE as STATE20_, servicesto0_.VERSION as VERSION20_ from SERVICE_STOPWATCH servicesto0_
where servicesto0_.RUNNING=? limit ?


-- x10:

select dtcservice0_.DTC_SERVICEUID as DTC1_7_0_, dtcservice0_.BIRTH_DATE as BIRTH2_7_0_,
dtcservice0_.DEATH_DATE as DEATH3_7_0_, dtcservice0_.DELETED as DELETED7_0_, dtcservice0_.DTCUID as
DTCUID7_0_, dtcservice0_.OPMODE as OPMODE7_0_, dtcservice0_.NAME as NAME7_0_, dtcservice0_.STATE as
STATE7_0_, dtcservice0_.VERSION as VERSION7_0_ from DTC_SERVICE dtcservice0_ where
dtcservice0_.DTC_SERVICEUID=?


-- Auditing Report

select auditactio0_.AUDIT_ACTIONUID as AUDIT1_15_, auditactio0_.CONTEXT as CONTEXT15_,
auditactio0_.ACTION_DATE as ACTION3_15_, auditactio0_.LOGIN as LOGIN15_, auditactio0_.TEXT as TEXT15_,
auditactio0_.TYPE as TYPE15_, auditactio0_.USERUID as USERUID15_, auditactio0_.VERSION as VERSION15_
from AUDIT_ACTION auditactio0_ limit ?
```

To effectively run the validation activity, it's required a command line shell positioned at validation directory and issue the following command:

```
/home/tomcat/validation tomcat$ ./startValidation.sh <property file>
```

The resulting output is a `.csv` file, containing:

```
sep=
Statistics for 20 sessions
Test    Count   Average (ms)    Max (ms)Min (ms)Ok
Schedule Creation x100  2000    390     1248    247     true
User Creation x100      2000    302     495     204     true
```

```
Login x10          200     798     1304    419     true
Logout x10         200     328     579     239     true
Failed Login x10 200       1094    1718    511     true
Schedule Modification x100         2000    804     1356    354     true
Schedule Retrieval x100  2000      516     843     330     true
User Modification x100   2000      352     616     244     true
Schedule Deletion x100   2000      359     618     240     true
User Deletion x100       2000      298     416     206     true
State Report x100        2000      723     1041    507     true
Auditing Report x100     2000      349     552     234     true
```

In concrete, Integration_4 was conducted by executing:

1. Validate Step#1
    a. Stop all nodes:
        i. `[tclouds@SLRep#] ./stopAll.sh`
    b. Start all nodes skipping f (tolerated faults) nodes:
        i. `[tclouds@SLRep#] ./copyDBs.sh`
        ii. `[tclouds@SLRep#] ./startAll.sh`
    c. Execute validation process from each SLApp:
        i. `[tomcat@SLApp#] ./startValidation.sh Integration4-step1-5sessions.properties`
    d. Monitor execution from `/home/tomcat/validation/log/validation-plain.log`
    e. Collect result from `/home/tomcat/validation/Integration4-step1-5.csv`
    f. Execute validation process from each SLApp:
        i. `[tomcat@SLApp#] ./startValidation.sh Integration4-step1-10sessions.properties`
    g. Monitor execution from `/home/tomcat/validation/log/validation-plain.log`
    h. Collect result from `/home/tomcat/validation/Integration4-step1-10.csv`
    i. Execute validation process from each SLApp:
        i. `[tomcat@SLApp#] ./startValidation.sh Integration4-step1-20sessions.properties`
    j. Monitor execution from `/home/tomcat/validation/log/validation-plain.log`
    k. Collect result from `/home/tomcat/validation/Integration4-step1-20.csv`
2. Validate Step#2
    a. Stop all nodes
    b. Start all nodes
    c. Execute validation process from each SLApp:
        i. `[tomcat@SLApp#] ./startValidation.sh Integration4-step2-5sessions.properties`
    d. Monitor execution from `/home/tomcat/validation/log/validation-plain.log`
    e. Collect result from `/home/tomcat/validation/Integration4-step1-5.csv`
    f. Execute validation process from each SLApp:
        i. `[tomcat@SLApp#] ./startValidation.sh Integration4-step2-10sessions.properties`
    g. Monitor execution from `/home/tomcat/validation/log/validation-plain.log`
    h. Collect result from `/home/tomcat/validation/Integration4-step2-10.csv`
    i. Execute validation process from each SLApp:
        i. `[tomcat@SLApp#] ./startValidation.sh Integration4-step2-20sessions.properties`
    j. Monitor execution from `/home/tomcat/validation/log/validation-plain.log`
    k. Collect result from `/home/tomcat/validation/Integration4-step2-20.csv`
3. Validate Step#3
    a. Stop all nodes
    b. Start all nodes
    c. Execute validation process from each SLApp:
        i. `[tomcat@SLApp#] ./startValidation.sh Integration4-step3-5sessions.properties`
    d. Monitor execution from `/home/tomcat/validation/log/validation-plain.log`

e. Half way through the execution, stop one replica node. Start the node when complete…
f. Collect result from `/home/tomcat/validation/Integration4-step1-5.csv`
g. Execute validation process from each SLApp:
    i. `[tomcat@SLApp#] ./startValidation.sh Integration4-step3-10sessions.properties`
h. Monitor execution from `/home/tomcat/validation/log/validation-plain.log`
i. Half way through the execution, stop one replica node. Start the node when complete…
j. Collect result from `/home/tomcat/validation/Integration4-step2-10.csv`
k. Execute validation process from each SLApp:
    i. `[tomcat@SLApp#] ./startValidation.sh Integration4-step3-20sessions.properties`
l. Monitor execution from `/home/tomcat/validation/log/validation-plain.log`
m. Half way through the execution, stop one replica node. Start the node when complete…
n. Collect result from `/home/tomcat/validation/Integration4-step3-20.csv`

Merging all results we end up with:

			SLApp1			SLApp2		
		Actions \ Sessions	**5**	**10**	**20**	**5**	**10**	**20**
Step#1	**1**	Schedule Creation x100	OK	OK	OK	OK	OK	OK
	2	User Creation x100	OK	OK	OK	OK	OK	OK
	3	Login x10	OK	OK	OK	OK	OK	OK
	4	Logout x10	OK	OK	OK	OK	OK	OK
	5	Failed Login x10	OK	OK	OK	OK	OK	OK
	6	Schedule Modification x100	OK	OK	OK	OK	OK	OK
	7	Schedule Retrieval x100	OK	OK	OK	OK	OK	OK
	8	User Modification x100	OK	OK	OK	OK	OK	OK
	9	Schedule Deletion x100	OK	OK	OK	OK	OK	OK
	10	User Deletion x100	OK	OK	OK	OK	OK	OK
	11	State Report x100	OK	OK	OK	OK	OK	OK
	12	Auditing Report x100	OK	OK	OK	OK	OK	OK
Step#1	**1**	Schedule Creation x100	OK	OK	OK	OK	OK	OK
	2	User Creation x100	OK	OK	OK	OK	OK	OK
	3	Login x10	OK	OK	OK	OK	OK	OK
	4	Logout x10	OK	OK	OK	OK	OK	OK
	5	Failed Login x10	OK	OK	OK	OK	OK	OK
	6	Schedule Modification x100	OK	OK	OK	OK	OK	OK
	7	Schedule Retrieval x100	OK	OK	OK	OK	OK	OK
	8	User Modification x100	OK	OK	OK	OK	OK	OK
	9	Schedule Deletion x100	OK	OK	OK	OK	OK	OK
	10	User Deletion x100	OK	OK	OK	OK	OK	OK
	11	State Report x100	OK	OK	OK	OK	OK	OK
	12	Auditing Report x100	OK	OK	OK	OK	OK	OK

Step#1	**1**	Schedule Creation x100	OK	OK	OK	OK	OK	OK
	2	User Creation x100	OK	OK	OK	OK	OK	OK
	3	Login x10	OK	OK	OK	OK	OK	OK
	4	Logout x10	OK	OK	OK	OK	OK	OK
	5	Failed Login x10	OK	OK	OK	OK	OK	OK
	6	Schedule Modification x100	OK	OK	OK	OK	OK	OK
	7	Schedule Retrieval x100	OK	OK	OK	OK	OK	OK
	8	User Modification x100	OK	OK	OK	OK	OK	OK
	9	Schedule Deletion x100	OK	OK	OK	OK	OK	OK
	10	User Deletion x100	OK	OK	OK	OK	OK	OK
	11	State Report x100	OK	OK	OK	OK	OK	OK
	12	Auditing Report x100	OK	OK	OK	OK	OK	OK

Table 29 - final outcome of Integration_4 validation activity

3.2.2.4.3 Integration_6 validation activity execution

The full comparison was extended to include 3 alternative setups, summing up to:

- Baseline @TC, reference response times, from a dedicated H2 database hosted on the same Trusted Cloud as the Application server
- Baseline @Azure, reference response times, from a dedicated H2 database hosted on Azure commodity cloud
- Baseline @EC2, reference response times, from a dedicated H2 database hosted on EC2 commodity cloud
- SteelDB @TC, system response time with all database replicas hosted within the same Trusted Cloud as the Application server
- SteelDB @CoC, system response time with Figure 132 setup

Merging all results we end up with:

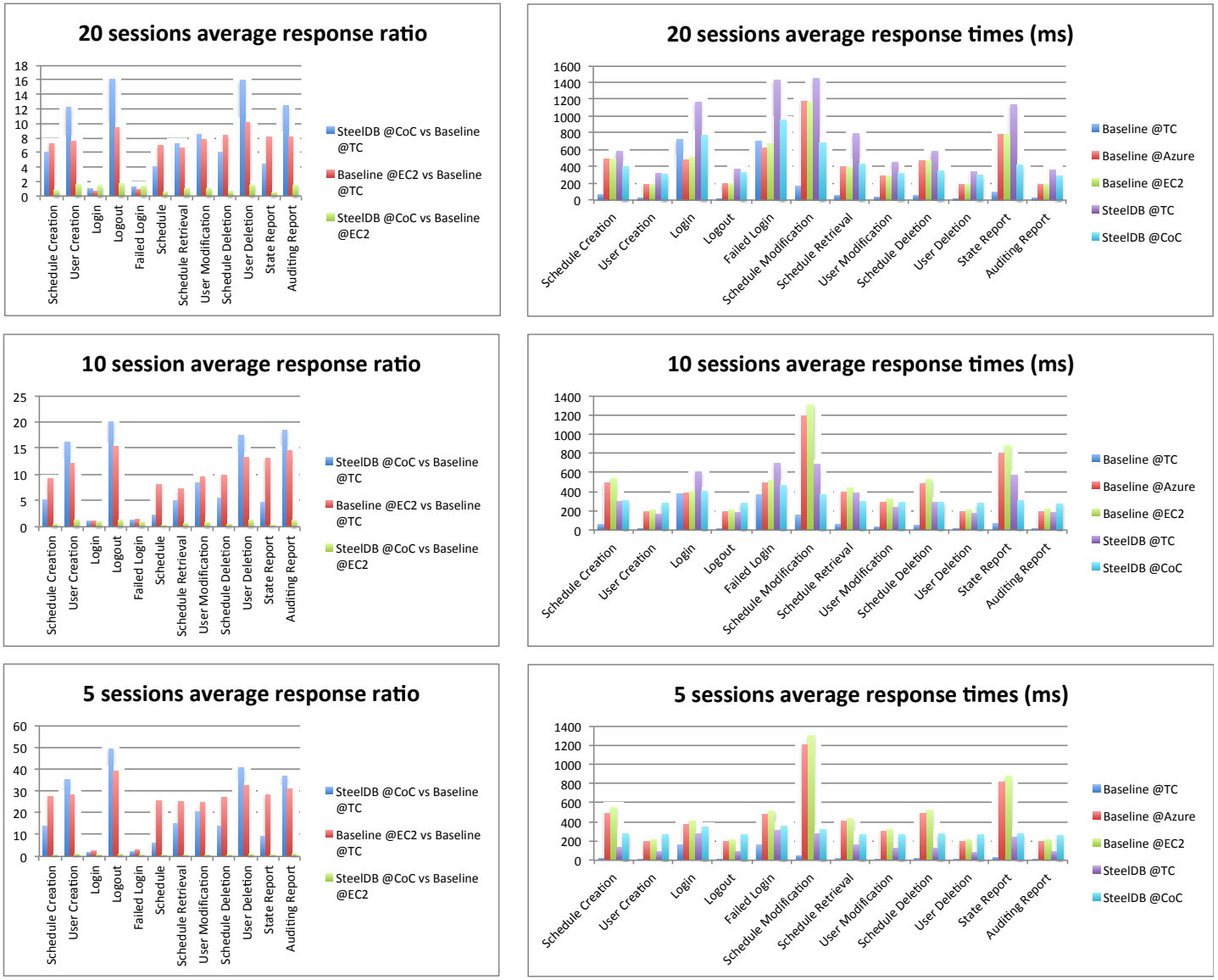


Figure 159 - Step#1 response times comparison charts

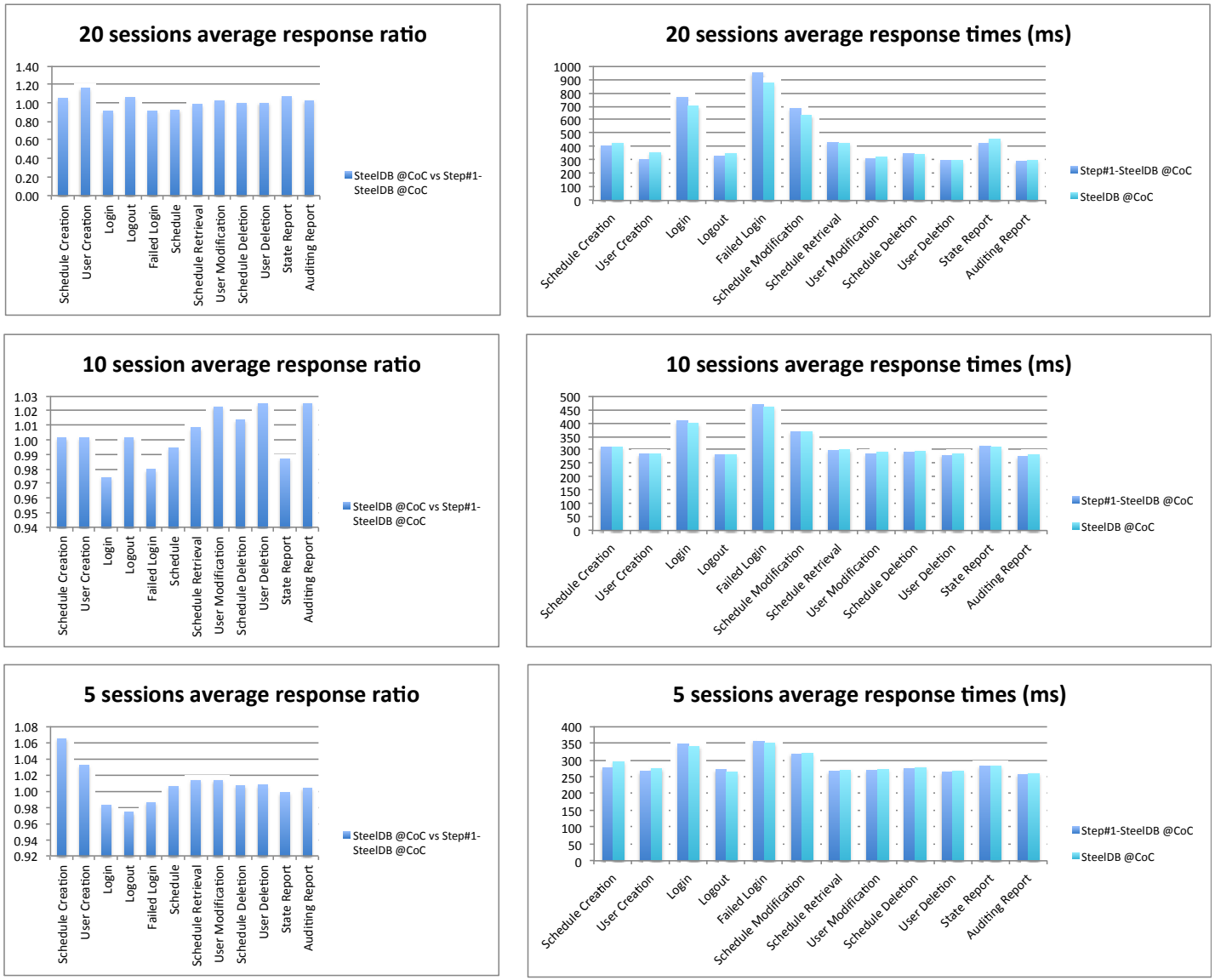


Figure 160 - Step#2 response times comparison charts

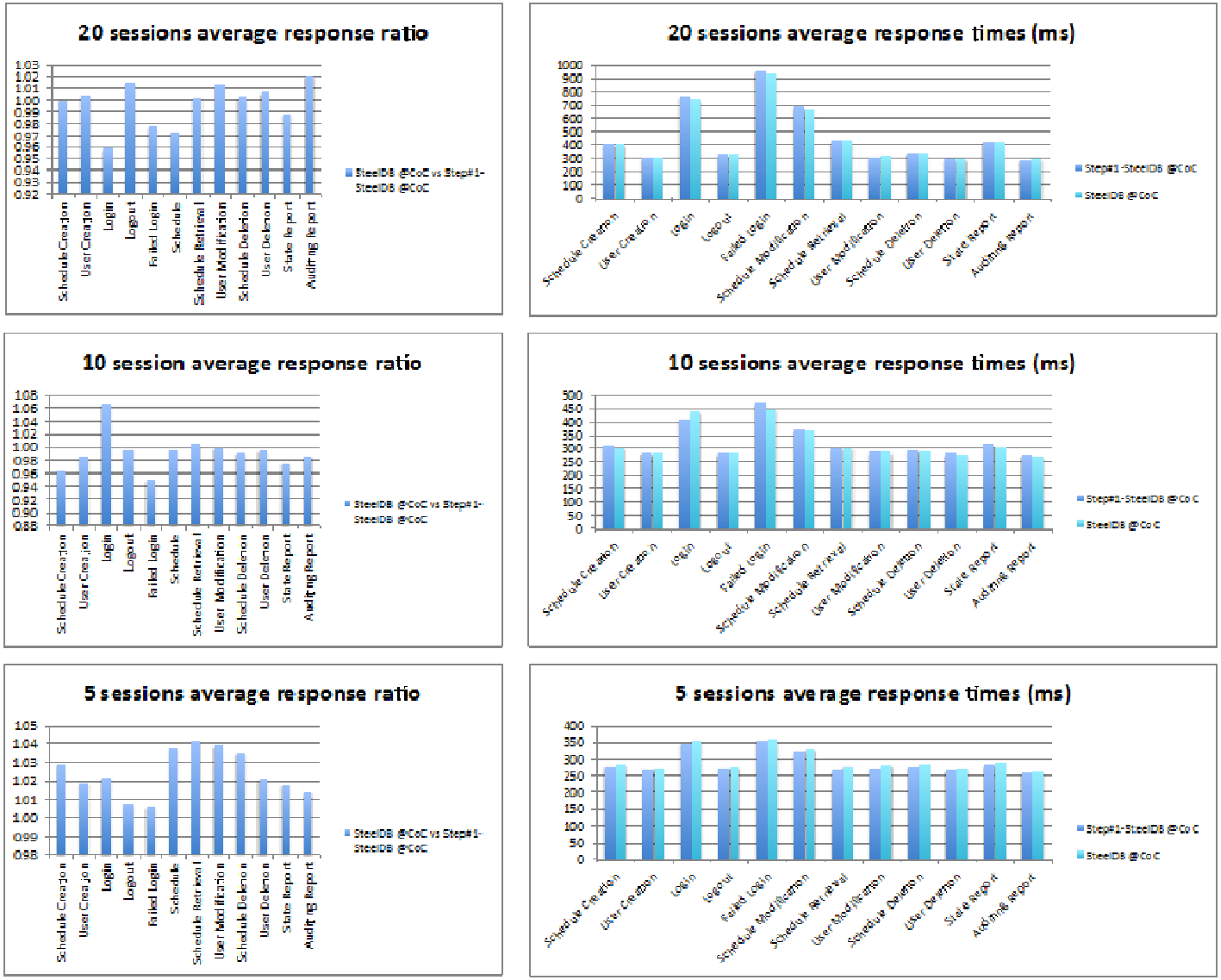


Figure 161 - Step#3 response times comparison charts

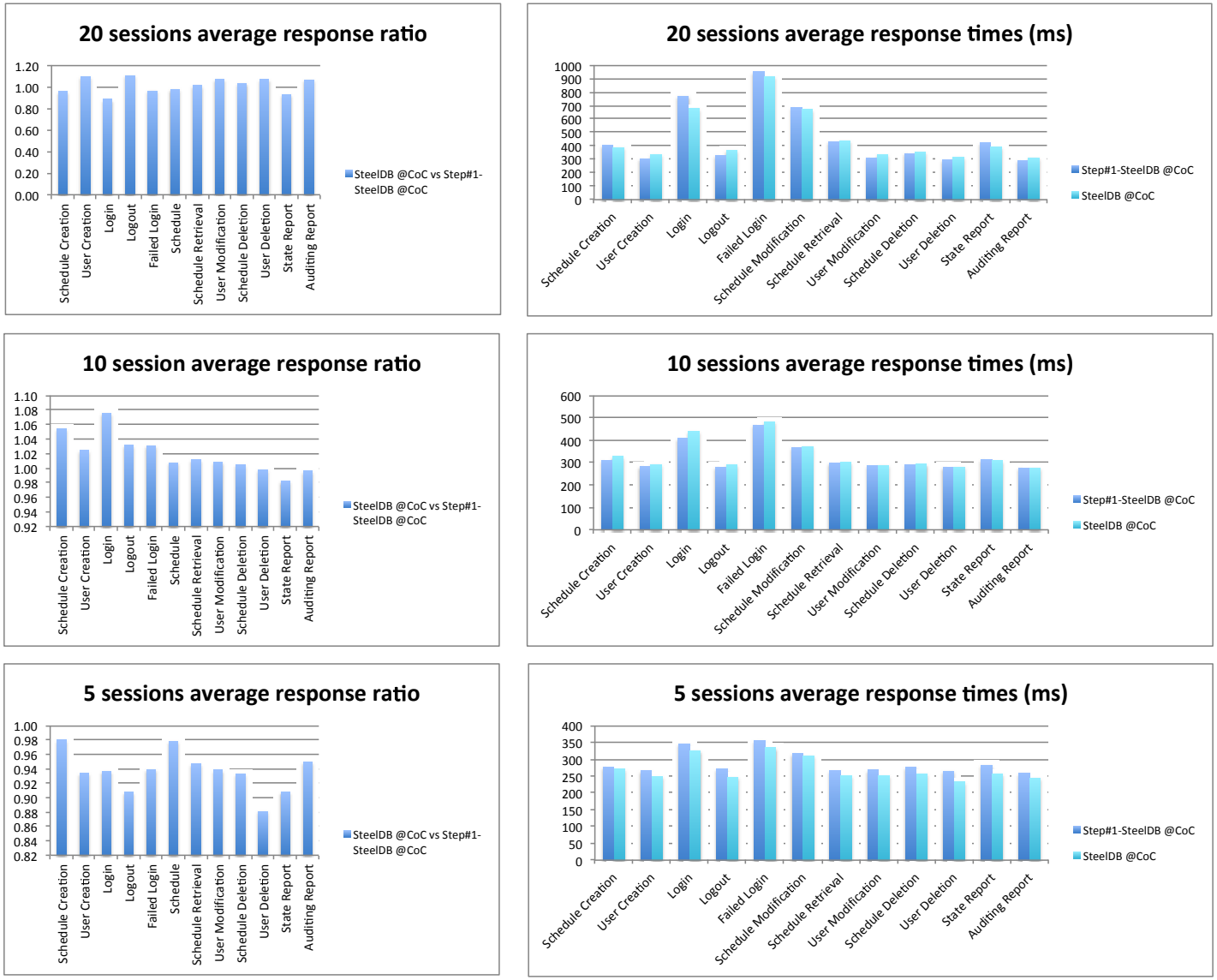


Figure 162 - Step#4 response times comparison charts

3.2.2.4.4 Conclusion

In this validation activity we stressed the resiliency of SteelDB, with full compliance to the requirement. We can mark Integration_4 and integration_6 as SUCCESSFULLY PASSED.

3.2.2.5 Integration_5

Activity ID	**Integration_5**
Activity type	Benchmarking
Activity description	Evaluate the infrastructure communications trustworthiness. 1- Confirm communications between nodes, enforce state of the art encryption, preventing any tapping
Acceptance Criteria	Step 1 is successful

3.2.2.5.1 Validation activity scenario

In order to properly execute this validation we have to setup the system in such a way all the main critical communication channels can be tested. The figure below describes the Validation scenario:

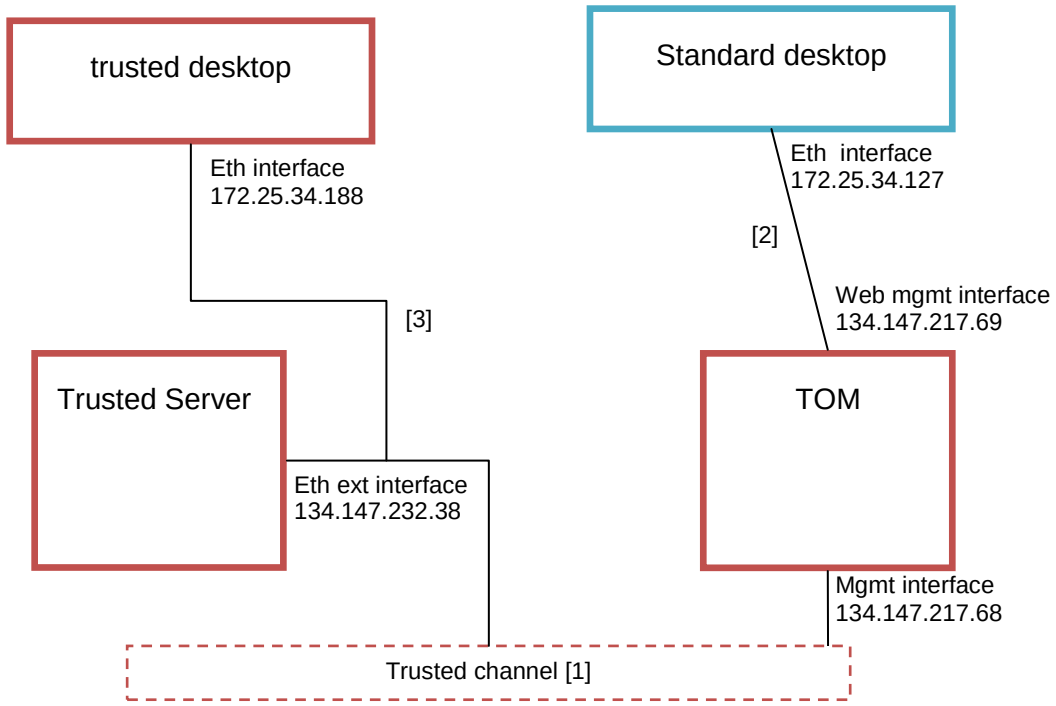


Figure 163 - Integration_5 scenario

In order to check the overall “trustworthiness” of the communications we have to address all those sensible parts in which data flows from one place to another. Within the Trustworthy Infrastrucutre we identified three main points: TrustedServer, Trusted Object Manager and The VPN connection to the nodes. Therfore, to perform the validation activity we splitted it in tree main parts:

1- Trusted channel communication from TS to TOM [1]

2- Communication to TOM [2]

3- VPN communication [3]

3.2.2.5.2 Validation activity execution

3.2.2.5.2.1 Step 1: TrustedChannel communication from TS to TOM:

As prerequisite of this step, we have to establish a communication between the Trusted Server (134.147.232.38) and the TrustedObjectsManager (134.147.217.68). The communication among this two peer pass through the TrustedChannel.In order to perform a dump of the tcp communication we decided to send a simple file among the two. For this validation activity we chose a new configuration file.

While sending the file we issued the following command on the Trusted Server:

```
tcpdump -i eth0 -s 0 -w TCdump.out -q '(tcp port 443)' and dst 134.147.217.68
```

The image below shows the output of tcpdump command:

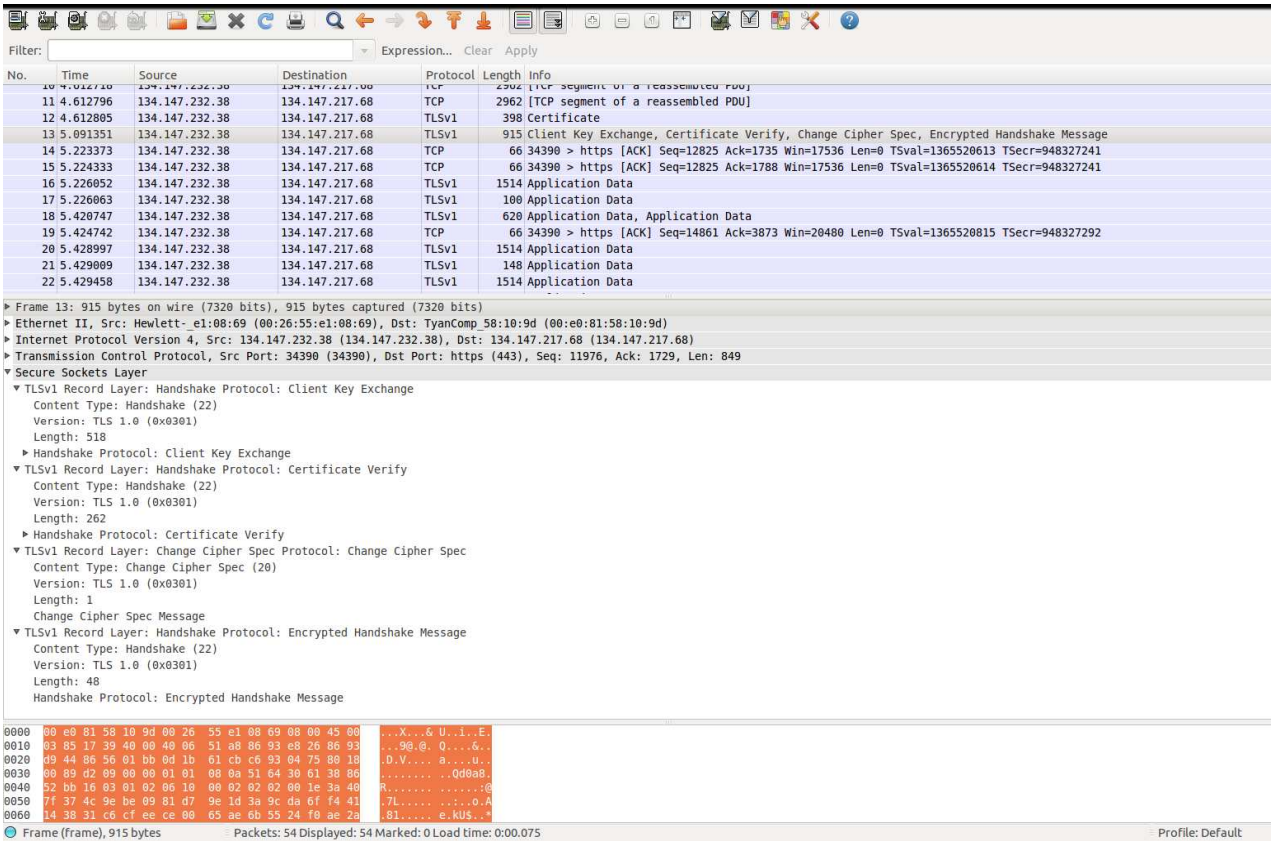


Figure 164 - Issuing tcpdump command and its output

As we can see from Figure 164 the two peers have established an encrypted communication (as we can see the exchange of the keys on the log file above). Please notice the ASCII dump of the first portion of the file that results as scrambled data, evidence of an encryption that has been made.

3.2.2.5.2.2 Step 2: Communication to TOM:

A connection to the TOM's web-management interface (134.147.217.69) from an arbitrary standard desktop system is established.

The network traffic is sniffed with the command line on the initiating desktop system via:

Command on legacy desktop system:

```
tcpdump -i eth0 -w output.dump -q '(tcp port 80) or (tcp port 443)' and dst 134.147.217.69 -n
```

The file output.dump is loaded into wireshark.

It can be seen, that the http-communication is encrypted with SSL /TLS1.0

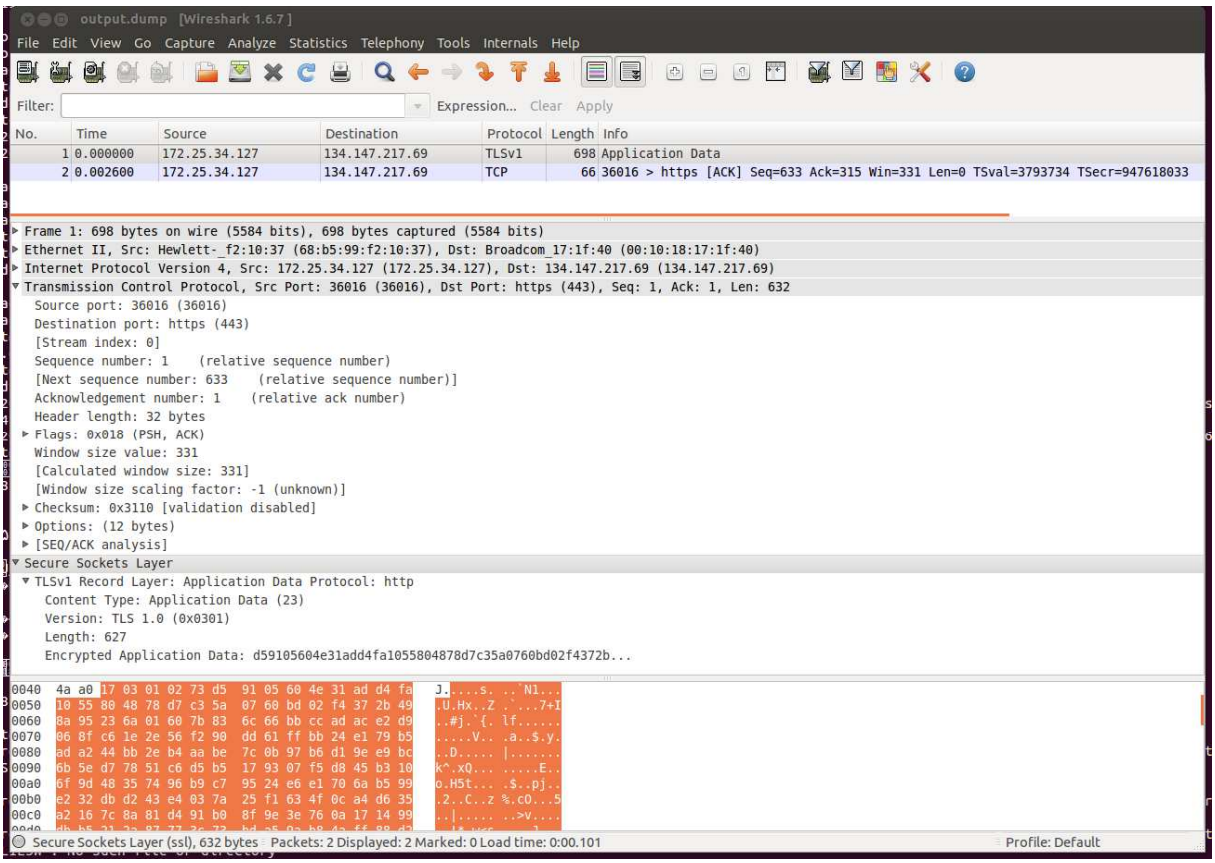


Figure 165 - issuing of tcpdump command and its output

3.2.2.5.2.3 Step 3: VPN communication

This step is described by [3] in Figure 163. In order to show the packets flowing through the connections we used the following command, issued from the Trusted Client:

```
Sirrix turaya # setkey –D
```

The command's output is shown below. Please remind that 134.147.232.38 is TrustedServer external interface' IP and 172.25.34.188 is TrustedDesktop external interface's IP.

```
134.147.232.38 172.25.34.188
        esp mode=tunnel spi=59116161(0x03860a81) reqid=2(0x00000002)
        E: aes-cbc  3de26596 890a860f 838a058b d83ba27c dffcabe9 7c5a44ad d235243f 618b30e1
        A: hmac-sha256  7c1758b9 df9b704f 28409a9c 2166a494 2449c224 18b32dd8 2579a90c 4ae2c492
        seq=0x00000000 replay=4 flags=0x00000000 state=mature
        created: Sep 20 17:41:24 2013    current: Sep 20 17:54:10 2013
        diff: 766(s)     hard: 86400(s)   soft: 69120(s)
        last: Sep 20 17:41:24 2013       hard: 0(s)       soft: 0(s)
        current: 21252(bytes)    hard: 0(bytes)   soft: 0(bytes)
        allocated: 253   hard: 0 soft: 0
        sadb_seq=1 pid=9931 refcnt=0
172.25.34.188 134.147.232.38
        esp mode=tunnel spi=244590219(0x0e94268b) reqid=1(0x00000001)
```

```
        E: aes-cbc  69746d78 6c8d7567 f43f4ffa 2e51daa8 7cdefa04 0e41f1bd cda8a57b e11fc121
        A: hmac-sha256  4091c7a3 a3e8791a c19a7a7d 746ea07a 1dab5ccf f6f26dd0 0e9d0167 9d49bbcb
        seq=0x00000000 replay=4 flags=0x00000000 state=mature
        created: Sep 20 17:41:24 2013    current: Sep 20 17:54:10 2013
        diff: 766(s)    hard: 86400(s)  soft: 69120(s)
        last: Sep 20 17:41:24 2013       hard: 0(s)      soft: 0(s)
        current: 21252(bytes)    hard: 0(bytes)  soft: 0(bytes)
        allocated: 253  hard: 0 soft: 0
        sadb_seq=0 pid=9931 refcnt=0
```

VPN tunnels are alive (mature) and encrypted with aes-cbc and for authentication hmac-sha256 is used.

```
Sirrix turaya # setkey -D
192.168.14.0/24[any] 192.168.29.0/24[any] any
        in prio high + 1073741324 ipsec
        esp/tunnel/172.25.34.188-134.147.232.38/unique:1
        created: Sep 20 17:39:10 2013  lastused: Sep 20 17:54:57 2013
        lifetime: 0(s) validtime: 0(s)
        spid=63832 seq=161 pid=11910
        refcnt=2

192.168.29.0/24[any] 192.168.14.0/24[any] any
        out prio high + 1073741324 ipsec
        esp/tunnel/134.147.232.38-172.25.34.188/unique:2
        created: Sep 20 17:39:11 2013  lastused: Sep 20 17:54:57 2013
        lifetime: 0(s) validtime: 0(s)
        spid=64353 seq=96 pid=11910
        refcnt=2

192.168.14.0/24[any] 192.168.29.0/24[any] any
        fwd prio high + 1073741324 ipsec
        esp/tunnel/172.25.34.188-134.147.232.38/require
        created: Sep 20 17:39:10 2013  lastused:
        lifetime: 0(s) validtime: 0(s)
        spid=63922 seq=150 pid=11910
        refcnt=2
```

These are the internal networks 192.168.14.0 and 192.168.29.0, which are connected via the above mentioned tunnels. These IP adress-ranges belong to the Administraton-TVD.

192.168.14.0 = AdministraionApp-VM on TrustedServer

192.168.29.0 = AdmistrationApp-VM on TrustedDesktop

3.2.2.5.3 Conclusions

The last output shows that the vpn-tunnels are up and alive. Only these specific communication channels are allowed and they are encrypted via aes-cbc.

Since the internal networks (192.158.14.0 and 192.168.29.0) are connected to different virtual-network devices on different physical machines (TS and TD), one can see that the VPN-connections between the class-C networks via the gateways (134.147.232.38 and 172.25.34.188) exists and are encrypted.

3.2.2.6 Trusted_O_1 and Truster_O_2 validation activities

Similarly for Integration_4 and Integration_6 these two activities share the same validation scenario, thus they will p be presented together.

Their aim is to assess the isolation among different tenants and among VMs.

Here below is shown the validation activity details:

Activity ID	**Trusted_O_1**
Activity type	Proof of concept
Activity description	TVD management from TOM 1- Create TVDs and deploy Smart Lighting System VMs in TVDs on TrustedServer. 2- check if corresponding PKI is properly deployed 3- Try to access from within the tvd to a resource outside the tvd boundaries 4- Try to access from outside to a resource into the tvd boundaries
Acceptance Criteria	Activity is passed if • point 2 show a properly deployment • Point 3 fails • Point 4 fails

Table 30 - Trusted_O_1 validation activity definition

Activity ID	**Trusted_O_2**
Activity type	Proof of concept
Activity description	Remote management access 1- Access remotely to TOM management system 2- Stop Smart Lighting VMs 3- Check that VM's are not accessible by non-authorized people, preventing confidentiality
Acceptance Criteria	Activity is passed if • point 3 pass

Table 31 - Trusted_O_2 validation activity definition

3.2.2.6.1 Validation activity details & setup

To get the grip of this validation activity we recall Figure 132 below:

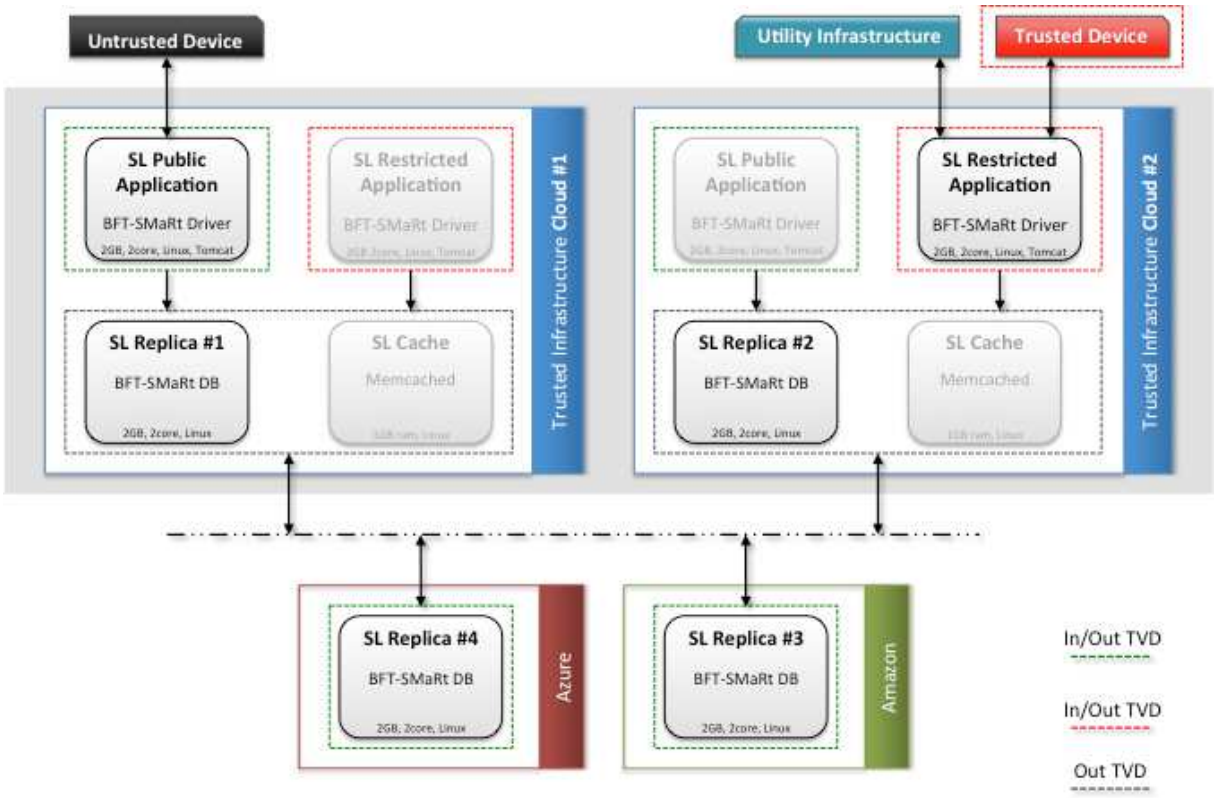


Figure 166 - SLS deployment scenario

What we are addressing are the TDVs (here above are represented as the Green, Red and Blue boxes) that has been setup among the virtual machines. The connection among the different VMs can be done through specific virtual Ethernet devices that connects the VM with the Trusted Server.

Here below are shown all the IP addresses used for each VM to refer to the trusted server:

Device	Secure channel	IP	TVD
SL-PublicCompartment	tun1832	192.168.28.1	Green
SL-AdministartionCompartment	tun1830	192.168.29.1	Red
SL-DBRep-1	tun1834	192.168.21.1	Blue
SL-DBRep-2	tun1836	192.168.22.1	
SL-DBRep-3	tun1838	192.168.23.1	
SL-DBRep-4	tun1840	192.168.24.1	
SL-MemCache	tun1872	192.168.30.1	

Table 32 - TVD definition and trusted channels

We can also inspect the server communication interfaces. By executing ifconfig command onto the Trusted Server we obtained the following output:

```
Sirrix turaya # ifconfig
eth0      Link encap:Ethernet  HWaddr 00:26:55:e1:08:69
          inet addr:134.147.232.38  Bcast:134.147.232.47  Mask:255.255.255.240
          inet6 addr: fe80::226:55ff:fee1:869/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:11348396 errors:0 dropped:0 overruns:0 frame:0
          TX packets:12708340 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:4599549524 (4.2 GiB)  TX bytes:5981750848 (5.5 GiB)
          Interrupt:41 Memory:fb7c0000-fb7e0000

tun1830   Link encap:Ethernet  HWaddr 56:1e:f9:e9:0e:ca
          inet addr:192.168.29.1  Bcast:192.168.29.255  Mask:255.255.255.0
          inet6 addr: fe80::541e:f9ff:fee9:eca/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:269 overruns:0 carrier:0
          collisions:0 txqueuelen:500
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

tun1832   Link encap:Ethernet  HWaddr be:5f:9a:8d:d5:b0
          inet addr:192.168.28.1  Bcast:192.168.28.255  Mask:255.255.255.0
          inet6 addr: fe80::bc5f:9aff:fe8d:d5b0/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:276 overruns:0 carrier:0
          collisions:0 txqueuelen:500
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

tun1834   Link encap:Ethernet  HWaddr 76:05:7d:61:f1:75
          inet addr:192.168.21.1  Bcast:192.168.21.255  Mask:255.255.255.0
          inet6 addr: fe80::7405:7dff:fe61:f175/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:295 overruns:0 carrier:0
          collisions:0 txqueuelen:500
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

tun1836   Link encap:Ethernet  HWaddr 1a:72:f5:61:63:4c
          inet addr:192.168.22.1  Bcast:192.168.22.255  Mask:255.255.255.0
          inet6 addr: fe80::1872:f5ff:fe61:634c/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:299 overruns:0 carrier:0
          collisions:0 txqueuelen:500
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

tun1838   Link encap:Ethernet  HWaddr 8a:6b:71:48:ca:77
          inet addr:192.168.23.1  Bcast:192.168.23.255  Mask:255.255.255.0
          inet6 addr: fe80::886b:71ff:fe48:ca77/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:275 overruns:0 carrier:0
          collisions:0 txqueuelen:500
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

tun1840   Link encap:Ethernet  HWaddr ee:75:0e:97:24:de
          inet addr:192.168.24.1  Bcast:192.168.24.255  Mask:255.255.255.0
          inet6 addr: fe80::ec75:eff:fe97:24de/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:290 overruns:0 carrier:0
          collisions:0 txqueuelen:500
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

tun1872   Link encap:Ethernet  HWaddr 4a:34:ae:37:f4:4b
          inet addr:192.168.30.1  Bcast:192.168.30.255  Mask:255.255.255.0
          inet6 addr: fe80::4834:aeff:fe37:f44b/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
```

```
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:0 errors:0 dropped:211 overruns:0 carrier:0
collisions:0 txqueuelen:500
RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

3.2.2.7 Trusted_O_1

In this validation activity we are going to address the TVD efficacy of isolate VMs within it.

3.2.2.7.1 Validation activity execution

3.2.2.7.2 Step1

By accessing to the Trusted Object Manager console, we can see the deployment details:

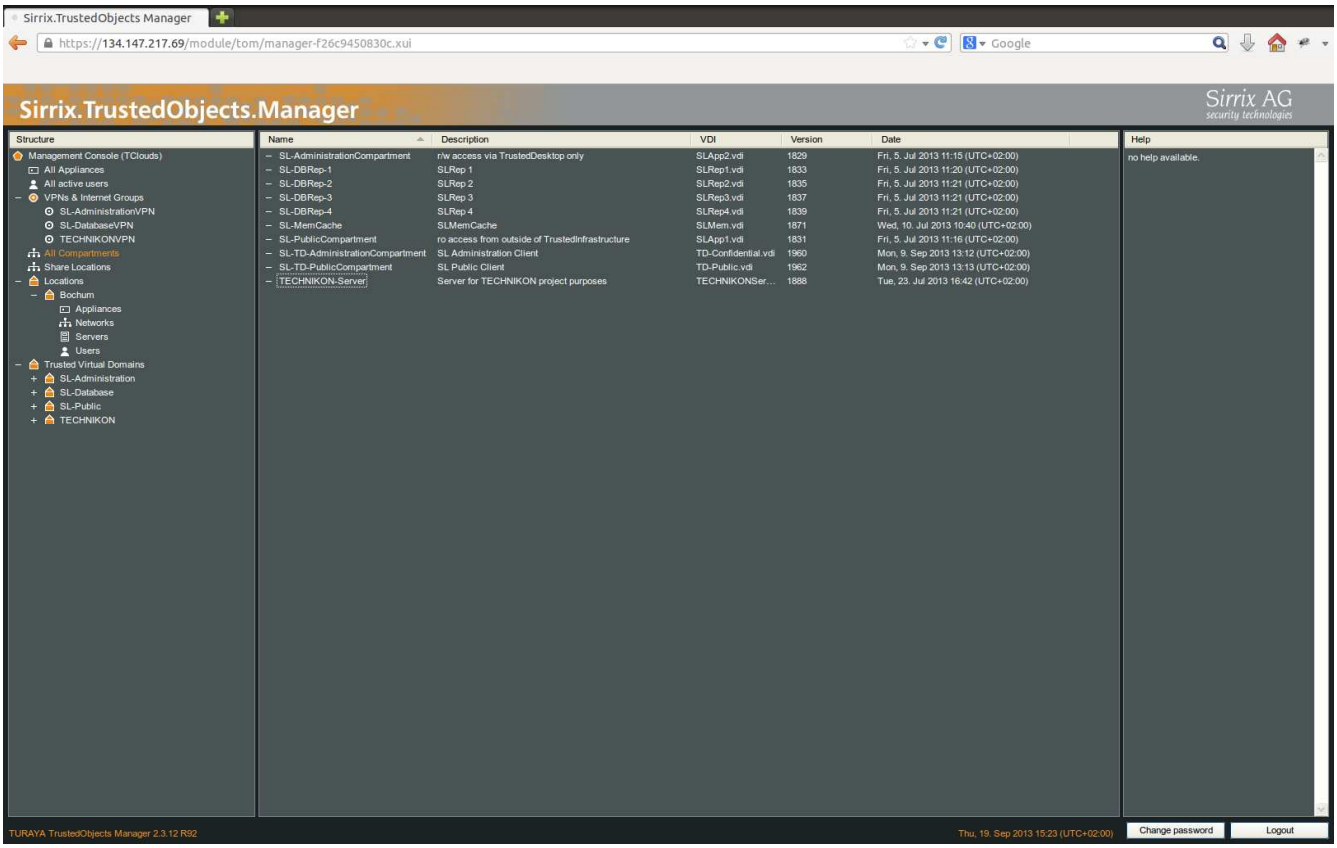


Figure 167- TOM interface showing the final deployment

As we can see the tenant (that owns the VMs, the TVDs and the Compartments are properly deployed

3.2.2.7.3 Step2

TOM ensures a proper deployment also for PKI (as seen in Figure 167). We can assess that keys are properly deployed otherwise the configuration would not be shown within the TOM

3.2.2.7.4 Step3

To check whether the TVD works we simply use the ping command (similarly as used for the TVD concept of Healthcare scenario). We accessed to the Red TVD (tun1830) and we tried to ping google.com domain with this result:

```
Sirrix turaya # ping -I tun1830 google.com
PING google.com (64.15.112.44) from 192.168.29.1 tun1830: 56(84) bytes of data.
^C
--- google.com ping statistics ---
6 packets transmitted, 0 received, 100% packet loss, time 5000ms
```

As seen is not possible to contact and route the ping request as the TVD compartment maintain within the TVD every ethernet packet.

3.2.2.7.5 Step4

Similarly to the previous step, we pinged the internal ip address by passing throught the external interface (eth0) please remind that the internal interface is bounded with tun1830 ethernet device.

```
PING 192.168.29.1 (192.168.29.1) from 134.147.232.38 eth0: 56(84) bytes of data.
^C
--- 192.168.29.1 ping statistics ---
6 packets transmitted, 0 received, 100% packet loss, time 5001ms
```

As seen, also in this case the connection fails

3.2.2.7.6 Conclusion

As shown in this validation activity, TVD concept allows tenants and VM isolation. We can conclude assessing Integration_5 validation activity and mark it as SUCCESSFULLY PASSED.

3.2.2.8 Trusted_O_2

3.2.2.8.1 Step1

This step is exactly as Step1 of Trusted_O_1 validation activity. Nothing more has to be added

3.2.2.8.2 Step2

From the Trusted infrastructure dashboard we are going to stop all the VMs. With the image below we can see that all compartments are actually stopped now:

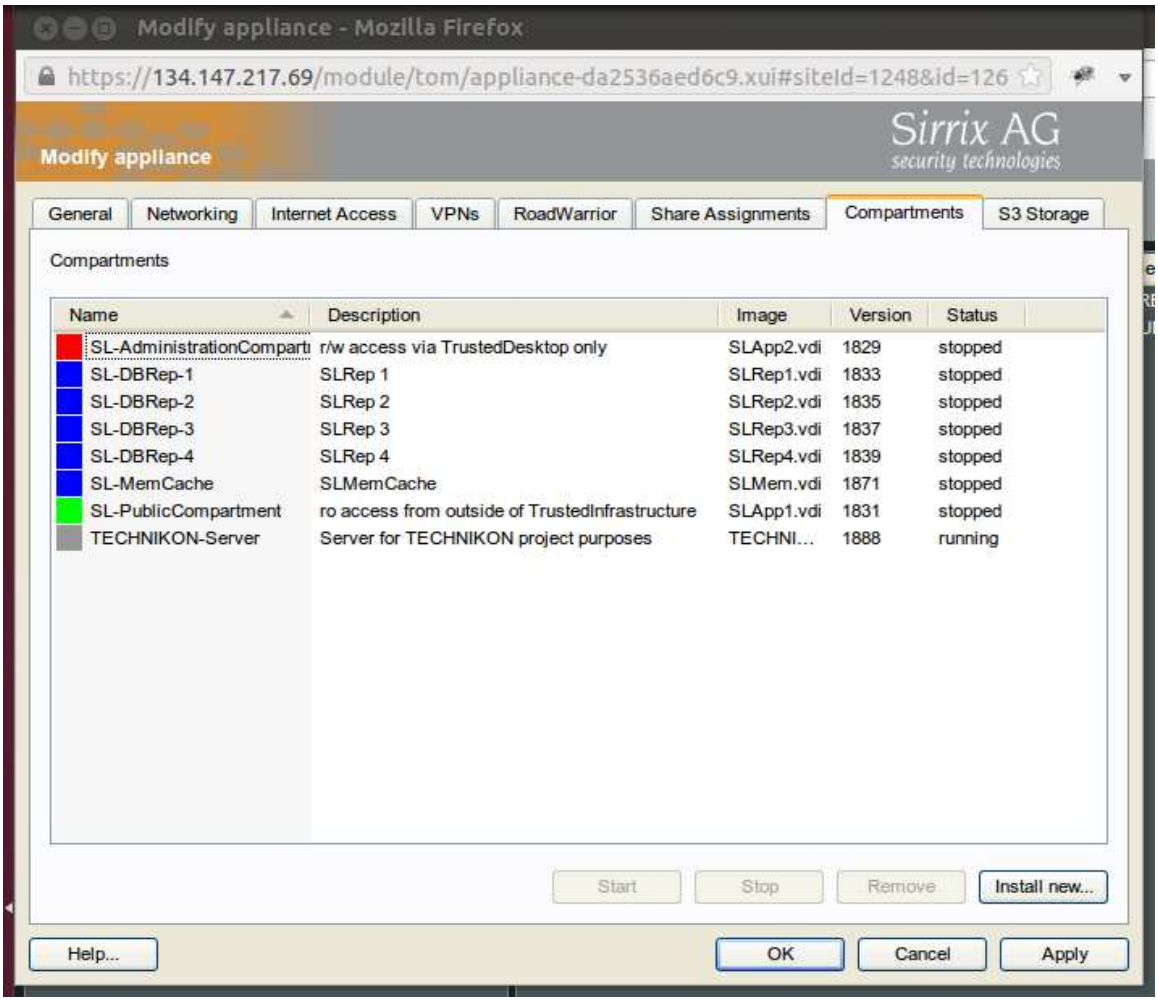


Figure 168- SLS VMs status

3.2.2.8.3 Step3

The first thing we are going to do is to check if the VMs hard disk are encrypted. In order to do this we will access to the Trusted Server as administrator and we will issue the command.

By accessing the VM folder we can see all the VM disks available by issuing the ls command:

```
Sirrix ~ # ls -l /vmdata/vmimages/

total 14572740
-rw-rw----         1         turaya         turaya         1357942784         Sep         19         15:49
0x0c3826d5e93a12351a773c8e948f3483cc45fea8220de0aba535256771917fa8.vdi
-rw-rw----         1         turaya         turaya         1678790656         Sep         19         15:49
0x68ab55ba633da2c2af7e5d95af1b1fd9549881e98095cf8adf4cf38fc9ac5a25.vdi
-rw-rw----         1         turaya         turaya         2095091712         Sep         19         15:49
0x6d87a8133d11dbf09a92ec752a4967da8338811f55e2e2b249e015a2f75b6605.vdi
-rw-rw----         1         turaya         turaya         1630572544         Sep         19         15:48
0x9acbd20a7f5f36595a253e2ed41bf543700e8b0bccf7f1cda686c7dd8c2ad3ff.vdi
-rw-rw----         1         turaya         turaya         2203095040         Sep         19         15:49
0xc81b8ac2019830315066dbfbf37c4e2cd89a0d60bf29cbea49c22d31acbdb5e3.vdi
-rw-rw----         1         turaya         turaya         2311098368         Sep         19         15:49
0xcd9d84d1f7521da77e968b92fe8fbafdbe76c2074717fce180a5b8a04aa298d6.vdi
-rw-rw----         1         turaya         turaya         1660981248         Sep         19         15:48
0xdba9c2947c243de716d3c0892ad1ccd317ab7a994a512b2f6846fff6fc07aeac.vdi
-rw-rw----         1         turaya         turaya         1970311168         Sep         19         15:49
0xe91b251bc7e43d2235e1fee332df73860bbffcc719424ad864548628bed35a0d.vdi
Sirrix ~ #
```

We chose one random file (they are all disks of SLS VMs) and we issued the file command. With this result:

```
Sirrix ~ # file /vmdata/vmimages/
0xe91b251bc7e43d2235e1fee332df73860bbffcc719424ad864548628bed35a0d.vdi
0x0c3826d5e93a12351a773c8e948f3483cc45fea8220de0aba535256771917fa8.vdi: encrypted data
Sirrix ~ #
```

As we can see from above, the disk results to be an encrypted file.

Continuing with the full-disk encryption. It has to be shown, that the system directory /vmdata/vmimages is encrypted

```
Sirrix ~ # mount
rootfs on / type rootfs (rw)
proc on /proc type proc (rw,nosuid,nodev,noexec,relatime)
sysfs on /sys type sysfs (rw,nosuid,nodev,noexec,relatime)
udev on /dev type tmpfs (rw,nosuid,relatime,size=10240k,mode=755)
devpts on /dev/pts type devpts (rw,relatime,gid=5,mode=620)
/dev/mapper/vgturaya-root on / type ext3 (rw,noatime,errors=remount-ro,user_xattr,acl,barrier=1,data=writeback)
rc-svcdir on /lib64/rc/init.d type tmpfs (rw,nosuid,nodev,noexec,relatime,size=1024k,mode=755)
securityfs on /sys/kernel/security type securityfs (rw,nosuid,nodev,noexec,relatime)
debugfs on /sys/kernel/debug type debugfs (rw,nosuid,nodev,noexec,relatime)
cgroup_root on /sys/fs/cgroup type tmpfs (rw,nosuid,nodev,noexec,relatime,size=10240k,mode=755)
cpuset on /sys/fs/cgroup/cpuset type cgroup (rw,nosuid,nodev,noexec,relatime,cpuset)
cpu on /sys/fs/cgroup/cpu type cgroup (rw,nosuid,nodev,noexec,relatime,cpu)
cpuacct on /sys/fs/cgroup/cpuacct type cgroup (rw,nosuid,nodev,noexec,relatime,cpuacct)
memory on /sys/fs/cgroup/memory type cgroup (rw,nosuid,nodev,noexec,relatime,memory)
devices on /sys/fs/cgroup/devices type cgroup (rw,nosuid,nodev,noexec,relatime,devices)
freezer on /sys/fs/cgroup/freezer type cgroup (rw,nosuid,nodev,noexec,relatime,freezer)
net_cls on /sys/fs/cgroup/net_cls type cgroup (rw,nosuid,nodev,noexec,relatime,net_cls)
blkio on /sys/fs/cgroup/blkio type cgroup (rw,nosuid,nodev,noexec,relatime,blkio)
fusectl on /sys/fs/fuse/connections type fusectl (rw,relatime)
shm on /dev/shm type tmpfs (rw,nosuid,nodev,noexec,relatime)
cachedir on /lib64/splash/cache type tmpfs (rw,nosuid,nodev,noexec,noatime,size=4096k,mode=755)
/dev/sda2 on /boot type ext3 (rw,noatime,errors=remount-ro,user_xattr,acl,barrier=1,data=writeback)
/dev/sda1 on /boot/grub type ext3 (rw,noatime,errors=remount-ro,user_xattr,acl,barrier=1,data=writeback)
none on /tmp type tmpfs (rw,relatime)
/dev/mapper/vgturaya-vmdata on /vmdata type ext3 (rw,noatime,errors=remount ro, user_xattr, acl, barrier=1,
data=writeback)
/dev/mapper/vgturaya-vmdata on /home type ext3 (rw,noatime,errors=remount-ro,user_xattr,acl,barrier=1,data=writeback)
/dev/mapper/vgturaya-config on /config type ext3 (rw,noatime,errors=remount-
ro,user_xattr,acl,barrier=1,data=writeback)
```

The directory `/dev/mapper/turaya-vmdata` is mounted to `/vmdata` and is a link to `/dev/mapper/vgturaya-vmdata` as shown here:

```
Sirrix ~ # ls -l /dev/vgturaya/vmdata
lrwxrwxrwx 1 root root 27 Sep  4 18:50 /dev/vgturaya/vmdata -> /dev/mapper/vgturaya-vmdata
```

In order to prove that /dev/vgturaya-vmdata is, we have a look to the logical volumes on the machine. It can be seen that /dev/vgturaya devices reside in a logical volume group named “vgturaya”

```
Sirrix ~ # lvdisplay
  --- Logical volume ---
  LV Name                /dev/vgturaya/swap
  VG Name                vgturaya
  LV UUID                eje0hQ-qToW-caRV-vWIM-PUBU-5yOm-Jf7KoK
  LV Write Access        read/write
```

```
  LV Status              available
  # open                 2
  LV Size                2.00 GiB
  Current LE             512
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           254:1

  --- Logical volume ---
  LV Name                /dev/vgturaya/root
  VG Name                vgturaya
  LV UUID                0ZpMed-Ixrp-gfXq-X9N1-YDSI-qmMj-y6JbRc
  LV Write Access        read/write
  LV Status              available
  # open                 1
  LV Size                10.00 GiB
  Current LE             2560
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           254:2

  --- Logical volume ---
  LV Name                /dev/vgturaya/config
  VG Name                vgturaya
  LV UUID                x4hn3u-xDZ7-khQP-cPB1-cVCq-oqM2-75cG5O
  LV Write Access        read/write
  LV Status              available
  # open                 1
  LV Size                10.00 GiB
  Current LE             2560
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           254:3

  --- Logical volume ---
  LV Name                /dev/vgturaya/vmdata
  VG Name                vgturaya
  LV UUID                m26n8y-uoF5-2d4D-6ydH-xgSR-0931-HGg5ex
  LV Write Access        read/write
  LV Status              available
  # open                 1
  LV Size                909.36 GiB
  Current LE             232795
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           254:4
```

The physical volume /dev/disk/by.uuid/ aDDp1q-L1GV-TiM8-BvRR-mVMI-xntI-N1gbxt" is where the logical volumes from above reside in:

```
Sirrix ~ # pvdisplay
  --- Physical volume ---
  PV Name               /dev/disk/by-uuid/aDDp1q-L1GV-TiM8-BvRR-mVMI-xntI-N1gbxt
  VG Name               vgturaya
  PV Size               931.36 GiB / not usable 0
  Allocatable           yes (but full)
  PE Size               4.00 MiB
  Total PE              238427
  Free PE               0
  Allocated PE          238427
  PV UUID               aDDp1q-L1GV-TiM8-BvRR-mVMI-xntI-N1gbxt
```

Here we can see that the physical volume from above points to /dev/dm-0

```
Sirrix ~ # ls -l /dev/disk/by-uuid/aDDp1q-L1GV-TiM8-BvRR-mVMI-xntI-N1gbxt
lrwxrwxrwx 1 root root 10 Sep  4 18:50 /dev/disk/by-uuid/aDDp1q-L1GV-TiM8-BvRR-mVMI-xntI-N1gbxt -> ../../dm-0
```

This was inspected with "dmsetup" in order to see what /dev/dm-0 is:

```
Sirrix ~ # dmsetup info /dev/dm-0
Name:              crturaya
State:             ACTIVE
Read Ahead:        256
Tables present:    LIVE
Open count:        4
Event number:      0
```

```
Major, minor:      254, 0
Number of targets: 1
UUID: CRYPT-LUKS1-a77eea85c0c649beb6097a5e17c803ec-crturaya
```

/dev/dm-0 is a crypted LUKS container. This can be seen by inspecting the logical volume “crturaya” with “cryptsetup”

```
Sirrix ~ # cryptsetup status crturaya
/dev/mapper/crturaya is active:
  cipher:  aes-xts-plain
  keysize: 256 bits
  device:  /dev/srxcrypt
  offset:  4096 sectors
  size:    1953196032 sectors
  mode:    read/write
Sirrix ~ #
```

Therefore the logical volume “srturaya” is an encrypted LUKS container, residing on /dev/srxcrypt, which itself is just a link to the block device partition /dev/sda3

```
Sirrix ~ #  ls -l /dev/srxcrypt
lrwxrwxrwx 1 root root 9 Sep  4 18:50 /dev/srxcrypt -> /dev/sda3
Sirrix ~ #
```

3.2.2.8.4 Conclusion

Within this activity it has been shows that the whole filesystem residing on the machines’ hard disk is encrypted. That includes the directory containing the virtual machine images

3.2.2.9 Trusted_O_3

By reviewing all the validation activities we have noticed that this validation activity has to be, in fact, performed exactly as Trusted_S_2, since it’s assessing the same concept and acceptance criteria corresponds.

Thus we decided to drop it. Please refer to Trusted_S_2 to know its results.

Activity ID	**Trusted_O_3**
Activity type	Proof of concept
Activity description	Operate TrustedServer via TOM and TrustedChannel 1- Manipulate data on server and try to boot. This should fail
Acceptance Criteria	Activity is passed if • Activity at point 1 fails

Table 33 - Trusted_O_3 validation activity description

REMOVED (same as Trusted_S_2)

3.2.2.10 Trusted_S_1

Activity ID	**Trusted_S_1**
Activity type	Proof of concept
Activity description	Inspect that there is no root account on TrustedServer Inspect an TrustedServer and ensure that there is no active root account where an administrator could log in.
Acceptance Criteria	Activity is passed if • There is no active root account at administrator login point

Table 34 - Trusted_S_1 validation activity description

Validation activity execution

`/etc/passwd` of the TrustedServer contains a root-account, but the user cannot login

```
Sirrix ~ # cat /etc/passwd
root:x:0:0:root:/root:/sbin/nologin
bin:x:1:1:bin:/bin:/bin/false
daemon:x:2:2:daemon:/sbin:/bin/false
lp:x:4:7:lp:/var/spool/lpd:/bin/false
sync:x:5:0:sync:/sbin:/bin/sync
halt:x:7:0:halt:/sbin:/sbin/halt
uucp:x:10:14:uucp:/var/spool/uucp:/bin/false
nobody:x:65534:65534:nobody:/var/empty:/bin/false
dhcp:x:101:247:added by portage for dhcp:/var/lib/dhcp:/sbin/nologin
pcscd:x:102:245:added by portage for pcsc-lite:/var/run/pcscd:/sbin/nologin
ldap:x:439:439:added by portage for openldap:/usr/lib64/openldap:/sbin/nologin
messagebus:x:103:243:added by portage for dbus:/dev/null:/sbin/nologin
avahi:x:104:241:added by portage for avahi:/dev/null:/sbin/nologin
avahi-autoipd:x:105:240:added by portage for avahi:/dev/null:/sbin/nologin
turaya:x:1005:1005:added by portage for TrustedServer0:/home/turaya:/sbin/nologin
```

This /etc/passwd is taken as a template for the security-kernel in a production environment. It is copied to the Turaya-system during the production of a TrustedServer. /etc/securetty is empty, so that no user is ever presented a login screen and cannot login from anywhere

```
Sirrix ~ # cat /etc/securetty
# /etc/securetty: list of terminals on which root is allowed to login.
# See securetty(5) and login(1).

Also the sudoers-file does not contain anything, so it is impossible to become a superuser.

Sirrix ~ # cat /etc/sudoers
## sudoers file.

Sirrix ~ #
```

3.2.2.10.1 Conclusion

This validation activity proves, though a root user exists, there is no possibilty to login via a shell on the one hand, and that it is also impossible to become a super on the other.

3.2.2.11 Trusted_S_2

Activity ID	**Trusted_S_2**
Activity type	Proof of concept
Activity description	Test Secure Boot of Trusted Server • Boot integer server, this should work properly. • Manipulate data on server and try to boot. This should fail.
Acceptance Criteria	Activity is passed if • Activity at point 2 fails

Table 35 - Trusted_S_2 validation activity description

In this validation activity we are going to prove the integrity check capabilities of TrustedServer. We are simulating an integrity-attack in a form, where some preconditions have to meet, which are not given in reality:

Preconditions:

- An attacker has physical access to TrustedServer
  - Should be prohibited, by the company's security policies
- An attacker is able to boot the machine from an external medium
  - This is prevented by a BIOS password, which should be set by a trustworthy administrator so that the BIOS settings cannot be changed.

An attack on TrustedServer is simulated by changing the original and therefore valid (in terms of PCR [3]values) initramfs to a modified initraramfsX, and excahanging the path within /tgrub/menu.lst.

Since the valid configuration is sealed against the original PCR values, the replacement of the initramfs is detected, already during boot-phase, so that initramfsX will not be loaded at all.

This results in the following output:

```
Failed to open the fbcon_decor control device.
>> Loading modules
   :: Scanning for scsi_transport_fc...scsi_tgt, scsi_transport_fc loaded.
   :: Scanning for scsi_wait_scan...scsi_wait_scan loaded.
   :: Scanning for dm-mirror...dm-log, dm-region-hash, dm-mirror loaded.
   :: Scanning for dm-snapshot, dm-snapshot loaded.
   :: Scanning for scsi_transport_iscsi...scsi_transport_iscsi loaded.
>> Hint: Use parameter scandelay[=seconds] if you need waiting here
>> Activating mdev
>> Unlocking Volumes ...
>> Found valid LUKS device /dev/sda3
>> Linking /dev/srxcrypt -> /dev/sda3
>> Linking /dev/srxboot -> /dev/sda1
>> Linking /dev/srxdisk -> /dev/sda
Extend successful, new value of PCR 15: 0x601cb5cccba1a2f5d35dcda452f4711dbe9a0283
Success!
ls: /mnt/keyfile.sealed: No such file or directory
>> Could not find keyfile.sealed on /dev/srxboot, searching ...
Linking /dev/srxboot -> /dev/sda2
/mnt/keyfile.sealed
microtssEngine: Exception thrown:(Turaya/HDDEncryption1/SealBundle.cxx:161) Unsealing failure (TPM_WRONGPCRVAL)
!! Failed to unseal LUKS key material!
>> Scanning for and activating Volume Groups
  No volume groups found
unmount: can't umount /mnt/config: Invalid argument
>> Invalidating PCRs ...
>> Determining root device...
!! Block device /dev/mapper/vgturaya-ROOT is not a valid root device...
!! Could not find the root block device in .
Please specify another value or: press Enter for the same, type "shell" for a shell, or "q" to skip...
root block device() :: shell
To leave and try again just press <Ctrl>+D
!! An error occured.
!! Rebooting in 10 seconds ...
```

[3] Platform Configuration Register. The TPM contains several PCRs (Platform Configuration Registers) that allow a secure storage of security relevant metrics. Source: wikipedia

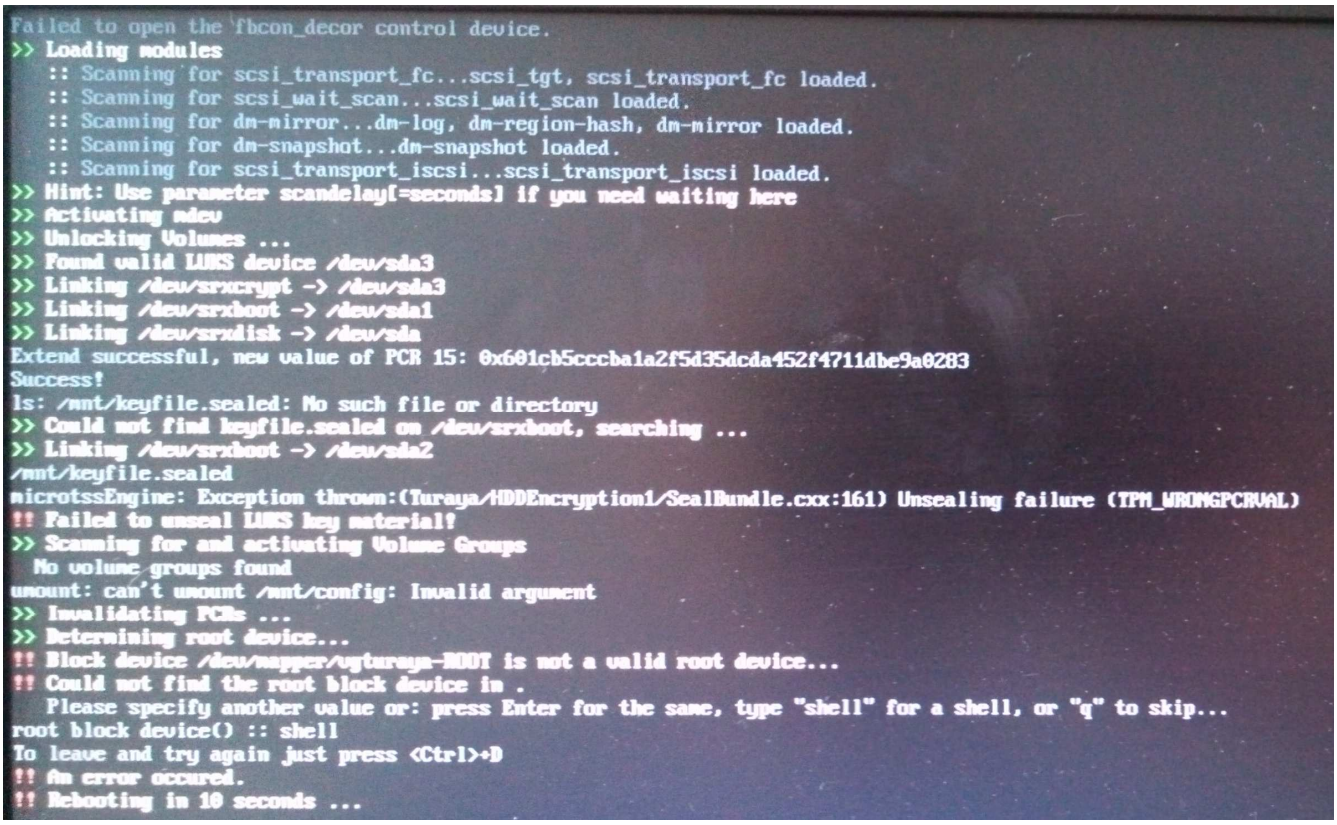


Figure 169 - output of command

3.2.2.11.1 Conclusion

As one can see, the attempt to unlock the key for the sealed hard disk fails, because of different PCR-values. Unless the TPM-stored PCR-values match the measured hashsums of the initramfs, the system will not boot up further.

3.2.2.12 Trusted_S_3

Activity ID	**Trusted_S_3**
Activity type	Proof of concept
Activity description	Test local disks of TrustedServer 1- Run TrustedServer according to the use cases on D2.4.2 chapter 4.4.1.2 2- Check if data on local disk is properly encrypted
Acceptance Criteria	Activity is passed if • Activity at point 2 shows encrypted data
References Documents:	(Deliverable D2.4.2, 2012)

Table 36 - Trusted_S_3 validation activity description

Trusted_O_2 shows, that the whole disc is encrypted, so that this activity is the same. This does not change anything on the activity's internal sense, if we take the use cases of D2.4.2 into account. It is and stays the same

3.2.2.13 Trusted_C_1

Activity ID	**Trusted_C_1**
Activity type	Proof of concept
Activity description	1- Establish trusted channel with Smart Lighting System VMs and check that data is properly encrypted
Acceptance Criteria	Activity is passed if • Activity at point 1 demonstrate the encryption of transferred data

Table 37 - Trusted_C_1 validation activity description

Because the TrustedChannel does not exist in this case.

The TC is established between the nodes (TrustedDesktop, TOM, TrustedServer) "only" and NOT between the VMs themselves, which are running "on" the nodes.

The TC exists for the purpose to deliver a valid configuration to the nodes (for the network- and VPN settings, VM settings, Remote Attestation). It is not intended to transfer data-into-or-from-the VMs to somewhere else. Vice versa, when the config has arrived on a node, the TrustedChannel can be shut down, and has nothing more to do, except in the case a new configuration exists. Then the game starts again.

The communication between the VMs relies (after a proper config deployment) only on the ipsec secured VPN communication.

For this reason we decided to drop the following validation activity since prerequisites are not satisfied and there is any trusted channel between the Trusted Infrastructure and a tenant's VM.

3.2.2.14 Trusted_C_2

Activity ID	**Trusted_C_2**
Activity type	Proof of concept
Activity description	1- establish trusted channel with authentic communication partners of Smart Lighting System appliance, check if communication works 2- try to establish channel with non-authentic partner, check that communication is refused
Acceptance Criteria	Activity is passed if • Activity at point 1 demonstrate the encryption of transferred data + the correct communication among VMs • Activity at point 2 fails

Table 38 - Trusted_C_2 validation activity description

Step 1:

Already shown in Trusted_O_1

Step 2:

A non-authentic partner is an arbitrary desktop system, i.e. without the Turaya.SecurityKernel installed, or a Turaya-system, which is not registered at the TOM.

In any case, the non-authentic partner will not be able to establish a TrustedChannel connecttion to the TOM in order to get its configuration. Therefore the legacy system, will not be aware of the i.e internal VPN, not even the Class-C network (which is created by the VMs running on TS).

Since this configuration is the starting point for every un-configured appliance connected to TrustedInfrastructures, every legacy system will not receive a configuration. Thus, the PKI will not be deployed at all, the appliance is not aware of the i.e. internal VPN, not even of the Class-C network (which is created by the VMs running on TS).

Already the TOM refuses connections from devices, which are not registered and attestated at the same time. Therefore a connection to the internal network and/or VPN of TS is not possible at all.

3.2.2.14.1 Conclusion

All Trusted Infrastructure validation activities were executed and conformed to expectations.

3.2.2.15 BFT-SMaRt

3.2.2.15.1 BFT-SMaRt_1

Activity ID	**BFT-SMaRt_1**
Activity type	Proof of concept
Activity description	1- Deploy Smart Lighting System VMs into TClouds infrastructure 2- Insert data in a key value store and query the server to guarantee that the data has been correctly saved
Acceptance Criteria	Point 2 is successful.

Table 39 - BFT-SMaRt_1 validation activity description

Test setup and execution

The test was executed using a demo application of BFT-SMaRt called BFTMap.

The application is a console where the user can insert and query data in tables. The data is replicated and queried using BFT-Smart.

Figure 170 shows the log of the client execution and Figure 171 shows the moment were a replica in Amazon EC2 cloud was interrupted. The test executed the following steps:

- The client creates a table called TClouds1;
- The client insert an entry in the table with the key 555 and value FFCUL;
- The replica in Amazon EC2 cloud was interrupted;
- The client queries the table Tclouds1 with the key 555 and gets the expected result, FFCUL.

```
tomcat@localhost:~/validation_fcul
[tomcat@localhost validation_fcul]$
[tomcat@localhost validation_fcul]$ ./startMapClient.sh
Connecting to replica 0 at /192.168.21.2:11000
Channel connected
Connecting to replica 1 at /192.168.22.2:11000
Channel connected
Connecting to replica 2 at tclouds-medium.cloudapp.net/137.135.219.145:11000
Channel connected
Connecting to replica 3 at ec2-54-229-177-243.eu-west-1.compute.amazonaws.com/54.229.177.
243:11000
Channel connected
select a command : 1. CREATE A NEW TABLE OF TABLES
select a command : 2. REMOVE AN EXISTING TABLE
select a command : 3. GET THE SIZE OF THE TABLE OF TABLES
select a command : 4. PUT VALUES INTO A TABLE
select a command : 5. GET VALUES FROM A TABLE
select a command : 6. GET THE SIZE OF A TABLE
select a command : 7. REMOVE AN EXISTING TABLE
select a command : 11. EXIT
1
Enter the HashMap name: TClouds1
select a command : 1. CREATE A NEW TABLE OF TABLES
select a command : 2. REMOVE AN EXISTING TABLE
select a command : 3. GET THE SIZE OF THE TABLE OF TABLES
select a command : 4. PUT VALUES INTO A TABLE
select a command : 5. GET VALUES FROM A TABLE
select a command : 6. GET THE SIZE OF A TABLE
select a command : 7. REMOVE AN EXISTING TABLE
select a command : 11. EXIT
4
Execute put function
Enter the valid table name in which you want to insert data: TClouds1
Enter a numeric key for the new record in the range 0 to 9999: 555
Enter the value for the new record: FFCUL
select a command : 1. CREATE A NEW TABLE OF TABLES
select a command : 2. REMOVE AN EXISTING TABLE
select a command : 3. GET THE SIZE OF THE TABLE OF TABLES
select a command : 4. PUT VALUES INTO A TABLE
select a command : 5. GET VALUES FROM A TABLE
select a command : 6. GET THE SIZE OF A TABLE
select a command : 7. REMOVE AN EXISTING TABLE
select a command : 11. EXIT
5
Execute get function
Enter the valid table name from which you want to get the values: TClouds1
Enter the key: 555
The value received from GET is: FFCUL
```

Figure 170 - Client execution of the BFTMap application

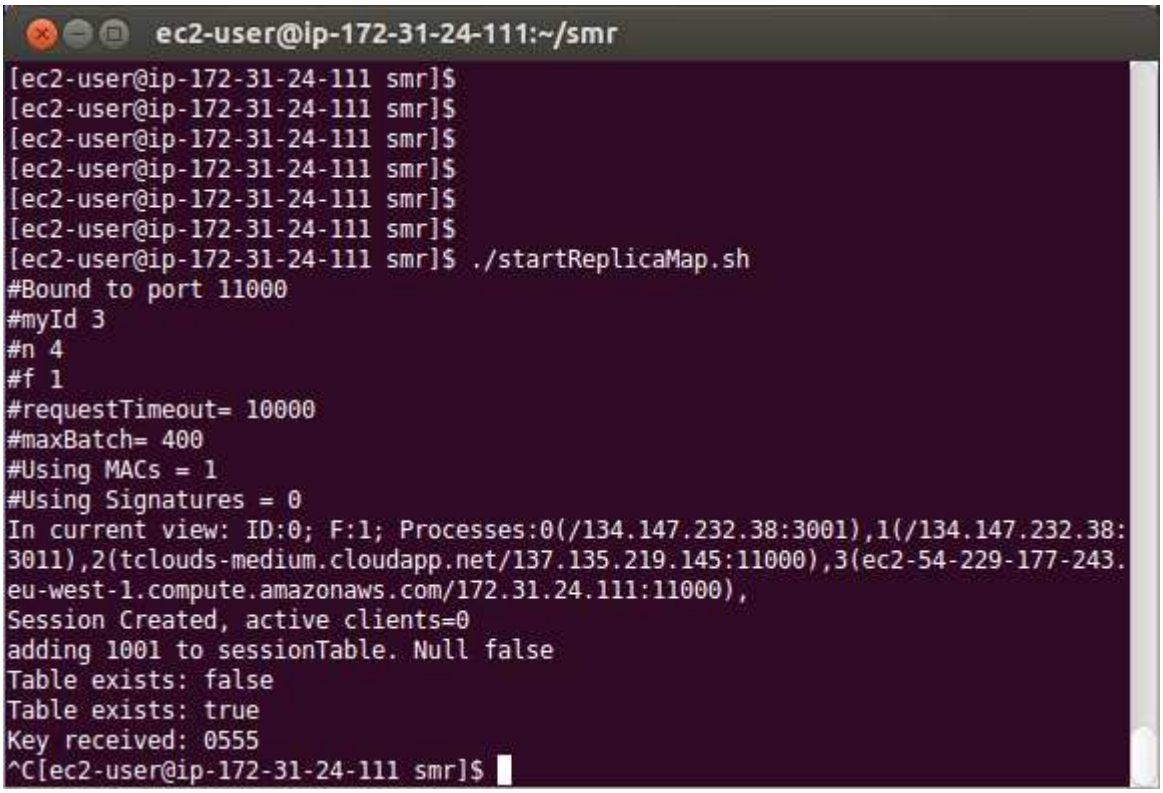


Figure 171 - The moment where the replica was interrupted in the Amazon EC2 cloud

3.2.2.15.2 BFT-SMaRt_2

Activity ID	**BFT-SMaRt_2**
Activity type	Proof of concept
Activity description	Test the protocol in a faulty non-leader replica 1- By using the Smart Lighting System appliance, continuously store and query data, checking that the queried data is always as expected. 2- Turn off a replica 3- Switch on the replica after a certain time
Acceptance Criteria	Point 1 never report an error while performing the disconnection of a replica.

Table 40 - BFT-SMaRt_2 validation activity description

Test setup and execution

A script was created to automate the execution of the test. This script start a YCSB Benchmark client and four replicas.

The script parses the logs returned by the replicas interrupting and restarting replicas to simulate the faults expected in the test case.

The result of the execution is displayed in Figure 172.

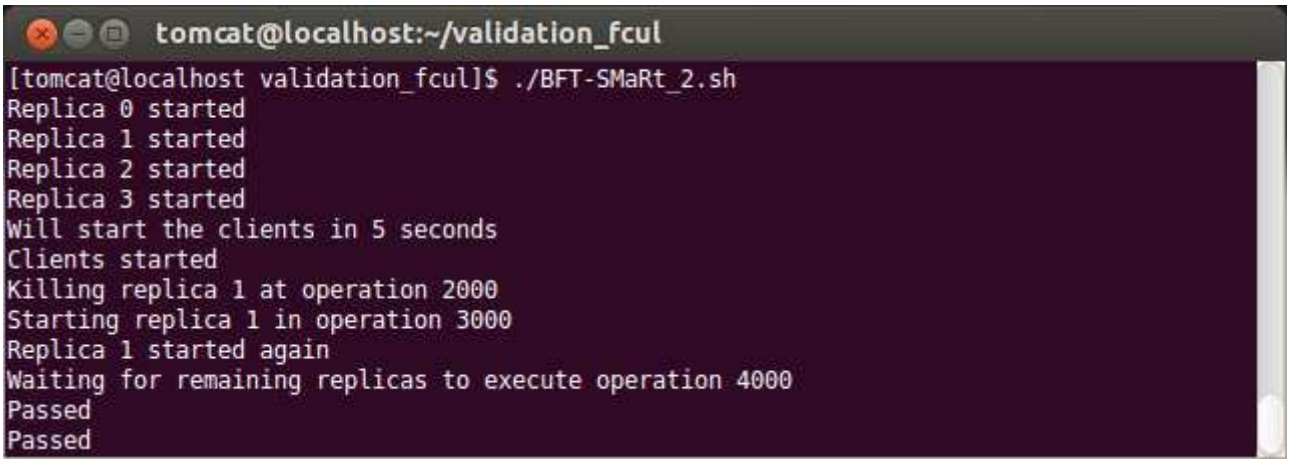


Figure 172 - Screen with the results from the script execution

Script code

```
#!/bin/sh

LOG_DIR=logs/BFT-SMaRt_2

ssh tclouds@192.168.21.2 "killall java; cd smr; ./startReplicaYCSB.sh" >
./$LOG_DIR/output_replica_0.log 2>&1 &
sshReplica0PID=$!
echo "Replica 0 started"
sleep 2;

ssh tclouds@192.168.22.2 "killall java; cd smr; ./startReplicaYCSB.sh" >
./$LOG_DIR/output_replica_1.log 2>&1 &
sshReplica1PID=$!
echo "Replica 1 started"
sleep 2;

ssh azureuser@tclouds-medium.cloudapp.net "killall java; cd smr; ./startReplicaYCSB.sh" >
./$LOG_DIR/output_replica_2.log 2>&1 &
sshReplica2PID=$!
echo "Replica 2 started"
sleep 2;

ssh -i ~/.ssh/tcloudsmedium.pem ec2-user@54.229.177.243 "killall java; cd smr; ./startReplicaYCSB.sh" >
./$LOG_DIR/output_replica_3.log 2>&1 &
sshReplica3PID=$!
echo "Replica 3 started"

echo "Will start the clients in 5 seconds"
sleep 5

./startYCSBClient.sh > ./$LOG_DIR/output_client.log 2>&1 &
clientPID=$!
echo "Clients started"

sleep 2

tail -n0 -F ./$LOG_DIR/output_replica_1.log | while read line1
do
    if echo $line1 | grep -q 'executing eid: 2000'; then
        echo "Killing replica 1 at operation 2000"
        ssh tclouds@192.168.22.2 "killall java" &
        pid=`ps aux | grep "tail -n0 -F ./$LOG_DIR/output_replica_1.log" | grep -v grep | awk '{print
$2}'`
        kill -9 $pid
```

```
        break
    fi
done

sleep 2

tail -n0 -F ./$LOG_DIR/output_replica_2.log | while read line
do
    if echo $line | grep -q 'executing eid: 3000'; then
        echo "Starting replica 1 in operation 3000"
        ssh tclouds@192.168.22.2 "killall java; cd smr; ./startReplicaYCSB.sh" >
./$LOG_DIR/output_replica_1.log 2>&1 &
        sshReplica1PID=$!
        echo "Replica 1 started again"
        sleep 2;
        pid=`ps aux | grep "tail -n0 -F ./$LOG_DIR/output_replica_2.log" | grep -v grep | awk '{print
$2}'`
        kill -9 $pid
    fi
done

echo "Waiting for remaining replicas to execute operation 4000"

tail -n0 -F ./$LOG_DIR/output_replica_2.log | while read line
do
    if echo $line | grep -q 'executing eid: 4000'; then
        echo 'Passed'
        pid=`ps aux | grep "tail -n0 -F ./$LOG_DIR/output_replica_2.log" | grep -v grep | awk '{print
$2}'`
        kill -9 $pid
    fi
done

kill -9 $sshReplica0PID
kill -9 $sshReplica2PID
kill -9 $sshReplica3PID
kill -9 $clientPID
```

3.2.2.15.3 BFT-SMaRt_3

Activity ID	**BFT-SMaRt_3**
Activity type	Proof of concept
Activity description	Test the protocol in a faulty non-leader replica 1- By using the Smart Lighting System appliance, continuously store and query data, checking that the queried data is always as expected. 2- Turn off a replica 3- Switch on the same replica after a certain time 4- Switch off another replica
Acceptance Criteria	• Point 1 never report an error while performing the disconnection of a replica • After point 4 the system is capable of responding to request by using the data it received from the state transfer protocol.

Table 41 - BFT-SMaRt_3 validation activity description

Test setup and results

We executed a script to start and stop replicas to validate the state transfer protocol.

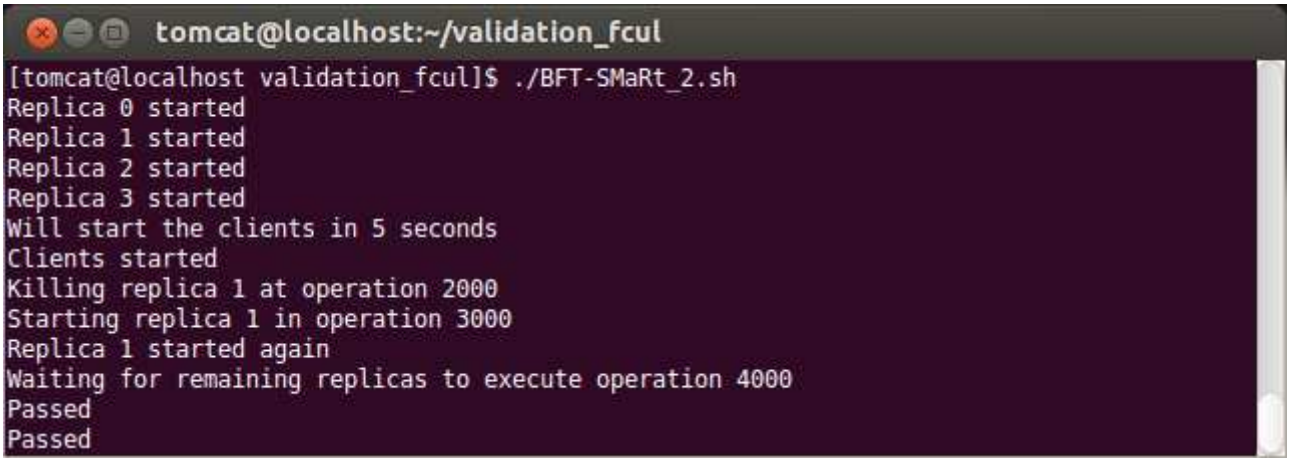


Figure 173 - Result of the script execution

The test passed as displayed in Figure 173.

Test script

```
#!/bin/sh

LOG_DIR=logs/BFT-SMaRt_3

ssh tclouds@192.168.21.2 "killall java; cd smr; ./startReplicaYCSB.sh" >
./$LOG_DIR/output_replica_0.log 2>&1 &
sshReplica0PID=$!
echo "Replica 0 started"
sleep 2;

ssh tclouds@192.168.22.2 "killall java; cd smr; ./startReplicaYCSB.sh" >
./$LOG_DIR/output_replica_1.log 2>&1 &
sshReplica1PID=$!
echo "Replica 1 started"
sleep 2;

ssh tclouds@192.168.23.2 "killall java; cd smr; ./startReplicaYCSB.sh" >
./$LOG_DIR/output_replica_2.log 2>&1 &
sshReplica2PID=$!
echo "Replica 2 started"
sleep 2;

ssh tclouds@192.168.24.2 "killall java; cd smr; ./startReplicaYCSB.sh" >
./$LOG_DIR/output_replica_3.log 2>&1 &
sshReplica3PID=$!
echo "Replica 3 started"

echo "Will start the clients in 5 seconds"
sleep 5

./startYCSBClient.sh > ./$LOG_DIR/output_client.log 2>&1 &
clientPID=$!
echo "Clients started"

sleep 2

tail -n0 -F ./$LOG_DIR/output_replica_1.log | while read line1
```

```
do
    if echo $line1 | grep -q 'executing eid: 2000'; then
        echo "Killing replica 1 at operation 2000"
        ssh tclouds@192.168.22.2 "killall java" &
        pid=`ps aux | grep "tail -n0 -F ./$LOG_DIR/output_replica_1.log" | grep -v grep | awk '{print
$2}'`
        kill -9 $pid
    fi
done

echo "Waiting for remaining replicas to execute operation 6000"

tail -n0 -F ./$LOG_DIR/output_replica_2.log | while read line
do
    if echo $line | grep -q 'executing eid: 6000'; then
        echo "Starting replica 1 in operation 6000"
        ssh tclouds@192.168.22.2 "killall java; cd smr; ./startReplicaYCSB.sh" >
./$LOG_DIR/output_replica_1.log 2>&1 &
        sshReplica1PID=$!
        echo "Replica 1 started again"
        sleep 2;
        pid=`ps aux | grep "tail -n0 -F ./$LOG_DIR/output_replica_2.log" | grep -v grep | awk '{print
$2}'`
        kill -9 $pid
    fi
done

tail -n0 -F ./$LOG_DIR/output_replica_2.log | while read line
do
    if echo $line | grep -q 'executing eid: 9000'; then
        echo "Killing replica 2 at operation 9000"
        ssh tclouds@192.168.23.2 "killall java" &
        pid=`ps aux | grep "tail -n0 -F ./$LOG_DIR/output_replica_2.log" | grep -v grep | awk '{print
$2}'`
        kill -9 $pid
    fi
done

tail -n0 -F ./$LOG_DIR/output_replica_3.log | while read line
do
    if echo $line | grep -q 'executing eid: 11000'; then
        echo 'Passed'
        pid=`ps aux | grep "tail -n0 -F ./$LOG_DIR/output_replica_3.log" | grep -v grep | awk '{print
$2}'`
        kill -9 $pid
    fi
done

kill -9 $sshReplica0PID
kill -9 $sshReplica1PID
kill -9 $sshReplica3PID
kill -9 $clientPID
```

3.2.2.15.4 BFT-SMaRt_4

Activity ID	**BFT-SMaRt_4**
Activity type	Proof of concept
Activity description	Test the protocol in a faulty leader replica 1- By using the Smart Lighting System appliance, continuously store and query data, checking that the

	queried data is always as expected. 2- Turn off a leader replica
Acceptance Criteria	Point 1 never report an error while performing the disconnection of a replica

Table 42 - BFT-SMaRt_4 validation activity description

Test setup and execution

The script executed the tests, stopping the leader replica. After some time it verified that the other replicas continued to make progress. This proves that the leader change protocol is working, as the SMR protocol can't make progress without a leader.

The result of the script execution is displayed in Figure 174 .

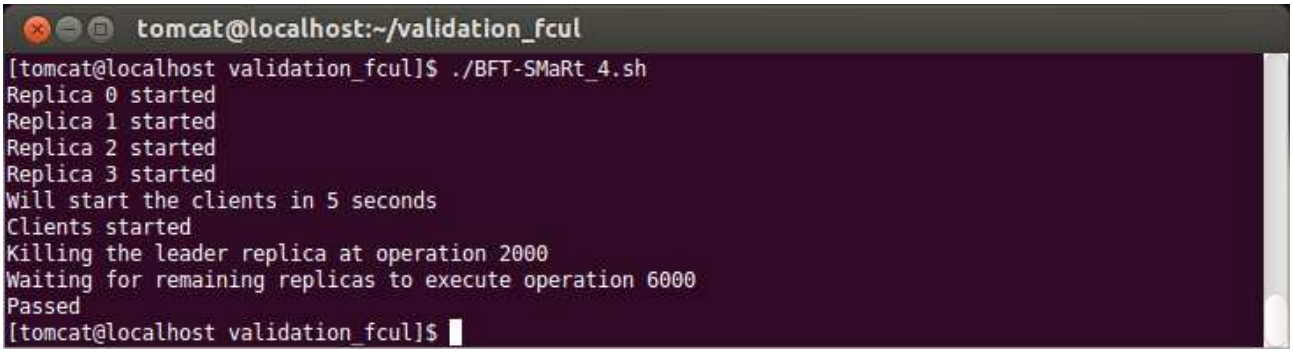


Figure 174 - Script output for the test execution

Test script

```
#!/bin/sh

LOG_DIR=logs/BFT-SMaRt_4

ssh tclouds@192.168.21.2 "killall java; cd smr; ./startReplicaYCSB.sh" >
./$LOG_DIR/output_replica_0.log 2>&1 &
sshReplica0PID=$!
echo "Replica 0 started"
sleep 2;

ssh tclouds@192.168.22.2 "killall java; cd smr; ./startReplicaYCSB.sh" >
./$LOG_DIR/output_replica_1.log 2>&1 &
sshReplica1PID=$!
echo "Replica 1 started"
sleep 2;

ssh tclouds@192.168.23.2 "killall java; cd smr; ./startReplicaYCSB.sh" >
./$LOG_DIR/output_replica_2.log 2>&1 &
sshReplica2PID=$!
echo "Replica 2 started"
sleep 2;

ssh tclouds@192.168.24.2 "killall java; cd smr; ./startReplicaYCSB.sh" >
./$LOG_DIR/output_replica_3.log 2>&1 &
sshReplica3PID=$!
echo "Replica 3 started"

echo "Will start the clients in 5 seconds"
```

```
sleep 5

./startYCSBClient.sh > ./$LOG_DIR/output_client.log 2>&1 &
clientPID=$!
echo "Clients started"

sleep 2

tail -n0 -F ./$LOG_DIR/output_replica_0.log | while read line1
do
    if echo $line1 | grep -q 'executing eid: 2000'; then
        echo "Killing the leader replica at operation 2000"
        ssh tclouds@192.168.21.2 "killall java" &
        pid=`ps aux | grep "tail -n0 -F ./$LOG_DIR/output_replica_0.log" | grep -v grep | awk '{print
$2}'`
        kill -9 $pid
    fi
done

echo "Waiting for remaining replicas to execute operation 6000"

tail -n0 -F ./$LOG_DIR/output_replica_3.log | while read line
do
    if echo $line | grep -q 'executing eid: 6000'; then
        echo 'Passed'
        pid=`ps aux | grep "tail -n0 -F ./$LOG_DIR/output_replica_3.log" | grep -v grep | awk '{print
$2}'`
        kill -9 $pid
    fi
done

kill -9 $sshReplica1PID
kill -9 $sshReplica2PID
kill -9 $sshReplica3PID
kill -9 $clientPID
```

3.2.2.15.5 BFT-SMaRt_5

Activity ID	**BFT-SMaRt_5**
Activity type	Proof of concept
Activity description	Test the leader change protocol and state transfer protocol 1- By using the Smart Lighting System appliance, continuously store and query data, checking that the queried data is always as expected. 2- All replicas, including the leader are switched on and off in a round robin fashion
Acceptance Criteria	Point 1 never report an error while performing the disconnection of a replica

Table 43 - BFT-SMaRt_5 validation activity description

Test setup and execution

The script verified that the system tolerates faults in leader and non-leader replicas, validating the leader change and state transfer protocols.

The script resuls are displayed in Figure 175

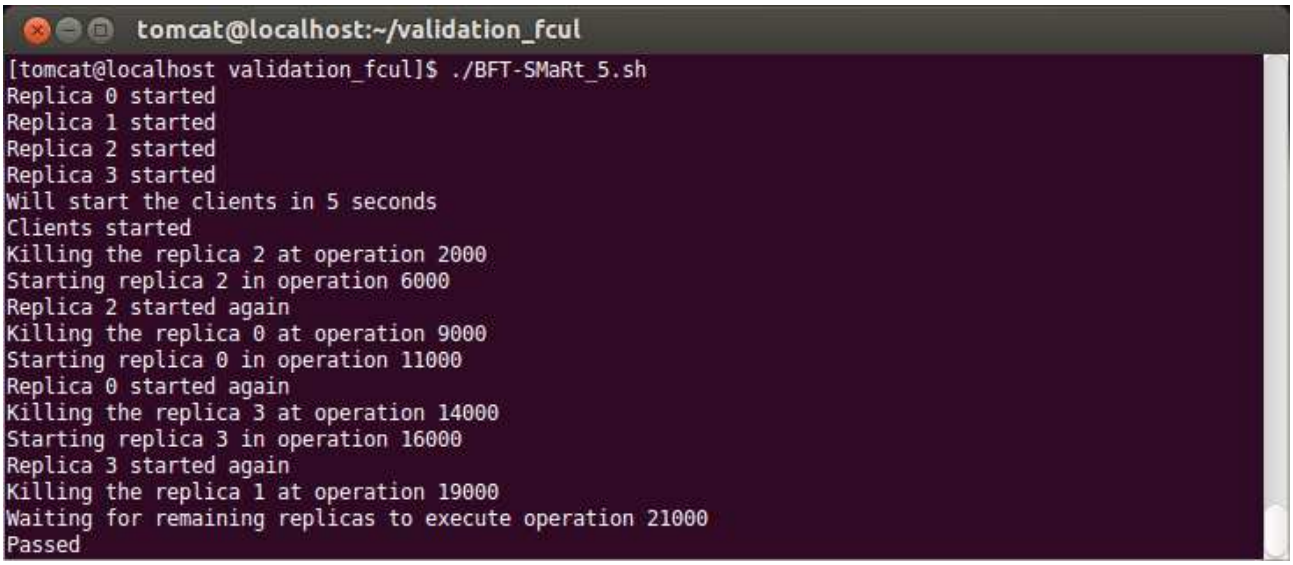


Figure 175 - Script results showing the correct execution of the test

Test script

```
#!/bin/sh

LOG_DIR=logs/BFT-SMaRt_5

ssh tclouds@192.168.21.2 "killall java; cd smr; ./startReplicaYCSB.sh" >
./$LOG_DIR/output_replica_0.log 2>&1 &
sshReplica0PID=$!
echo "Replica 0 started"
sleep 2;

ssh tclouds@192.168.22.2 "killall java; cd smr; ./startReplicaYCSB.sh" >
./$LOG_DIR/output_replica_1.log 2>&1 &
sshReplica1PID=$!
echo "Replica 1 started"
sleep 2;

ssh tclouds@192.168.23.2 "killall java; cd smr; ./startReplicaYCSB.sh" >
./$LOG_DIR/output_replica_2.log 2>&1 &
sshReplica2PID=$!
echo "Replica 2 started"
sleep 2;

ssh tclouds@192.168.24.2 "killall java; cd smr; ./startReplicaYCSB.sh" >
./$LOG_DIR/output_replica_3.log 2>&1 &
sshReplica3PID=$!
echo "Replica 3 started"

echo "Will start the clients in 5 seconds"
sleep 5

./startYCSBClient.sh > ./$LOG_DIR/output_client.log 2>&1 &
clientPID=$!
echo "Clients started"

sleep 2

tail -n0 -F ./$LOG_DIR/output_replica_2.log | while read line1
do
    if echo $line1 | grep -q 'executing eid: 2000'; then
        echo "Killing the replica 2 at operation 2000"
```

```
        ssh tclouds@192.168.23.2 "killall java" &
        pid=`ps aux | grep "tail -n0 -F ./$LOG_DIR/output_replica_2.log" | grep -v grep | awk '{print
$2}'`
        kill -9 $pid
    fi
done

sleep 2

tail -n0 -F ./$LOG_DIR/output_replica_3.log | while read line
do
    if echo $line | grep -q 'executing eid: 6000'; then
        echo "Starting replica 2 in operation 6000"
        ssh tclouds@192.168.23.2 "killall java; cd smr; ./startReplicaYCSB.sh" >
./$LOG_DIR/output_replica_2.log 2>&1 &
        sshReplica2PID=$!
        echo "Replica 2 started again"
        sleep 2;
        pid=`ps aux | grep "tail -n0 -F ./$LOG_DIR/output_replica_3.log" | grep -v grep | awk '{print
$2}'`
        kill -9 $pid
    fi
done

sleep 2

tail -n0 -F ./$LOG_DIR/output_replica_0.log | while read line1
do
    if echo $line1 | grep -q 'executing eid: 9000'; then
        echo "Killing the replica 0 at operation 9000"
        ssh tclouds@192.168.21.2 "killall java" &
        pid=`ps aux | grep "tail -n0 -F ./$LOG_DIR/output_replica_0.log" | grep -v grep | awk '{print
$2}'`
        kill -9 $pid
    fi
done

sleep 2

tail -n0 -F ./$LOG_DIR/output_replica_1.log | while read line
do
    if echo $line | grep -q 'executing eid: 11000'; then
        echo "Starting replica 0 in operation 11000"
        ssh tclouds@192.168.21.2 "killall java; cd smr; ./startReplicaYCSB.sh" >
./$LOG_DIR/output_replica_0.log 2>&1 &
        sshReplica0PID=$!
        echo "Replica 0 started again"
        sleep 2;
        pid=`ps aux | grep "tail -n0 -F ./$LOG_DIR/output_replica_1.log" | grep -v grep | awk '{print
$2}'`
        kill -9 $pid
    fi
done

tail -n0 -F ./$LOG_DIR/output_replica_3.log | while read line1
do
    if echo $line1 | grep -q 'executing eid: 14000'; then
        echo "Killing the replica 3 at operation 14000"
        ssh tclouds@192.168.24.2 "killall java" &
        pid=`ps aux | grep "tail -n0 -F ./$LOG_DIR/output_replica_3.log" | grep -v grep | awk '{print
$2}'`
        kill -9 $pid
    fi
done
```

```
sleep 2

tail -n0 -F ./$LOG_DIR/output_replica_2.log | while read line
do
    if echo $line | grep -q 'executing eid: 16000'; then
        echo "Starting replica 3 in operation 16000"
        ssh tclouds@192.168.24.2 "killall java; cd smr; ./startReplicaYCSB.sh" >
./$LOG_DIR/output_replica_3.log 2>&1 &
        sshReplica3PID=$!
        echo "Replica 3 started again"
        sleep 2;
        pid=`ps aux | grep "tail -n0 -F ./$LOG_DIR/output_replica_2.log" | grep -v grep | awk '{print
$2}'`
        kill -9 $pid
    fi
done

sleep 2

tail -n0 -F ./$LOG_DIR/output_replica_1.log | while read line1
do
    if echo $line1 | grep -q 'executing eid: 19000'; then
        echo "Killing the replica 1 at operation 19000"
        ssh tclouds@192.168.22.2 "killall java" &
        pid=`ps aux | grep "tail -n0 -F ./$LOG_DIR/output_replica_1.log" | grep -v grep | awk '{print
$2}'`
        kill -9 $pid
    fi
done

sleep 2

echo "Waiting for remaining replicas to execute operation 21000"

tail -n0 -F ./$LOG_DIR/output_replica_3.log | while read line
do
    if echo $line | grep -q 'executing eid: 21000'; then
        echo 'Passed'
        pid=`ps aux | grep "tail -n0 -F ./$LOG_DIR/output_replica_3.log" | grep -v grep | awk '{print
$2}'`
        kill -9 $pid
    fi
done

kill -9 $sshReplica0PID
kill -9 $sshReplica2PID
kill -9 $sshReplica3PID
kill -9 $clientPID
```

3.2.2.16 Conclusion

We deployed BFT-SMaRt in the Trusted Infrastructure. Through the execution of the scripts created to automate the tests we verified that replicas continued to make progress even in the presence of up to *f* faults, being them in the leader and non-leader replicas. That proved that the protocol tolerates faults and reduces the windows of vulnerability through the use of leader change and state transfer protocols.

All BFT-SMaRt validation activities were executed and conformed to expectations.

3.3 Validation of components not used by Healthcare and Smart Light System scenario

TClouds infrastructure resulted to be a comprehensive tool able to host different customer needs. The Healthcare and Smart Light System scenario represent two particular realities that needs specific cloud features. Nonetheless TClouds encompass other subcomponents that might be useful for other needs. Most of these components are high level components that take advantage of the SaaS paradigm and become useful for all those companies that don't have to setup complex platform or systems, but just need cloud features for internal needs and to externalize IT infrastructures.

Given this, in the following paragraphs we will show the validation activities of TClouds those subcomponents that have not been used directly by Healthcare of Smart Light System scenario and of those subcomponents that have been used by TEC to externalize their IT systems used to manage TClouds project itself (such as SVN and Jabber systems)

3.3.1 Results of Tecnikon validation

During the last period of TClouds it become essential to test all the components, developed during the project. TECHNIKON declared their willingness to test and evaluate three different components in combination with its WEB 2.0 services. The three components are:

1.) **S3 Proxy**

Developed by SRX; test functionality;

2.) **ICStore**

Developed by IBM; test functionality;

3.) **Trusted Infrastructure**

Developed by SRX; test usability;

The main idea behind this evaluation was to build a system with a strong data protection and availability which eventually can be used for data exchange between the Consortium and the European Commission. Furthermore every partner involved in this evaluation should profit from it, especially by reporting bugs in their modules and in case of Trusted Infrastructure by making suggestions for a better usability. For the final test it is planned to use this system to provide all final deliverables for the reviewers of TClouds. This will be the first time that somebody from outside the project will get access to one of the developed secure cloud platforms.

3.3.1.1 Activity Description

Activity ID	**S3 Proxy Evaluation**
Activity type	functional test
Activity description	Evaluate the functionality of the S3 proxy by using it in combination with one of TEC's Web services. 1. Create TVD and deploy the VM with the service on it 2. Integrate S3 proxy on VM 3. Synchronise data with S3 proxy 4. Check if data is written correctly to S3 proxy and the cloud
Acceptance Criteria	The Activity is passed if all data which has been handed over to S3 proxy is automatically encrypted and synchronised with the cloud.

Table 44 - Activity S3 Proxy Evaluation

Activity ID	ICStore Evaluation
Activity type	functional test
Activity description	Evaluate the functionality of the ICStore by using it in combination with one of TEC's Web services. 1. Create TVD and deploy the VM with the service on it 2. Integrate and run ICStore on VM 3. Check if data is written correctly to cloud (backup folder)
Acceptance Criteria	The Activity is passed if all data is automatically encrypted and synchronised with the cloud.

Table 45 - Activity ICStore Evaluation

Activity ID	Trusted Infrastructure Evaluation
Activity type	usability test, reliability test
Activity description	Evaluate the usability of the Trusted Infrastructure. 1. Test GUI of Trusted Desktop 2- Test GUI of Trusted Objects Manager 2. Test accessibility and some other technical features like encryption 3. Write report with suggestions for SRX how to improve it.
References Documents	Evaluation Report TI

Table 46 - Activity Trusted Infrastructure

3.3.1.2 Execution of Activities

First of all a VM has to be prepared to perform the tests. After correlation with SRX we decided that we have to create a VM with Oracle VM Virtual Box to run it on their Trusted Infrastructure. It is an openSUSE 12.1 (Asparagus) with 256MB RAM and with 4GB of hard disk space.

3.3.1.2.1 S3 Proxy Evaluation

The service which is running on the VM to test the S3 proxy is a SVN server 1.6.21 and later on it will be also used to evaluate ICStore. After we have finished the setup of the VM we handed it over to SRX. On starting the VM the first time on SRX's Trusted Infrastructure, some problems occurred and the VM won't start. After some time of analysing and troubleshooting we were able to start it.

To reach the VM over the Internet some changes in the configuration inside the VM and also in the TOM have to be done. The VM was reachable over the public IP 134.147.232.38:2400 for SSH. Furthermore the Apache Web server has to be configured to use ports 3080 for http and 3443 for https. Now the SVN server was reachable and we could start to configure and integrate the S3 proxy.

The following chart depicts an organizational overview of how S3 proxy operates.

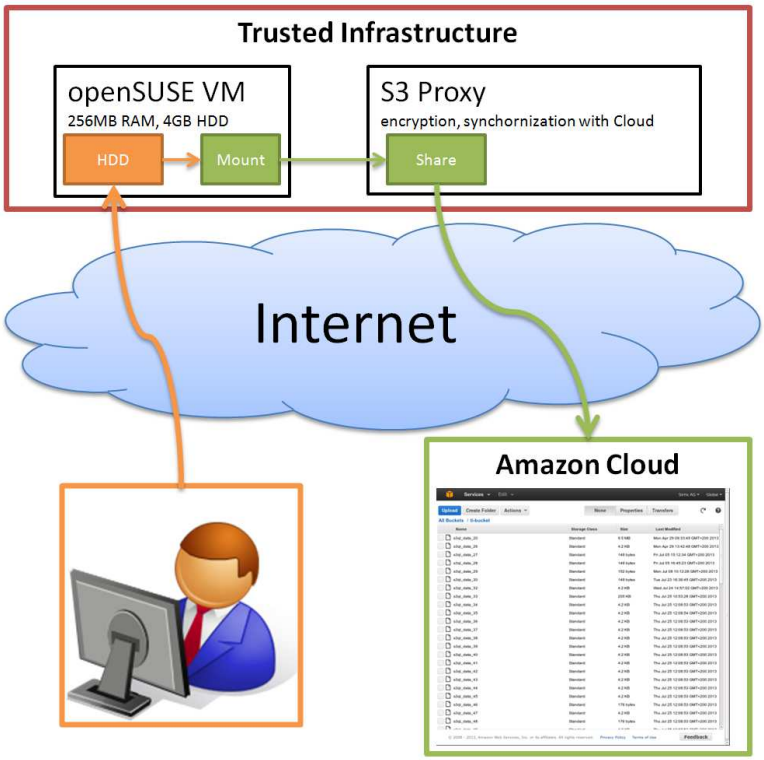


Figure 176 - S3 Proxy overview

The proxy is like a NFS share for the client and was mounted on */mnt/AZ/AZ3* configured to synchronise all data with Amazon S3. All the data which have been changed on the SVN was automatically copied to the share every five minutes.

3.3.1.2.2 ICStore Evaluation

For the Evaluation of IC Store we used the nearly the same scenario like for the evaluation of S3 proxy. It is the same VM, the same SVN server and the same Data. But, ICStore is a bit different comparing with S3 proxy. S3 proxy is a separate VM which shares a folder to mount on your VM. ICStore is a script which is running in the background and monitors the folder you have configured. When there is a change it takes the file, encrypt it and copy it to the destination you have configured. In our case it was first configured to store the data in a local directory because we had no user account for a cloud provider, the test with the Amazon S3 cloud storage was conducted later on.

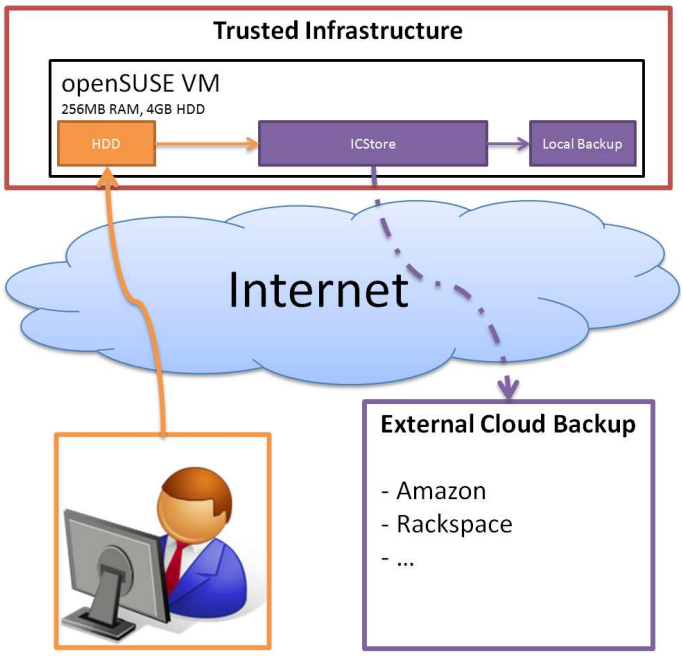


Figure 177 - ICStore overview

We received the ICStore package from Nikola Knezevic (IBM). Before ICStore could be used it requires **Java**, **zsh** and **attr** packet to be installed and some other configuration changes in the ***options.conf*** file. The following configuration has been used to run ICStore within our scenario.

```
##!/usr/bin/env zsh

#############################################################################################
## Common options for all ICStore scripts
#############################################################################################

# which directory should ICStore monitor (the one that is exported via NFS/SMB)
MONITORDIR=/root/ICStore-Monitor/              monitors SVN-Repository

# where is java?
JAVA=/usr/bin/java
JAVAOPTS=-Dfile.encoding=UTF-8

# container name into which one will store the files
CONTAINERNAME=tcloudsbackup

# config file to use
CONFFILE=config/ilm-single-fs.icstore Perform backup local

# sleep time in seconds between checks
SLEEP_TIME=60                          changed from 120 to 60

# extended attributes, replace with your systems functions
# write_xattr(ATTR, VALUE, FILE)
write_xattr()
{
    # xattr -w $1 $2 $3                        Command for MacOS
    # Linux:
    setfattr -n user.$1 -v $2 $3               Command for Linux
}

# read_xattr(ATTR, FILE)
read_xattr()
{
    # xattr -p $1 $2                           Command for MacOS
    # Linux:
    getfattr -n user.$1 $2                     Command for Linux
}

# should we be verbose? (no by default)
VERBOSE=0
```

With this configuration, all files which have been changed in */srv/svn/tc-test/* are synchronized with the specified location every 60 seconds. In our case we decided to use a local backup. This was done with the config file *ilm-single-fs.icstore*.

By starting the script run-icstore.sh the first time several errors appear. After some troubleshooting in cooperation with IBM we found the reasons for those problems.

The script runs smoothly and copies all data when you start it but it currently has some problems with the monitoring. IBM has investigated this issue subsequently.

After doing this, the above configuration was modified in order to make ICStore utilize the Amazon S3 cloud storage by supplying *ilm-single-s3.icstore* as configuration-file. In the said configuration file, the corresponding credentials for the Amazon S3 had to be provided. After start-up an appropriate container was created on the Amazon storage and the files were uploaded as intended. Again, the files are changed in the monitored folder are synchronized every 60 seconds.

3.3.1.3 Outcome

3.3.1.3.1.1 S3 Proxy

S3 Proxy was working as promised by SRX. The test was running over a period of 2 weeks without any problem on accessibility, encryption or file integrity. To make the outcome easier to understand you can find some screenshots here.

This is the plain mounted share. All data is encrypted and will be only available if it is mounted in an appropriate TVD.

Figure 178 - Plain mounted share

Here you can see the data which have been created under our TVD. The first two files are encrypted and from another TVD.

Figure 179 - TVD data

This screenshot shows that S3 proxy is mounted and the encryptfs-layers are on it.

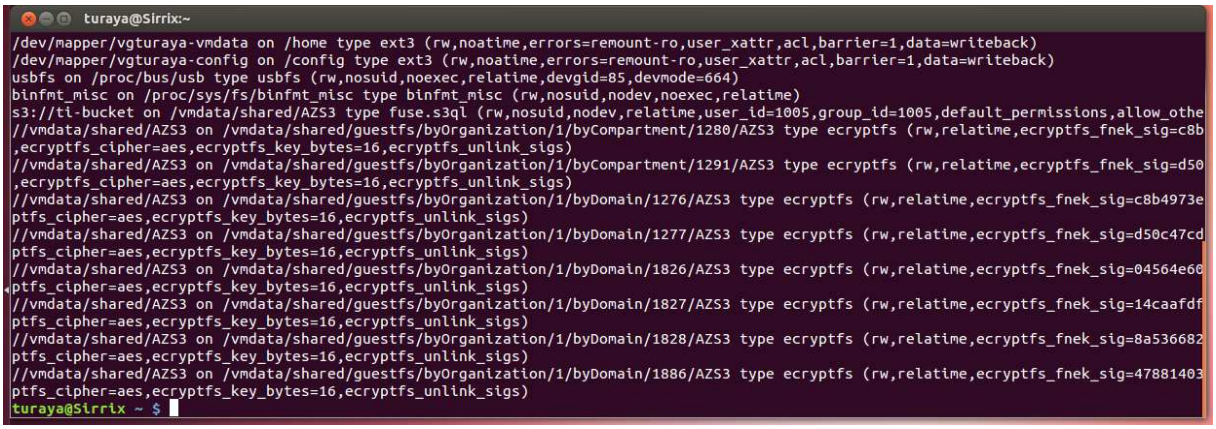


Figure 180 - mounted S3 proxy with encryptfs-layers

This is the data when you look at it over the Amazon console. Amazon doesn't really know what to do with the provided data so the save it as only some files.

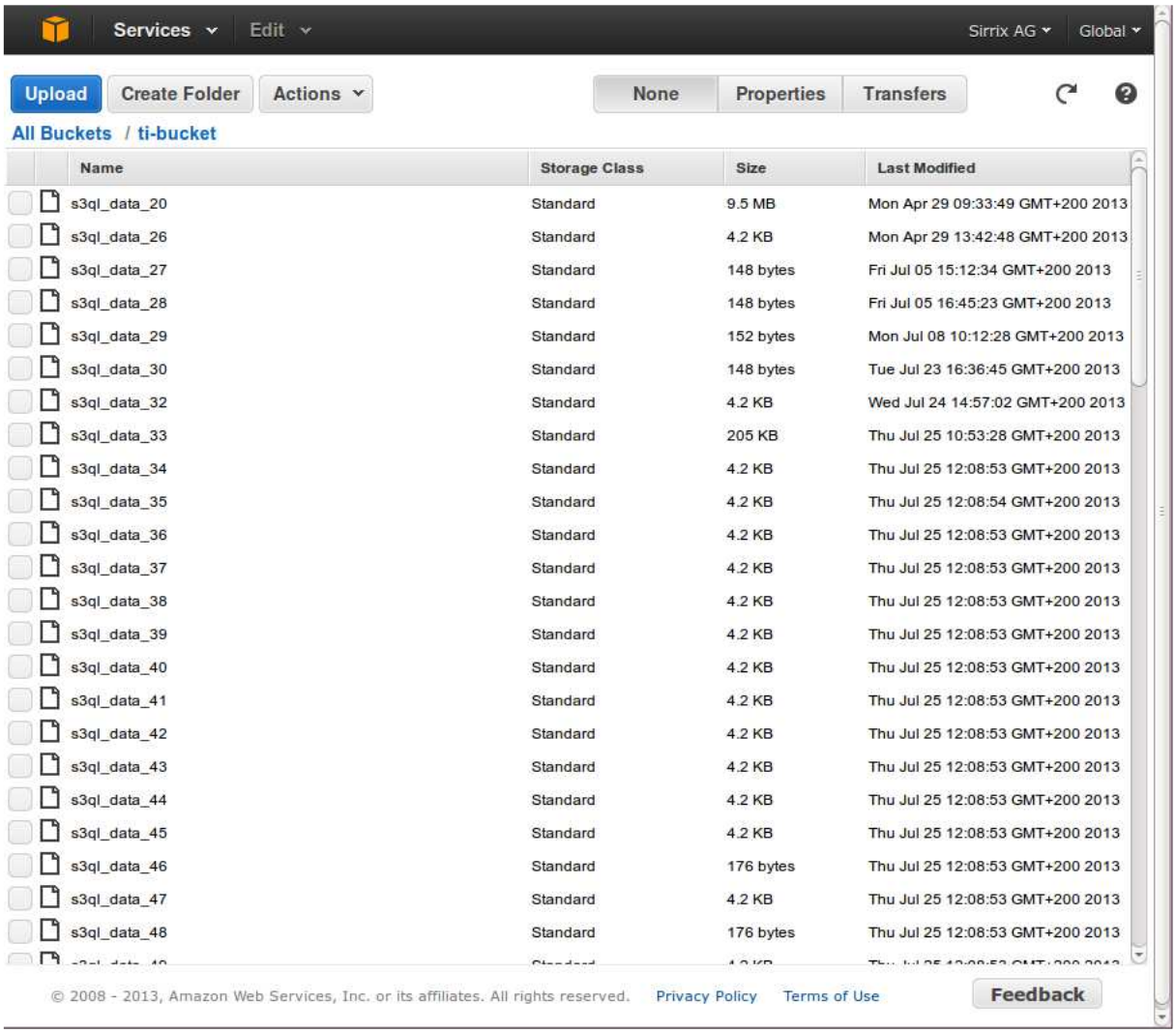


Figure 181 - Amazon S3 bucket with data

3.3.1.3.1.2 ICStore

ICStore was working as intended. Again, some screenshots have been provided for a better understanding of the outcomes.

The image below depicts our monitored folder; the files which are put or changed in here are synced to the local file system or to the Amazon cloud storage respectively.

```
TC-Test-Server:~ # ls -al ICStore-Monitor/
total 84
drwxr-xr-x  2 root root 4096 Sep 19 09:09 .
drwx------ 10 root root 4096 Sep 18 15:07 ..
-rw-r--r--  1 root root    4 Sep 18 14:58 TEST-1.TXT
-rw-r--r--  1 root root    4 Sep 18 14:58 TEST-10.TXT
-rw-r--r--  1 root root    4 Sep 18 14:58 TEST-2.TXT
-rw-r--r--  1 root root    4 Sep 18 14:58 TEST-3.TXT
-rw-r--r--  1 root root    4 Sep 18 14:58 TEST-4.TXT
-rw-r--r--  1 root root    4 Sep 18 14:58 TEST-5.TXT
-rw-r--r--  1 root root    4 Sep 18 14:58 TEST-6.TXT
-rw-r--r--  1 root root    4 Sep 18 14:58 TEST-7.TXT
-rw-r--r--  1 root root    4 Sep 18 14:58 TEST-8.TXT
-rw-r--r--  1 root root    4 Sep 18 14:58 TEST-9.TXT
-rw-r--r--  1 root root    4 Sep 18 14:58 TEST.TXT
-rw-r--r--  1 root root 2152 Oct  4  2012 post-commit
-rw-r--r--  1 root root  308 Aug 10  2009 svn.redirect.dragon-project.eu.conf
-rw-r--r--  1 root root 1605 Nov  8  2012 svnmailer.conf
-rw-r--r--  1 root root 1376 Jul  2 13:38 tc-test.technikon.com.conf
-rw-r--r--  1 root root  654 Jul  2 15:38 tc-test_80.technikon.com.conf
-rw-r--r--  1 root root  237 Aug  7 10:31 tclouds_redirect.conf
-rw-r--r--  1 root root  894 Jul 11 15:11 test.conf
```

Figure 182 - Monitored folder on the local file system

In the following examples ICStore is started with the command line option '-f' which forces an initial upload of all files.

On the following screenshot it can be seen how ICStore operates on the local file system; it creates an appropriate container (in this case a folder on the local FS) to sync the contents of the monitored Folder into.

```
TC-Test-Server:~ # ./icstore-0.8.1-SNAPSHOT/run-icstore.sh -v -f &
[1] 17887
TC-Test-Server:~ # Starting...
08:47:55,356 |-INFO in ch.qos.logback.core.joran.action.AppenderAct
08:47:55,521 |-INFO in ch.qos.logback.core.joran.action.AppenderAct
08:47:56,075 |-INFO in ch.qos.logback.core.joran.action.NestedCompl
08:47:56,859 |-INFO in ch.qos.logback.core.joran.action.AppenderAct
08:47:56,864 |-INFO in ch.qos.logback.core.joran.action.AppenderAct
08:47:56,888 |-INFO in ch.qos.logback.core.joran.action.NestedCompl
08:47:56,925 |-INFO in ch.qos.logback.classic.joran.action.LoggerAc
08:47:56,928 |-INFO in ch.qos.logback.classic.joran.action.LoggerAc
08:47:56,933 |-INFO in ch.qos.logback.classic.joran.action.LoggerAc
08:47:56,934 |-INFO in ch.qos.logback.classic.joran.action.RootLogg
08:47:56,943 |-INFO in ch.qos.logback.core.joran.action.AppenderRef
08:47:56,972 |-INFO in ch.qos.logback.classic.joran.action.Configur
08:47:56,993 |-INFO in ch.qos.logback.classic.joran.JoranConfigurat
[main][]        BlobStoreConnection:build - BlobStore loaded for pr
[main][]      SizeHandler:queryModuleTree - Queried enc0 for SizeRe
[main][]      SizeHandler:queryModuleTree - Queried fileSystem0 for
[main][]                SizeHandler:call - Max Slice Size for 3276
[main][]                ICStore:prologue - client client/tcloudsba
[main][enc0]             Encryption:prologue - enc0 client/tclouds
[pool-9-thread-1][fileSystem0]      BlobStoreConnection:acquire - G
[pool-9-thread-1][fileSystem0]     BlobStoreConnection:prologue - f
[pool-9-thread-1][fileSystem0]      BlobStoreConnection:release - G
[pool-9-thread-1][fileSystem0] BlobStoreConnection:epilogueDone - f
[pool-9-thread-1][enc0]          Encryption:epilogueDone - enc0 cli
[pool-9-thread-1][client]                ICStore:release - GUARD
[pool-9-thread-1][client]             ICStore:epilogueDone - client
*** Container creation completed ***
```

Figure 183 - ICStore starting up and creating the local container

After that, it starts scanning the monitored folder and syncs the files to the local container. This scan is configured to be carried out every 60 seconds. If changed or new files are found, these are uploaded again.

```
Scanning...
Doing TEST-1.TXT -> test-1.txt
08:48:50,642 |-INFO in ch.qos.logback.core.joran.action.AppenderAction -
08:48:50,797 |-INFO in ch.qos.logback.core.joran.action.AppenderAction -
08:48:51,296 |-INFO in ch.qos.logback.core.joran.action.NestedComplexPro
08:48:52,131 |-INFO in ch.qos.logback.core.joran.action.AppenderAction -
08:48:52,136 |-INFO in ch.qos.logback.core.joran.action.AppenderAction -
08:48:52,146 |-INFO in ch.qos.logback.core.joran.action.NestedComplexPro
08:48:52,171 |-INFO in ch.qos.logback.classic.joran.action.LoggerAction
08:48:52,175 |-INFO in ch.qos.logback.classic.joran.action.LoggerAction
08:48:52,184 |-INFO in ch.qos.logback.classic.joran.action.LoggerAction
08:48:52,188 |-INFO in ch.qos.logback.classic.joran.action.RootLoggerAct
08:48:52,193 |-INFO in ch.qos.logback.core.joran.action.AppenderRefActio
08:48:52,217 |-INFO in ch.qos.logback.classic.joran.action.Configuration
08:48:52,243 |-INFO in ch.qos.logback.classic.joran.JoranConfigurator@61
[main][]         BlobStoreConnection:build - BlobStore loaded for provide
[main][]       SizeHandler:queryModuleTree - Queried enc0 for SizeRequire
[main][]       SizeHandler:queryModuleTree - Queried fileSystem0 for Size
[main][]                  SizeHandler:call - Max Slice Size for 32768 is
[main][]       SizeHandler:queryModuleTree - Queried enc0 for SizeRequire
[main][]       SizeHandler:queryModuleTree - Queried fileSystem0 for Size
[main][]                  SizeHandler:call - Max Total Size for 4 is 36
[main][]                  ICStore:prologue - client client/tcloudsbackup/
[main][enc0]               Encryption:prologue - enc0 client/tcloudsbacku
[pool-9-thread-1][fileSystem0]      BlobStoreConnection:acquire - GUARD
[pool-9-thread-1][fileSystem0]      BlobStoreConnection:acquire - GUARD
[pool-9-thread-1][fileSystem0]     BlobStoreConnection:prologue - fileSy
[Thread-4][fileSystem0]      BlobStoreConnection:release - GUARD - clien
[Thread-4][fileSystem0]      BlobStoreConnection:release - GUARD - clien
[Thread-4][fileSystem0] BlobStoreConnection:epilogueDone - fileSystem0 c
[Thread-4][enc0]          Encryption:epilogueDone - enc0 client/tcloudsb
[Thread-4][client]                  ICStore:release - GUARD - client/tcl
[Thread-4][client]                  ICStore:release - GUARD - client/tcl
[Thread-4][client]             ICStore:epilogueDone - client client/tclo
*** Upload completed ***
```

Figure 184 - ICStore detects and uploads files

The synchronized files can then be looked up in the corresponding folder.

```
TC-Test-Server:~ # ls -al ICStore-Mirror/tcloudsbackup.0/
total 68
drwxr-xr-x 2 root root 4096 Sep 19 08:56 .
drwxr-xr-x 3 root root 4096 Aug 28 16:13 ..
-rw-r--r-- 1 root root 2180 Aug 28 16:13 post-commit
-rw-r--r-- 1 root root  340 Aug 28 16:14 svn.redirect.dragon-project.eu.conf
-rw-r--r-- 1 root root 1636 Aug 28 16:14 svnmailer.conf
-rw-r--r-- 1 root root 1412 Aug 28 16:14 tc-test.technikon.com.conf
-rw-r--r-- 1 root root  676 Aug 28 16:26 tc-test_80.technikon.com.conf
-rw-r--r-- 1 root root  260 Aug 28 16:25 tclouds_redirect.conf
-rw-r--r-- 1 root root   36 Sep 19 09:11 test-1.txt
-rw-r--r-- 1 root root   36 Sep 19 09:12 test-10.txt
-rw-r--r-- 1 root root   36 Sep 19 08:51 test-2.txt
-rw-r--r-- 1 root root   36 Sep 19 08:52 test-3.txt
-rw-r--r-- 1 root root   36 Sep 19 08:53 test-4.txt
-rw-r--r-- 1 root root   36 Sep 19 08:54 test-5.txt
-rw-r--r-- 1 root root   36 Sep 19 08:55 test-6.txt
-rw-r--r-- 1 root root   36 Sep 19 08:56 test-7.txt
-rw-r--r-- 1 root root  916 Aug 28 16:14 test.conf
```

Figure 185 - Container on the local file system

Here you can see ICStore starting up when operating on Amazon S3. The necessary container (bucket) is created automatically.

```
TC-Test-Server:~ # ./icstore-0.8.1-SNAPSHOT/run-icstore.sh -v -f &
[2] 13603
[1]   Terminated              ./icstore-0.8.1-SNAPSHOT/run-icstore.sh
TC-Test-Server:~ # Starting...
15:41:36,371 |-INFO in ch.qos.logback.core.joran.action.AppenderActio
15:41:36,681 |-INFO in ch.qos.logback.core.joran.action.AppenderActio
15:41:37,483 |-INFO in ch.qos.logback.core.joran.action.NestedComplex
15:41:38,541 |-INFO in ch.qos.logback.core.joran.action.AppenderActio
15:41:38,546 |-INFO in ch.qos.logback.core.joran.action.AppenderActio
15:41:38,607 |-INFO in ch.qos.logback.core.joran.action.NestedComplex
15:41:38,724 |-INFO in ch.qos.logback.classic.joran.action.LoggerActi
15:41:38,735 |-INFO in ch.qos.logback.classic.joran.action.LoggerActi
15:41:38,745 |-INFO in ch.qos.logback.classic.joran.action.LoggerActi
15:41:38,756 |-INFO in ch.qos.logback.classic.joran.action.RootLogger
15:41:38,761 |-INFO in ch.qos.logback.core.joran.action.AppenderRefAc
15:41:38,781 |-INFO in ch.qos.logback.classic.joran.action.Configurat
15:41:38,812 |-INFO in ch.qos.logback.classic.joran.JoranConfigurator
[main][]        BlobStoreConnection:build - BlobStore loaded for prov
[main][]        BlobStoreConnection:build - BlobStore loaded for prov
[main][]      SizeHandler:queryModuleTree - Queried enc0 for SizeRequ
[main][]      SizeHandler:queryModuleTree - Queried s3eu for SizeRequ
[main][]                SizeHandler:call - Max Slice Size for 262144
[main][]                ICStore:prologue - client client/tcloudsback
[main][enc0]             Encryption:prologue - enc0 client/tcloudsba
[pool-9-thread-1][s3eu]      BlobStoreConnection:acquire - GUARD - gr
[pool-9-thread-1][s3eu]     BlobStoreConnection:prologue - s3eu client
[user thread 1][s3eu]      BlobStoreConnection:release - GUARD - clie
[user thread 1][s3eu] BlobStoreConnection:epilogueDone - s3eu client/
[user thread 1][enc0]          Encryption:epilogueDone - enc0 client/
[user thread 1][client]                ICStore:release - GUARD - cl
*** Container creation completed ***
```

Figure 186 - ICStore starts up and creates a bucket on Amazon S3

Again, after that the files are synched into it one after another. Here, the monitored folder is also scanned for updates every 60 seconds.

```
Doing TEST-1.TXT -> test-1.txt
15:46:30,850 |-INFO in ch.qos.logback.core.joran.action.AppenderAction - About t
15:46:31,033 |-INFO in ch.qos.logback.core.joran.action.AppenderAction - Naming
15:46:31,598 |-INFO in ch.qos.logback.core.joran.action.NestedComplexPropertyIA
15:46:32,560 |-INFO in ch.qos.logback.core.joran.action.AppenderAction - About t
15:46:32,571 |-INFO in ch.qos.logback.core.joran.action.AppenderAction - Naming
15:46:32,588 |-INFO in ch.qos.logback.core.joran.action.NestedComplexPropertyIA
15:46:32,615 |-INFO in ch.qos.logback.classic.joran.action.LoggerAction - Settin
15:46:32,620 |-INFO in ch.qos.logback.classic.joran.action.LoggerAction - Settin
15:46:32,625 |-INFO in ch.qos.logback.classic.joran.action.LoggerAction - Settin
15:46:32,630 |-INFO in ch.qos.logback.classic.joran.action.RootLoggerAction - Se
15:46:32,635 |-INFO in ch.qos.logback.core.joran.action.AppenderRefAction - Atta
15:46:32,650 |-INFO in ch.qos.logback.classic.joran.action.ConfigurationAction -
15:46:32,689 |-INFO in ch.qos.logback.classic.joran.JoranConfigurator@59566a6c -
[main][]         BlobStoreConnection:build - BlobStore loaded for provider s3
[main][]         BlobStoreConnection:build - BlobStore loaded for provider filesy
[main][]       SizeHandler:queryModuleTree - Queried enc0 for SizeRequirements: [
[main][]       SizeHandler:queryModuleTree - Queried s3eu for SizeRequirements: [
[main][]                  SizeHandler:call - Max Slice Size for 262144 is 262180
[main][]       SizeHandler:queryModuleTree - Queried enc0 for SizeRequirements: [
[main][]       SizeHandler:queryModuleTree - Queried s3eu for SizeRequirements: [
[main][]                  SizeHandler:call - Max Total Size for 4 is 36
[main][]                  ICStore:prologue - client client/tcloudsbackup/test-1.t
[main][enc0]               Encryption:prologue - enc0 client/tcloudsbackup/test-1
[pool-9-thread-1][s3eu]       BlobStoreConnection:acquire - GUARD - granting the
[pool-9-thread-1][s3eu]       BlobStoreConnection:acquire - GUARD - granting the
[pool-9-thread-1][s3eu]      BlobStoreConnection:prologue - s3eu client/tcloudsba
[user thread 0][s3eu]       BlobStoreConnection:release - GUARD - client/tcloudsb
[user thread 0][s3eu]       BlobStoreConnection:release - GUARD - client/tcloudsb
[user thread 0][s3eu] BlobStoreConnection:epilogueDone - s3eu client/tcloudsback
[user thread 0][enc0]           Encryption:epilogueDone - enc0 client/tcloudsback
[user thread 0][client]                  ICStore:release - GUARD - client/tcloud
[user thread 0][client]                  ICStore:release - GUARD - client/tcloud
[user thread 0][client]              ICStore:epilogueDone - client client/tclouds
*** Upload completed ***
```

Figure 187 - ICStore synchronizes files to cloud storage

Here is the view of the files as they are seen on the Amazon S3 web interface, after synchronization.

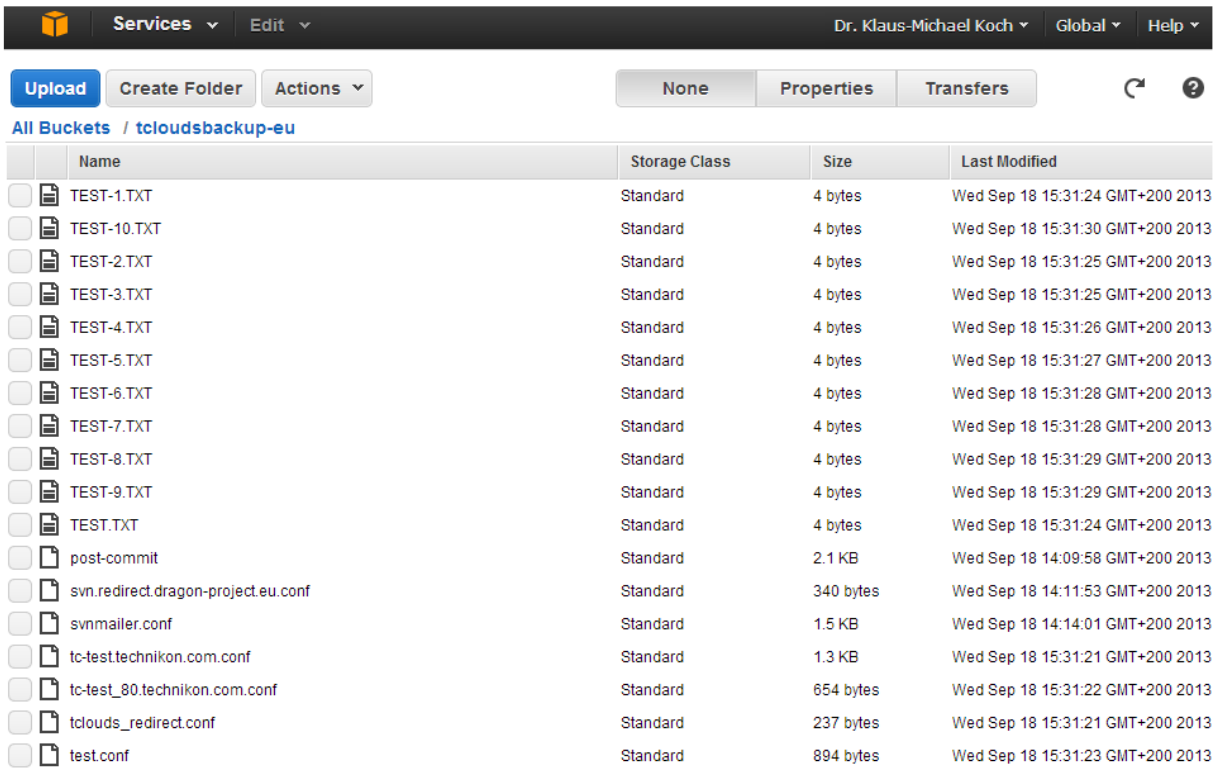


Figure 188 - Files on the Amazon S3 bucket

3.3.1.3.1.3 Trusted Infrastructure

3.3.1.3.2 Introduction

Sirrix AG (SRX) develops within TCLOUDS a platform for trusted cloud computing called "Trusted Infrastructure" (TI). The goal of the platform is to ensure that all instances within the cloud are running on a secure and trusted hardware and therefore being seen as trustworthy. TECHNIKON (TEC) proposed to perform some evaluation of this platform. The goal is to perform measurements scenarios and to propose to the developers findings and if feasible recommendations.

3.3.1.3.3 Starting Position

SRX provided TEC access to the web interface of the Trusted Objects Manager (TOM) and provided a Trusted Desktop (TD) to test and evaluate the platform. A VPN connection between TD and TOM is needed for certain tests. Such a connection has not yet been established by SRX. It was possible to evaluate the features on the TD only. We focussed on the ease of using the device in terms of an end user and investigated its technical features. Access to the TOM was not completely open to us. We had access to the administration web interface or so called dashboard. There we could configure, monitor and manage the cloud.

Trusted Desktop specifications:

Manufacturer	Hewlett Packard
Model	HP EliteBook 8440p
CPU	Intel(R) Core i7 M620
RAM	4096 MB
Screen	14", 1366x768 pixel
OS	TURAYA.SecurityKernel Version 0.50.133
TPM	Yes

Trusted Objects Manager specifications:

OS	TURAYA TrustedObjects Manager 2.3.12 R92
TPM	Yes

3.3.1.3.4 Evaluation Method

The evaluation was performed on four different scenarios. The first three of them were focused on the TD and the other one on the TOM. The scenarios were done step-by-step. First findings and recommendations are charted in the tables below.

Scenario 1: Starting and Stopping of the TD and a compartment

ID	***Step***	***Analysis***
1.1	Start TD	Boot time: about 39 seconds; BIOS is password protected
1.2	Login on TD	Maybe it is unclear for some users what is the meaning of "Organization" and "Session" on the login screen and what they are used for. ***RECOMMENDATION:*** Place an info button or a link where the user can see a short description of the options on mouse-hover.
1.3	Start a compartment	Boot time: about 17 seconds; Sometimes the desktop of a VM is not loading correctly. The screen is

		completely black until a button is pressed but then only the button becomes “normal” and the rest is still black.
1.4	Start a second compartment	Boot Time: about 17 seconds
1.5	Stop a compartment	Click on the Button “Stop”
1.6	Shut down a compartment	Eventually place an extra button for shutdown under “Compartment Management” so that the user doesn’t have to go into the VM to shutdown it.
1.7	Shut down TD	There is no “shutdown”-Button and therefore we were urged to kill the machine with the hardware button.

Scenario 2: Functionality of “Compartment Management” and the TD menu

ID	***Step***	***Analysis***
2.1	Start TD	Boot time: about 39 seconds
2.2	Login on TD	Maybe it is unclear for some users what is the meaning of “Organization” and “Session” on the login screen and what they are used for. ***RECOMMENDATION:*** Place an info button or a link where the user can see a short description of the options on mouse-hover.
2.3	Start a compartment	Select a compartment and click on “Start”
2.4	Stop the compartment	Click on the Button “Stop”
2.5	Reset the compartment via “Compartment Management”	Click on the Button “Reset”
2.6	Check if the reset was working.	Start the compartment and check if it is booting or not.
2.7	Stop the compartment	Click on the Button “Stop”
2.8	Export one compartment on an external USB drive	Time: 22 Minutes for 3,2GB; NTFS is not supported; All data is encrypted as desired; Impossible to select a folder on the USB device where to save the export ***RECOMMENDATION:*** Support NTFS (a lot of HDD are NTFS formatted because of big data, like movies); The user should be able to select a folder where he/she wants to store the backup
2.9	Import the compartment	Only possible with Samba share ***RECOMMENDATION:*** Implement import from USB
2.10	Shutdown TD	There is no “shutdown”-Button and therefore we were urged to kill the machine with the hardware button.

Scenario 3: Functionality of compartments

ID	***Step***	***Analysis***
3.1	Start TD	Boot time: about 39 seconds
3.2	Login on TD	Maybe it is unclear for some users what is the meaning of “Organization” and “Session” on the login screen and what they are used for. ***RECOMMENDATION:*** Place an info button or a link where the user can see a short description of the options on mouse-hover.

3.3	Start TD-Public and TD-Confidential	For a user which is not so used to IT, it might be difficult to handle the Graphical User Interface (GUI) when he/she has no clue what a Virtual Machine (VM) is ***RECOMMENDATION:*** Produce a short video with a tutorial how the TD is working so that every user can work with it.
3.4	Go to one of the compartments	It is impossible to switch directly from the "Settings" tab, which is also the home screen, to the running compartments or to the overview of the running. The ordering of the "Overview" and "Settings" tabs is also a bit confusing. ***RECOMMENDATION:*** Implement the "Settings" tab so that a direct change will work and change the positions of the two tabs.
3.5	Copy something from TD-Confidential to TD-Public	Files don't work Text is automatically encrypted
3.6	Attach USB stick to a compartment	I wasn't able to bring it to work, because the device was not listed in the list of "Available USB devices"
3.7	Test functionality of the other tabs	"Help" was not working (nothing happened) "Sound Control" was not working (nothing happened) "Radio" was not working (nothing happened) "CD-drive" was working well "USB" was working but I wasn't able to attach my USB-stick to a compartment "Battery indicator" was working well "Logout" was working well
3.8	Test the working of compartments	It is depending on the running Operating System (OS), but when the user what to open an application for example in Ubuntu. It could easily happen that he is changing the compartment because the buttons are really close together.
3.9	Stop the compartments	Click on the Button "Stop"
3.10	Shut down TD	Time: about 8 seconds

Scenario 4: GUI evaluation of Trusted Objects Manager

ID	***Step***	***Analysis***
4.1	Connect to the TOM Dashboard via a Web-browser https://134.147.217.69/	The area for the login could be placed more centrally.
4.2	Login on TOM	Insert the credentials. It is good that the user have to wait 30 seconds after entering the wrong credentials three times.
4.3	Try to get an overview of the functions available	The structure of the Web-GUI is very clear and looks well organized. It is a bit old school but administrators will like it and finally they are the persons which will use it. The "Help" column with all its information is - as the name says - very helpful for using the management interface. Nevertheless I have one recommendation. ***RECOMMENDATION:***

ID	Step	Analysis
		A short video with a tutorial or a step-by-step introduction like on some web platforms when you first login would be very helpful for the users to get an overview.
4.4	Navigate through the menu in the left column	The tree view in the left column is very well structured but in larger environments with a lot of locations and a lot of Trusted Virtual Domains (TVD) it maybe will become confusing and difficult to use.
4.5	Open the properties of one appliance	Double-click or Right-click to open the properties
4.6	Download the configuration	Click the left button at the bottom
4.7	Open Advanced settings	Click the right button at the bottom
4.8	Add a location	Right-click on “Locations” in the left menu. It is easy to understand how it works.
4.9	Create a new TVD	Right-click on “Trusted Virtual Domains” in the left menu
4.10	Change password	You have to look twice, but you can find the button in the lower right corner.
4.11	Logout	Right next to Logout
4.12	Test the new password	Insert the new password and see if it works.

3.3.1.3.5 Summary

Our work revealed potential for improvements. Some of them can be implemented without large efforts. The current structure of the Web-GUI on the TOM and the GUI on TD is very clear and looks well organized. Users without a solid technical background will have some difficulties to manage the tools and to acquire an overview quickly. A tutorial with a video or animation will bring it closer to the users and help them a lot, on working with the platform.

3.3.2 FT-BPEL – Validation Activity

This chapter describes the validation of the FT-BPEL subsystem. As stated in chapter 3.2.5.1 of D3.3.3, the validation activity of this subsystem is introduced for the first time in this document, since it is not actively used by the two TClouds scenarios. The main reason behind this is that at their current state neither the Healthcare scenario nor the Smart Lighting System scenario use remote services with such kind that would have been made the application of FT-BPEL reasonable to be used.

Despite this, the introduction of this subsystem within the TClouds Infrastructure allows systems with more complex workflows to be implemented and executed in a reliable and highly available fashion. We can imagine, for example, the use of FT-BPEL within the Healthcare Platform where activities of multiple parties like hospital IT facilities, doctors, and patients have to be coordinated and where core services are involved, for instance, authentication services or services collecting critical patient data.

Below, the validation activity is described that will be executed in the following chapters.

Activity ID	**FTBPEL_1**
Activity type	Proof of concept
Activity description	The scenario for the validation of FT-BPEL is a calculator service written in BPEL which offers a multiply-add operation. The multiply-add operation is implemented by invoking at first a multiplier service and

	then an adder service. 1) # Start a screen environment ./rbpel.bash screen 2) # Configure calculator scenario for the unreplicated (standard) case ./rbpel.bash setup_scen -f 0 calc 3) # Start system: # - adder and multiplier services (service group 1, host 1) # - single BPEL engine hosting the calculator service (service group 2, host 1) # - client ./rbpel.bash start http://services.net/calculator 4) # Simulate crash of the adder and multiplier service instances # -> Requests of the client cannot be processed anymore ./rbpel.bash kill_host 1 1 5) # Terminate all running instances and delete the tmp (/tmp/rbpel-`whoami`/) directory containing the log files ./rbpel.bash cleanup -t 6) # Repeat the test but this time kill the BPEL engine # -> Again, the service is rendered unavailable through the (simulated) crash of a single machine ./rbpel.bash start http://services.net/calculator ./rbpel.bash kill_host 2 1 ./rbpel.bash cleanup -t 7) # Configure calculator scenario for the replicated case (using the FT-BPEL platform) ./rbpel.bash setup_scen -f 1 calc 8) # Start system: # - ZooKeeper service # - adder and multiplier services (service group 1, hosts 1 to 3) # - single BPEL engine hosting the calculator service (service group 2, hosts 1 to 3) # - client ./rbpel.bash start http://services.net/calculator 9) # Simulate crash of one adder and one multiplier service instance # -> Service remains available ./rbpel.bash kill_host 1 1 10) # Simulate crash of one BPEL engine replica # -> Service is still available ./rbpel.bash kill_host 2 1 ./rbpel.bash cleanup -t
Acceptance Criteria	At step 3 the system works properly and the simple calculator does its calculations correctly

	At step 4 and 6 the system should crash (since there are no replica) After step 9 and 10 the system continues to work since replicas are active

Table 47 - FTBPEL_1 validation activity

3.3.2.1 Validation scenario

The scenario for the validation of RBPEL is a calculator service written in BPEL which offers a multiply-add operation. The multiply-add operation is implemented by invoking at first a multiplier service and then an adder service. All services are hosted within Tomcat instances. For the execution of the BPEL processes, Apache ODE is employed. ODE in turn is implemented as a Web application also hosted by means of Tomcat. To simulate multiple machines or replicas, several Tomcat instances are started.

The client constantly issues multply-add requests with changing arguments. The results are checked and the client would stop immediately if it detected a faulty calculation. From the perspective of the client application, it does not make any difference whether the service is replicated or not. All the work necessary for the replicated case is carried out by a proxy which is configured to intercept Web service invocations.

3.3.2.2 Validation setup

The validation activity has been held in San Raffaele facilities, the architecture has been hosted directly on one single machine. The following deployment scenario has been used for FT-BPEL validation activity:

Machine Type: Virtual Machine
Operative System: Ubuntu 12.04
CPU: Quad Core 64Bit
RAM: 2GB
HD: 20GB
Required SW: Java virtual machine.

Description: The FT-BPEL software has been installed in a specific directory into the file system. FT-BPEL comes in a preconfigured zip package. The configuration for the validation involves a three-time replicated platform, that is, three replicas of each component of the platform are used. On top of the platform a composed calculator service is executed.

The package contains also a client has been set up to issue continuously new calculator requests.

The FT-BPEL calculator service has been configured in such a way that it can be started in an easy and straightforward manner with a simple API. As you start the bundle system, it will automatically set up the replicas of the BPEL engines, the replicas of the basic Web services, the employed Apache ZooKeeper service, and the client.

```
#./rbpel.bash start http://services.net/calculator
```

The system generates an extensive log file set. It allows direct and easy access to understand the behavior of the whole system.

3.3.2.3 Validation execution

In order to execute properly the validation, some specific care has been taken to monitor all the output fluxes and resource monitoring.

More specifically:

- The output of all created screens has been logged in order to easily process them. Since all the output runs on Linux “screens”, the screen program has been started with the –L option:

```
$sudo screen –L
```

In order to run the FT-BPEL protocol properly, some initial setup has been made:

```
# Start all components on localhost
./rbpel.bash setup_hosts local

# Enable logging
./rbpel.bash setup_logging con
```

Than the screen program has been started in order to allow the bundle software to work:

```
$sudo screen –L
```

The validation scenario is split in two parts: the first shows the effective work of the BPEL calculator, the second shows the replication ability.

In order to run the first part (the standard BPEL, non-replicated) it is necessary to issue the following commands:

```
# Configure calculator scenario for the unreplicated (standard) case
./rbpel.bash setup_scen -f 0 calc

# Start system:
# - adder and mutliplier services (service group 1, host 1)
# - single BPEL engine hosting the calculator service (service group 2, host 1)
# - client
./rbpel.bash start http://services.net/calculator
```

The above command will:

- Start the calculator services
- Start the client to continuously issue calculation requests in BPEL fashion

At this stage, the system is up and running. It is possible to see it running by watching the output file, we can see that the client is correctly sending and retrieving data:

```
Load hosts from '/home/hsr/rbpel-valid/bin/config/hosts.config.local'
Start client 10 on localhost with 1 threads
Start now!
sc.nclients          1
sc.nwarmclients      1
sc.warmup            0
sc.run               -1
sc.pause             10
Warm up...
Get serious!
Test run with 1 clients, -1 secs
    1 cnt        7  time   927826  avg   132546  min    96038  max   255744
    2 cnt       10  time  1048087  avg   104808  min    87362  max   123844
    3 cnt       10  time   975455  avg    97545  min    88900  max   102899
    4 cnt       10  time  1043924  avg   104392  min    61471  max   151464
    5 cnt       10  time   967743  avg    96774  min    88482  max   108745
    6 cnt       10  time  1016693  avg   101669  min    94405  max   111912
    7 cnt       11  time  1015931  avg    92357  min    86704  max    98195
    8 cnt       10  time   958824  avg    95882  min    92368  max    99671
```

```
    9 cnt       11  time  1060579  avg    96416  min    88135  max   119292
   10 cnt       10  time   946849  avg    94684  min    88016  max    96567
   11 cnt       11  time  1020119  avg    92738  min    88177  max   102404
```

Listing 48 - Snippet of client output

Each line has the following form:

```
240        cnt   391        time 951402   avg   2433       min   1001       max  94057
second | number of service invocations | total time in [µSec] | average time in [µSec] | fastest request [µSec] | slowest request [µSec]
```

We can also inspect cpu consumption by each party (client, BPEL engine and calculator services)

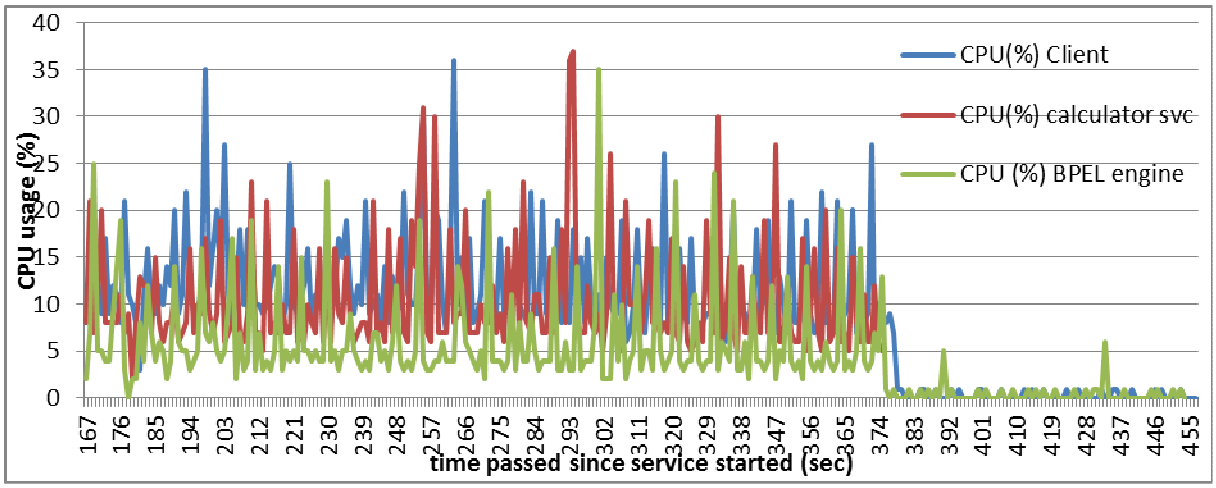


Figure 189 - CPU consumption for non-replicated BPEL system and failure of the calculator service

The system works as expected and the BPEL engine orchestrates properly the calculator services in order to provide the right result:

```
Load hosts from '/home/hsr/rbpel-valid/bin/config/hosts.config.local'
Start client 10 on localhost with 1 threads
Start now!
sc.nclients          1
sc.nwarmclients      1
sc.warmup            0
sc.run               -1
sc.pause             10
Warm up...
Get serious!
Test run with 1 clients, -1 secs
    1 cnt        7  time   927826  avg   132546  min    96038  max   255744
    2 cnt       10  time  1048087  avg   104808  min    87362  max   123844
    3 cnt       10  time   975455  avg    97545  min    88900  max   102899
    4 cnt       10  time  1043924  avg   104392  min    61471  max   151464
    5 cnt       10  time   967743  avg    96774  min    88482  max   108745
    6 cnt       10  time  1016693  avg   101669  min    94405  max   111912
    7 cnt       11  time  1015931  avg    92357  min    86704  max    98195
    8 cnt       10  time   958824  avg    95882  min    92368  max    99671
    9 cnt       11  time  1060579  avg    96416  min    88135  max   119292
   10 cnt       10  time   946849  avg    94684  min    88016  max    96567

               11 cnt       11  time  1020119  avg    92738  min    88177  max   102404
```

Figure 190 - client output of non-replicated BPEL system. The client receives correctly the calc result

3.3.2.3.1.1 Crash simulation

At this point we can try to crash the basic Web services the composed calculator service is based on. We have to issue the following command:

```
# Simulate crash of the adder and multiplier service instances
# -> Requests of the client cannot be processed anymore
./rbpel.bash kill_host 1 1
```

As expected, the system has not been able to provide any valid output anymore. We can see that the system has stopped working by either inspecting the CPU consumption in Figure 189 or by watching the client output:

```
371 cnt       11  time   964383  avg    87671  min    84845  max    88923
  372 cnt       12  time  1050934  avg    87577  min    83384  max    91429
  373 cnt       11  time   975887  avg    88717  min    83548  max   105433
  374 cnt       11  time   971429  avg    88311  min    87235  max    91583
javax.xml.ws.soap.SOAPFaultException: java.net.ConnectException: Connection refused
        at com.sun.xml.internal.ws.fault.SOAP11Fault.getProtocolException(SOAP11Fault.java:178)
        at com.sun.xml.internal.ws.fault.SOAPFaultBuilder.createException(SOAPFaultBuilder.java:111)
        at com.sun.xml.internal.ws.client.sei.SyncMethodHandler.invoke(SyncMethodHandler.java:108)
        at com.sun.xml.internal.ws.client.sei.SyncMethodHandler.invoke(SyncMethodHandler.java:78)
        at com.sun.xml.internal.ws.client.sei.SEIStub.invoke(SEIStub.java:129)
        at com.sun.proxy.$Proxy22.multiplyadd(Unknown Source)
        at eu.tclouds.rbpel.demo.calculator.CalculatorProxy.invoke(CalculatorProxy.java:25)
        at eu.tclouds.rbpel.demo.ServiceProxy.invoke(ServiceProxy.java:41)
        at
eu.tclouds.rbpel.demo.ThroughputClientBase$Worker.invokeService(ThroughputClientBase.java:268)
        at eu.tclouds.rbpel.demo.ThroughputClientBase$Worker.run(ThroughputClientBase.java:219)
  375 cnt        5  time   459318  avg    91863  min    84217  max   102766
  376 cnt        0  time        0  avg        0  min        0  max        0
  377 cnt        0  time        0  avg        0  min        0  max        0
  378 cnt        0  time        0  avg        0  min        0  max        0
  379 cnt        0  time        0  avg        0  min        0  max        0
```

As we can see after second 375 the system has not been able anymore to provide a valid output.

We have also reproduced the same behavior by crashing the BPEL engine. These are the results:

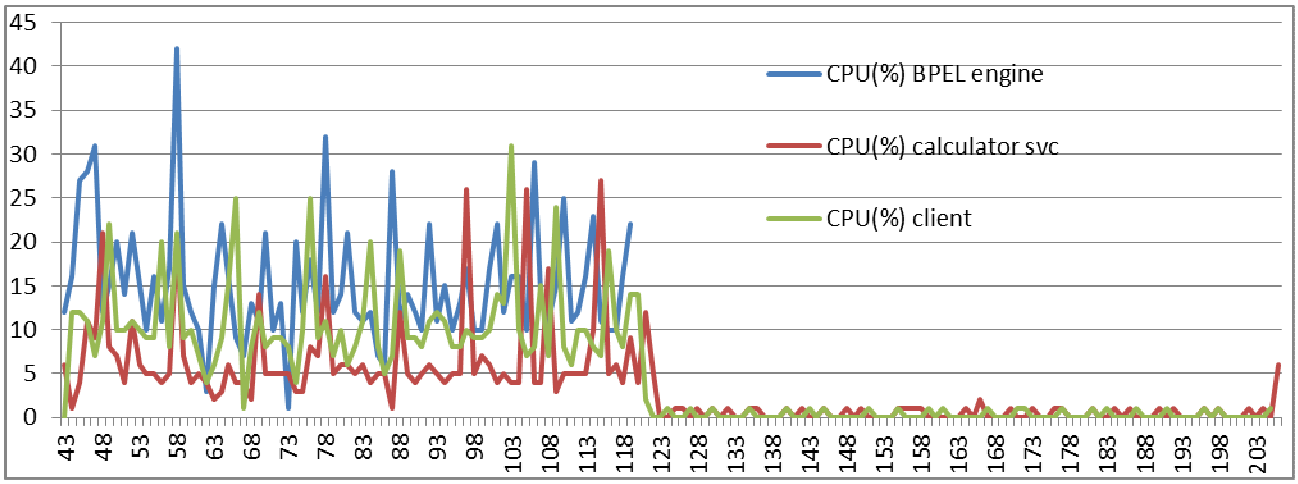


Figure 191 - resource usage and crash of BPEL engne

```
117 cnt       11  time  1007959  avg    91632  min    88449  max    92825
  118 cnt        7  time   635256  avg    90750  min    84075  max    96779
javax.xml.ws.WebServiceException: java.net.ConnectException: Connection refused
        at
com.sun.xml.internal.ws.transport.http.client.HttpClientTransport.readResponseCodeAndMessage(HttpClientTranspo
rt.java:196)
        at
com.sun.xml.internal.ws.transport.http.client.HttpTransportPipe.createResponsePacket(HttpTransportPipe.java:21
2)
        at com.sun.xml.internal.ws.transport.http.client.HttpTransportPipe.process(HttpTransportPipe.java:203)
        at
com.sun.xml.internal.ws.transport.http.client.HttpTransportPipe.processRequest(HttpTransportPipe.java:122)
        at
com.sun.xml.internal.ws.transport.DeferredTransportPipe.processRequest(DeferredTransportPipe.java:95)
        at com.sun.xml.internal.ws.api.pipe.Fiber.__doRun(Fiber.java:626)
        at com.sun.xml.internal.ws.api.pipe.Fiber._doRun(Fiber.java:585)
        at com.sun.xml.internal.ws.api.pipe.Fiber.doRun(Fiber.java:570)
        at com.sun.xml.internal.ws.api.pipe.Fiber.runSync(Fiber.java:467)
        at com.sun.xml.internal.ws.client.Stub.process(Stub.java:308)
        at com.sun.xml.internal.ws.client.sei.SEIStub.doProcess(SEIStub.java:146)
        at com.sun.xml.internal.ws.client.sei.SyncMethodHandler.invoke(SyncMethodHandler.java:98)
        at com.sun.xml.internal.ws.client.sei.SyncMethodHandler.invoke(SyncMethodHandler.java:78)
        at com.sun.xml.internal.ws.client.sei.SEIStub.invoke(SEIStub.java:129)
        at com.sun.proxy.$Proxy22.multiplyadd(Unknown Source)
        at eu.tclouds.rbpel.demo.calculator.CalculatorProxy.invoke(CalculatorProxy.java:25)
        at eu.tclouds.rbpel.demo.ServiceProxy.invoke(ServiceProxy.java:41)
        at eu.tclouds.rbpel.demo.ThroughputClientBase$Worker.invokeService(ThroughputClientBase.java:268)
        at eu.tclouds.rbpel.demo.ThroughputClientBase$Worker.run(ThroughputClientBase.java:219)
Caused by: java.net.ConnectException: Connection refused
        at java.net.PlainSocketImpl.socketConnect(Native Method)
        at java.net.AbstractPlainSocketImpl.doConnect(AbstractPlainSocketImpl.java:339)
        at java.net.AbstractPlainSocketImpl.connectToAddress(AbstractPlainSocketImpl.java:200)
        at java.net.AbstractPlainSocketImpl.connect(AbstractPlainSocketImpl.java:182)
        at java.net.SocksSocketImpl.connect(SocksSocketImpl.java:392)
        at java.net.Socket.connect(Socket.java:579)
        at java.net.Socket.connect(Socket.java:528)
        at sun.net.NetworkClient.doConnect(NetworkClient.java:180)
        at sun.net.www.http.HttpClient.openServer(HttpClient.java:378)
        at sun.net.www.http.HttpClient.openServer(HttpClient.java:473)
        at sun.net.www.http.HttpClient.parseHTTPHeader(HttpClient.java:709)
        at sun.net.www.http.HttpClient.parseHTTP(HttpClient.java:579)
        at sun.net.www.protocol.http.HttpURLConnection.getInputStream(HttpURLConnection.java:1322)
        at java.net.HttpURLConnection.getResponseCode(HttpURLConnection.java:468)
        at
com.sun.xml.internal.ws.transport.http.client.HttpClientTransport.readResponseCodeAndMessage(HttpClientTranspo
rt.java:192)
        ... 18 more
  119 cnt        0  time        0  avg        0  min        0  max        0
  120 cnt        0  time        0  avg        0  min        0  max        0
  121 cnt        0  time        0  avg        0  min        0  max        0
  122 cnt        0  time        0  avg        0  min        0  max        0
```

3.3.2.3.1.2 Run replicated BPEL system

The replicated BPEL system differs from the standard case in that for each component (BPEL engine executing the composed calculator service and basic Web services) there are three replicas executing the corresponding service.

In this new set-up the system becomes fault tolerant and able to resist the faults of one BPEL engine replica and one replica of all involved basic Web services.

In order to start the replicated service, we have to set up the system by invoking the command:

```
# Configure calculator scenario for the replicated case (using the RBPEL platform)
./rbpel.bash setup_scen -f 1 calc
```

We can start the system by issuing the same command as in the previous run:

```
# Start system:
# - ZooKeeper service
# - adder and mutliplier services (service group 1, hosts 1 to 3)
# - single BPEL engine hosting the calculator service (service group 2, hosts 1 to 3)
# - client
./rbpel.bash start http://services.net/calculator
```

At this point we can see all the replicas running and providing the right output to the client:

```
Load hosts from '/home/hsr/rbpel-valid/bin/config/hosts.config.local'
Start client 10 on localhost with 1 threads
Start now!
sc.nclients          1
sc.nwarmclients      1
sc.warmup            0
sc.run               -1
sc.pause             10
2013-09-08     22:46:31,542     [myid:]     -     INFO          [main:Environment@100]     -     Client
environment:zookeeper.version=3.4.5-1392090, built on 09/30/2012 17:52 GMT
2013-09-08 22:46:31,548 [myid:] - INFO  [main:Environment@100] - Client environment:host.name=ubuntu-
rbpel
2013-09-08     22:46:31,548     [myid:]     -     INFO          [main:Environment@100]     -     Client
environment:java.version=1.7.0_25
2013-09-08 22:46:31,552 [myid:] - INFO  [main:Environment@100] - Client environment:java.vendor=Oracle
Corporation

[…LINES OMITTED FOR SAKE OF READABILITY …]

2013-09-08 22:46:33,589 [myid:] - INFO  [Thread-1-SendThread(localhost:4888):ClientCnxn$SendThread@849]
- Socket connection established to localhost/127.0.0.1:4888, initiating session
2013-09-08          22:46:33,599          [myid:]          -          INFO                    [Thread-1-
SendThread(localhost:4888):ClientCnxn$SendThread@1207]  -  Session  establishment  complete  on  server
localhost/127.0.0.1:4888, sessionid = 0x140ff5353e7000c, negotiated timeout = 4000
~ 2013-09-08 22:46:33.599 WatchedEvent state:SyncConnected type:None path:null
~    2013-09-08    22:46:33.601    Created    ZooKeeper    connection    State:CONNECTED    Timeout:4000
sessionid:0x140ff5353e7000c  local:/127.0.0.1:57698  remoteserver:localhost/127.0.0.1:4888  lastZxid:0
xid:1      sent:1      recv:1      queuedpkts:0      pendingresp:0      queuedevents:0      -      constr
localhost:4888,localhost:4889,localhost:4890, time 17289 µs
    1 cnt        0  time        0  avg        0  min        0  max        0
    2 cnt        1  time  1927305  avg  1927305  min  1927305  max  1927305
    3 cnt        3  time  1007139  avg   335713  min   327363  max   349776
    4 cnt        2  time   866096  avg   433048  min   389669  max   476426
    5 cnt        3  time  1033179  avg   344393  min   248899  max   453958
    6 cnt        3  time  1027364  avg   342454  min   304118  max   393276
    7 cnt        3  time  1004662  avg   334887  min   280279  max   392293
    8 cnt        0  time        0  avg        0  min        0  max        0
    9 cnt        3  time  1901808  avg   633936  min   270176  max  1309669
   10 cnt        4  time  1184588  avg   296147  min   227328  max   345918
```

3.3.2.3.1.3 Crash simulation

Also in this case we have simulated a crash of single machines. We simply killed two processes that refer to a BPEL engine replica and to a replica of the basic Web services.

In order to crash the calculator replica, we issued the command:

```
# Simulate crash of one adder and one multiplier service instance
# -> Service remains available
./rbpel.bash kill_host 1 1
```

Despite the crash, the system is able to maintain its integrity and the client receives the correct results:

```
305 cnt        4  time  1073145  avg   268286  min   248143  max   288307
  306 cnt        4  time   991116  avg   247779  min   215997  max   304227
  307 cnt        4  time  1028675  avg   257168  min   183980  max   308983
  308 cnt        3  time   847121  avg   282373  min   258666  max   309047
  309 cnt        4  time  1081217  avg   270304  min   214187  max   348624
```

```
  310 cnt        4  time  1033274  avg   258318  min   208524  max   295653
  311 cnt        3  time   798401  avg   266133  min   229408  max   320003
  312 cnt        2  time  1187282  avg   593641  min   584394  max   602887
  313 cnt        0  time        0  avg        0  min        0  max        0
  314 cnt        1  time  1587576  avg  1587576  min  1587576  max  1587576
  315 cnt        1  time   936740  avg   936740  min   936740  max   936740
  316 cnt        1  time   643573  avg   643573  min   643573  max   643573
  317 cnt        2  time  1842705  avg   921352  min   763395  max  1079309
  318 cnt        1  time   857484  avg   857484  min   857484  max   857484
  319 cnt        3  time  1041368  avg   347122  min   213071  max   492931
```

Than we crashed also the BPEL engine replica:

```
# Simulate crash of one BPEL engine replica
# -> Service is still available
# (If the leader of the three BPEL engine replicas is killed, the current configuration detects this
crash
# within 4 seconds. This leads to a delay of 4 seconds for all requests processed at the time of the
crash.
# After the reconfiguration of the system, all requests are processed as before the crash.)
./rbpel.bash kill_host 2 1
```

With the following outcome of the client:

```
421 cnt        3  time   808824  avg   269608  min   248314  max   311546
  422 cnt        4  time  1262195  avg   315548  min   257264  max   355962
  423 cnt        3  time   956527  avg   318842  min   278928  max   389364
  424 cnt        3  time   842859  avg   280953  min   255396  max   324587
~ 2013-09-08 22:53:39.931 OP-5096c789-91e0-4031-a72e-a4f8c5e5dee4: Error while invoking proxy e558b690-
110d-4581-93e5-0c18c519874a         for         request         de80130576284a52_b30d68922755ea71/001366
javax.xml.ws.WebServiceException:                                 java.util.concurrent.ExecutionException:
javax.xml.ws.WebServiceException: java.net.ConnectException: Connection refused
  425 cnt        4  time  1244092  avg   311023  min   233339  max   361013
  426 cnt        1  time   247192  avg   247192  min   247192  max   247192
~ 2013-09-08 22:53:41.607 OP-5096c789-91e0-4031-a72e-a4f8c5e5dee4: Error while invoking proxy e558b690-
110d-4581-93e5-0c18c519874a         for         request         de80130576284a52_b30d68922755ea71/001369
javax.xml.ws.WebServiceException:                                 java.util.concurrent.ExecutionException:
com.sun.xml.internal.ws.client.ClientTransportException:          HTTP          transport          error:
java.net.ConnectException: Connection refused
  427 cnt        2  time  1476079  avg   738039  min   399123  max  1076955
~ 2013-09-08 22:53:43.120 OP-5096c789-91e0-4031-a72e-a4f8c5e5dee4: Error while invoking proxy e558b690-
110d-4581-93e5-0c18c519874a         for         request         de80130576284a52_b30d68922755ea71/001373
javax.xml.ws.WebServiceException:                                 java.util.concurrent.ExecutionException:
com.sun.xml.internal.ws.client.ClientTransportException:          HTTP          transport          error:
java.net.ConnectException: Connection refused
  428 cnt        3  time  1173748  avg   391249  min   340373  max   482156
~ 2013-09-08 22:53:44.026 OP-5096c789-91e0-4031-a72e-a4f8c5e5dee4: Error while invoking proxy e558b690-
110d-4581-93e5-0c18c519874a         for         request         de80130576284a52_b30d68922755ea71/001376
javax.xml.ws.WebServiceException:                                 java.util.concurrent.ExecutionException:
com.sun.xml.internal.ws.client.ClientTransportException:          HTTP          transport          error:
java.net.ConnectException: Connection refused
  429 cnt        3  time   904231  avg   301410  min   240390  max   334402
  430 cnt        4  time  1246309  avg   311577  min   283096  max   371293
  431 cnt        3  time   892093  avg   297364  min   244448  max   358149
```

Although two crashes of machines had been simulated, the FT-BPEL system was still able to provide the correct service.

3.3.2.4 Conclusion

Examining FT-BPEL behavior, with its ability to be tolerate crashes within each employed component, we can assess that RBPEL_1 validation activity has SUCESSFULLY PASSED

3.4 Summary tables of activities

In this chapter we are going to summarize all the validation activities performed and we will map the Healthcare and Smart Light System scenario's requirements by recalling table 1 on page 24 of D3.3.3:

requirement	LREQ1	LREQ2	LREQ3	LREQ4	LREQ5	AHSECREQ1	AHSECREQ2	AHSECREQ3	AHSECREQ4	AHSECREQ5	AHSECREQ6	AHSECREQ7	AHSECREQ8	ASSECREQ1	ASSECREQ2	ASSECREQ3	ASSECREQ4	ASSECREQ5	ASSECREQ6	OUTCOME
Validation activity																				
SBS + SVM 1	x	x				x	x													100%
LogService 1	x		x	x	x						x	x		x						100%
CheapBFT 1		x					x	X	x	x			x							100%
DepSky 1	x	x				x	x	x	x	x			X							100%
DepSky 2	x	x				x	x	X	x	x			x							100%
ACaaS 1			x																	100%
Remote 1					x						x	x		X						100%
Ontology_1	x	x				x				x	x									100%
Memcached_1								X												100%
Memcached_2								X												100%
SAVE_1					x															100%
Integration 1															x					100%
Integration 2															x					100%
Integration 3															x					100%
Integration 4																	x			100%
Integration 5																		x		100%
Integration 6																			x	100%
Integration 7																			x	100%
Trusted O 1															x					100%
Trusted O 2															x					100%
Trusted O 3															x	x				100%
Trusted S 1	x														x					100%
Trusted S 2	x														x					100%
Trusted S 3	x															x				100%
Trusted C 1	x		x															x		100%
Trusted C 2	x		x															x		100%
BFT-Smart 1		x														x	x			100%
BFT-Smart 2		x														x	x			100%
BFT-Smart 3		x														x	x	x		100%
BFT-Smart 4		x														x	x			100%
BFT-Smart 5		x														x	x	x		100%

Chapter 4

Conclusion

Chapter Authors:
Marco Abitabile (FCSR)

This chapter wraps up all the results and conclusions derived by the surveys and validation activities. During the TClouds project both Healthcare and Smart Lighting System Scenarios had the chance to understand deeply the implications of use a cloud technology considering their respective data management issues and security/performances/trustworthiness needs.

TClouds infrastructure has shown capabilities that are still not achieved by any other cloud technology and responds to the real needs of industry.

The TClouds project has created a solid base for trustworthiness in cloud technology, and after our validation activities, A3 noticed that TClouds Infrastructure looks more as an "ecosystem" or features that can be combined and used separately to assess specific cloud user needs. Moreover, since TClouds Infrastructure is at prototype level, some extra work needs to be done in order build products. Some subsystems look mature and stable while others still need an extra effort. In the next chapter we will describe the pitfalls and strength of all the subcomponents used.

Being trustworthy

One of the main objectives of TClouds is to be "trustworthy".

Being trustworthy has several meanings. We can list some synonym of "trustworthiness" to have a better clue of what it means to be:

accurate, authentic, authoritative, believable, convincing, credible, honest, honorable, mature, principled, realistic, responsible, all right, always there, exact, open, rock, secure, there, valid

source: http://thesaurus.com/browse/trustworthy

In TClouds we have created two main "trust models". The former, which is used by the Healthcare scenario, considers the cloud owner as someone to trust a-priori. In healthcare realities, hospitals are unwilling to place their data into the cloud for three main reasons: they need to have their data legally compliant, they need to have a clear exit strategy, they need to have a strong relationship with the cloud partner since huge hospitals systems, once in place and cloud-ready, cannot be stopped anymore. Hospital systems tend to be "evolving systems" that embrace simultaneously legacy and brand new IT features. TClouds, with its Trustworthy OpenStack prototype is able to address healthcare needs under many aspects, from legal to technical, to geo-location of data. Trustworthy OpenStack Prototype, assumes the cloud customer has to trust the cloud owner. In the healthcare scenario this complicity between a healthcare institution and the cloud owner is accepted and well seen, since it is the "level of trust" that healthcare needs.

Given all the features provided by Trustworthy OpenStack, the healthcare industry can start facing cloud needs with less fear when considering TClouds infrastructure.

TClouds Trusted Infrastructure component set and BFT-SMaRt, provides the second “trust model”, with high resiliency, availability and confidentiality to the Smart Lighting scenario, enabling such critical solutions a feasible option to be hosted on a cloud environment.

The tight control, and isolation often required by Smart Grid solutions that usually rely on a utility private network and dedicated datacentres, may now take advantage of public cloud environments economy of scale and high scalability, without endangering confidentiality requirements.

Also, the redundancy levels required by these systems have now an alternative option with TClouds BFT-SMaRt improved availability and increased resiliency.

4.1 TODO list & waiting list

This conclusive chapter discusses all the TClouds components that have been used by the two scenarios (Healthcare and Smart Lighting System) under the spotlight of the potentialities and future work that can be done. This chapter aims at offering valuable and constructive suggestions in order to move towards a product-oriented development, proposing useful features and enlightening pitfalls to be fixed.

Crypto as a Service (CaaS)

CaaS is one of the shining TClouds subcomponents as seen from A3 perspective. It provides solid trust models that goes beyond the Trustworthy OpenStack trust model that provides and increases the overall “trustworthiness” of the infrastructure.

While doing the validation activities we did not notice any issues nor things to do. The concept looks valid and constitutes a good candidate for OpenStack extension. At the time of writing, ram memory of virtual machines is not encrypted, however, this does not seems to be an issue since cloud administrators do not have access to ram memory of virtual machines.

Access Control as a Service (ACaaS)

ACaaS provides a very business-valid feature that enables the Healthcare platform to provide value added features to professional customers such as hospitals and clinics, allowing them to ensure physical location of data. At this stage this component is able to control the deployment of pre-set hosts, therefore the healthcare institution needs to know the location of the physical machine before deploying the VM (thus, being adherent with the Trustworthy OpenStack trust model). However, if we imagine a vast cloud system composed of hundreds of physical nodes running the Healthcare Platform, it would be necessary to provide automated geo-location capabilities of the host. This task looks extremely difficult and not solvable in a naïve way, nonetheless this feature would be able to provide a significant added value to the cloud infrastructure.

Ontology Based Reasoner

For any cloud customer, isolation of tenants plays an important role. TVD sub-component satisfies this need. We have not seen any major issue while performing validation activities of TVD subcomponent.

Remote Attestation (RA)

The validation activity of RA sub-component showed its abilities to keep track of system changes at the level of packages installed into the system. This feature looks very interesting, and theoretically works even at VM level. This can provide an extra layer of

security and assurance also at the Healthcare users' level. While performing the validation task we did not notice any particular drawbacks, except that it would be useful to have at OpenStack UI level the capabilities to perform remote attestation (and, if possible, provide REST-like API)

Cheap-BFT

Cheap-BFT sub-component has a very high potential since having live replica of the healthcare application means increasing the fault tolerance of the entire system. Of course the main drawback of this component is the resource usage (you need three times more the resource you have with a single replica) but it can create an extremely robust system, capable of byzantine attacks. Systems of such capabilities are still not available in the cloud scenario and can provide 100% availability for those very critical systems (such as First Aid systems). While performing the validation activity we noticed that not all of the "byzantine" faults are still tolerated, but the effort to make the system complete "byzantine" does not look like a long process. The inevitable drawback of this component is that it is not transparent at application level and the healthcare platform should be properly crafted to make the system work completely.

DepSky

DepSky features resulted in a high level service for VMs applications. DepSky has very promising capabilities. Moreover its abilities of being fault tolerant of byzantine failures and data cyphers makes it very appealing at end-user usage.

While performing the validation activities we noticed some nice-to-have features that should be implemented before going to market: data recovery of damaged replica and data span over multiple clouds as an extension of hard drive storage.

LogService

LogService features enable TPaaS platform to provide a trusted source while inspecting and doing forensics. The "as a Service" nature of LogService allows TPaaS to provide such features up to the final user and give enhanced added value for their third party applications and activities. In order to have a complete product that can be used extensively from many different tenants we suggest to extend Log Service features with multiple tenants support and make use of a proper database in order to store log entries.

Tailored Memcached

Caching features are always used in large systems. Tailored Memcached resulted as the best option within TClouds infrastructure in order to have a caching service that can be even used to span cached data among different VMs. Moreover its lightweight dimension makes it very attractive from the Cloud customer point of view.

Trusted Infrastructure

The component set of TClouds Trusted Infrastructure has been validated as working seamlessly and flawlessly as advertised, providing a high level of confidentiality to critical solutions.

BFT-SMaRt

As an alternative method to usual clustering methods, BFT-SMaRt, provides a concise resilient solution ready for distributed cloud environments. Even though the response times seen during the validation activities are not compatible with near-real time Smart Grid systems, it's believed this is mostly due to network latencies between clouds than BFT-SMaRt implementation itself. Also of notice, is the current limited compliance to SQL and RDBMS construction set (such as joins, and views).

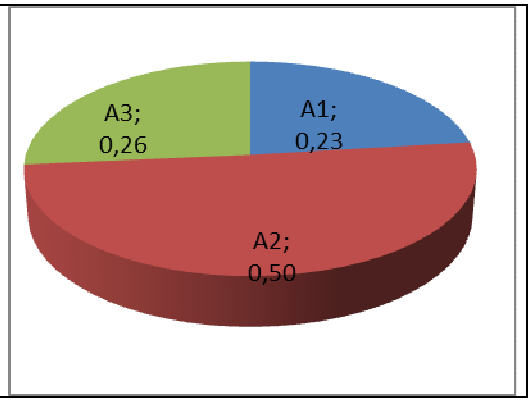
We are certain though, that all these limitations can be overcome with continued investment into the component development.

Appendix 1 – Healthcare survey' score calculation

Surveys' score has been calculated adopting the strategy explained in D3.3.3. Let's take an example:

QUESTION 1:
If you were using an internet platform to log, save and share your patients' data, how important would the following topics be to you?

Score	Answer		1st	2nd	3rd
3,69	A1		3	2	8
8,08	A2		8	5	0
4,23	A3		2	6	5
	Score		10	5	1



This question received 13 total answers (that is, there has been 13 people interviewed). We can see it by summing one of the 3 columns (1st, 2nd or 3rd).

1st column says that there has been a certain amount of people to choose A1/A2/A3 answer as the first answer (in our example there has been 3 people choosing A1 answer, 8 people choosing A2 answer and 2 people choosing A3 answer).

2nd column says that there has been people choosing the answer as second option (in our example: 2 people choose A1 as second option, 5 people choose A2 as second option and 6 people choose A3 as second option) and so on with the other columns.

The "score" row indicates the weight applied for each rank. Scoring system start from 10 for 1st answer to 1 for the last answer.

The final score (in the "Score" column, instead) is calculated by averaging the weighted sum that each Answer has obtained. Weighted sun is given by:

$$\sum_{i=1}^{max(ranking)} rankScore_i * peopleAnswering$$