

Private Public Partnership Project (PPP)

Large-scale Integrated Project (IP)



fi-ware

D.8.3.2: FI-WARE Installation and Administration Guide

Project acronym: FI-WARE

Project full title: Future Internet Core Platform

Contract No.: 285248

Strategic Objective: FI-ICT-2011.1.7 Technology foundation: Future Internet Core Platform

Project Document Number: ICT-2011-FI-285248-WP8-D.8.3.2

Project Document Date: 2013-04-30

Deliverable Type and Security: Public

Author: FI-WARE Consortium

Contributors: FI-WARE Consortium

1.1 Executive Summary

This document describes the installation and administration process of each Generic Enabler developed within in the "Security" chapter. The system requirements for the installation of a Generic Enabler are outlined with respect to necessary hardware, operating system and software. Each GE has a section dedicated to the software installation and configuration process as well as a section, which describes sanity check procedures for the system administrator to verify that the GE installation was successful.

1.2 About This Document

The "FI-WARE Installation and Administration Guide" comes along with the software implementation of components, each release of the document referring to the corresponding software release (as per D.x.2), to facilitate the users/adopters in the installation (if any) and administration of components (including configuration, if any).

1.3 Intended Audience

The document targets system administrators as well as system operation teams of FI-WARE Generic Enablers from the FI-WARE project.

1.4 Chapter Context

The overall ambition of the Security Architecture of FI-WARE is to demonstrate that the Vision of an Internet that is "secure by design" is becoming reality. Based on achievements to date and/or to come in the short-term (both from a technological but also a standardization perspective) we will show that "secure by design" is possible for the most important core (basic) and shared (generic) security functionalities as anticipated by the FI-WARE project and in accordance with the requirements of external stakeholders and users such as the FI PPP Use Case projects. The "secure by design" concept will, therefore, address both the security properties of the FI-WARE platform itself and the applications that will be built on top of it.

In this section the foreseen high-level functional architecture is described, introducing the main modules and their expected relationships, then depicting the most important modules in detail along with their main functionalities.

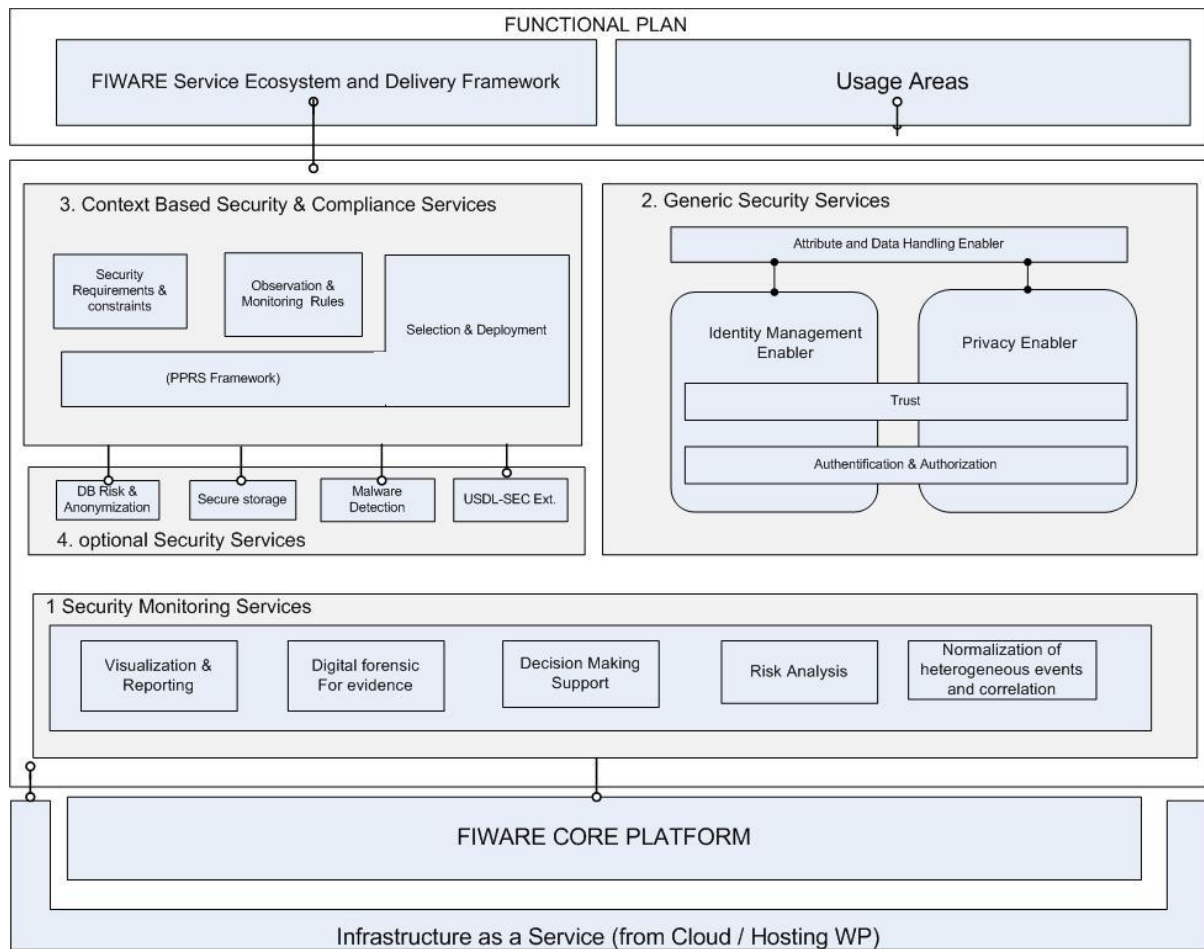
The high level architecture is formed by four main modules: Security monitoring mechanisms (M1), a set of General Core Security Mechanisms (e.g. Identity Management and Privacy solutions) (M2), Context-Based Security and Compliance (M3) where an enhanced version of USDL for security will support the matching of security goals with available security services while addressing compliance management, and a set of universally discoverable Optional Generic Security Services (M4) that will be instantiated at runtime and can be dynamically reconfigured (triggered by M3) based on the needs of specific scenarios.

The overall security plane of the FI-WARE architecture will interlink with practically all its functional modules. In order to simplify the description of these links subsequently the main components as well as their technical relationships with only the Application and Service Ecosystem and Delivery Framework and FI PPP Use Case projects are depicted:

The core general security mechanisms for the FI-WARE project will be provided by M2, including support for Identity Management, Authentication Authorization and Access, and Privacy. M3 will provide the required language and tools for describing services in the FI and their security needs. Where specific scenarios will require optional generic security services these can be consumed on a basis of what is provided by M4. A key architectural assumption is that security services may fail. Security monitoring mechanisms as provided by M1 may detect deviations with respect to the expected behaviour and signal this to M3 to take action (e.g. invoke alternative security services or trigger countermeasures if under attack).

FI-WARE GEs to be developed and/or integrated as part of the Security chapter will materialize the (Security) Reference Architecture sketched in Figure below. This Reference Architecture comprises:

- A component able to dynamically invoke and compose security services to answer related security needs while dealing with constraints which may apply (e.g. regulatory).
- A set of GEs for a number of shared security concerns (i.e. identity and access management as well as privacy and auditing) that are considered core and therefore present in any FI-WARE Instance.
- A set of optional Security GEs to address current and future requests from concrete Usage Areas.
- An advanced security monitoring system that covers the whole spectrum from acquisition of events up to display, going through analysis but also going beyond thanks to a digital forensic tool and assisted decision support in case of cyber attacks.



FI-WARE High Level Security Architecture

More information on the Security Chapter and FI-WARE in general can be found within the following pages:

<http://wiki.fi-ware.eu>

[The Architecture of Security in FI-WARE](#)

[Materializing Security in FI-WARE](#)

1.5 Structure of this Document

The document is generated out of a set of documents provided in the public FI-WARE wiki. For the current version of the documents, please visit the public wiki at <http://wiki.fi-ware.eu/>

The following resources were used to generate this document:

D.8.3.2 Installation and Administration Guide front page

[Security Monitoring - Installation and Administration Guide](#)

[Security Monitoring / Visualisation Framework - Installation and Administration Guide](#)

[Security Monitoring / Service Level SIEM - Installation and Administration Guide](#)

[Security Monitoring / IoT Fuzzer - Installation and Administration Guide](#)

[Security Monitoring / Android Vulnerability Assessment - Installation and Administration Guide](#)

[Security Monitoring / Remediation - Installation and Administration Guide](#)

[Security Monitoring / MulVAL Attack Paths Engine Web Application - Installation and Administration Guide](#)

[Security Monitoring / Scored Attack Paths - Installation and Administration Guide](#)

[Identity Management - Installation and Administration Guide](#)

[Identity Management - One-IDM Installation and Administration Guide](#)

[Identity Management - GCP Installation and Administration Guide](#)

[Data Handling GE - Installation and Administration Guide](#)

[Access Control - Installation and Administration Guide](#)

[DB Anonymizer GE - Installation and Administration Guide](#)

[Malware Detection Service - Installation and Administration Guide](#)

[Context-based security & compliance - Installation and Administration Guide](#)

[Android Flow Monitoring - Installation and Administration Guide](#)

[Content-based Security - Installation and Administration Guide](#)

1.6 Typographical Conventions

Starting with October 2012 the FI-WARE project improved the quality and streamlined the submission process for deliverables, generated out of the public and private FI-WARE wiki. The project is currently working on the migration of as many deliverables as possible towards the new system.

This document is rendered with semi-automatic scripts out of a MediaWiki system operated by the FI-WARE consortium.

1.6.1 Links within this document

The links within this document point towards the wiki where the content was rendered from. You can browse these links in order to find the "current" status of the particular content.

Due to technical reasons part of the links contained in the deliverables generated from wiki pages cannot be rendered to fully working links. This happens for instance when a wiki page references a section within the same wiki page (but there are other cases). In such scenarios we preserve a link for readability purposes but this points to an explanatory page, not the original target page.

In such cases where you find links that do not actually point to the original location, we encourage you to visit the source pages to get all the source information in its original form. Most of the links are however correct and this impacts a small fraction of those in our deliverables.

1.6.2 Figures

Figures are mainly inserted within the wiki as the following one:

```
[[Image:....|size|alignment|Caption]]
```

Only if the wiki-page uses this format, the related caption is applied on the printed document. As currently this format is not used consistently within the wiki, please understand that the rendered pages have different caption layouts and different caption formats in general. Due to technical reasons the caption can't be numbered automatically.

1.6.3 Sample software code

Sample API-calls may be inserted like the following one.

```
http://[SERVER_URL]?filter=name:Simth*&index=20&limit=10
```

1.7 Acknowledgements

The current document has been elaborated using a number of collaborative tools, with the participation of Working Package Leaders and Architects as well as those partners in their teams they have decided to involve.

1.8 Keyword list

FI-WARE, PPP, Architecture Board, Steering Board, Roadmap, Reference Architecture, Generic Enabler, Open Specifications, I2ND, Cloud, IoT, Data/Context Management, Applications/Services

Ecosystem, Delivery Framework , Security, Developers Community and Tools , ICT, es.Internet, Latin American Platforms, Cloud Edge, Cloud Proxy.

1.9 Changes History

Release	Major changes description	Date	Editor
v1	First Version	2013-04-22	TID
v2	First complete Version	2013-05-24	Thales
v3	Final Version	2013-05-29	Thales

1.10 Table of Contents

1.1	Executive Summary	2
1.2	About This Document	3
1.3	Intended Audience	3
1.4	Chapter Context	3
1.5	Structure of this Document	5
1.6	Typographical Conventions	6
1.6.1	Links within this document	7
1.6.2	Figures	7
1.6.3	Sample software code	7
1.7	Acknowledgements	7
1.8	Keyword list	7
1.9	Changes History	8
1.10	Table of Contents	8
2	<i>Security_Monitoring_- _Installation_and_Administration_Guide</i>	16
3	<i>Security Monitoring / Visualisation Framework - Installation and Administration Guide</i>	17
3.1	Introduction	17
3.2	Installation	17
3.2.1	Requirements	17
3.2.2	Build the Visualisation Framework	17
3.2.3	Install the Visualisation Framework	18
3.3	Launching the Visualisation Web Application	18
3.4	Administration	18
3.5	Sanity check procedures	19
3.5.1	End to End testing	19
3.5.2	List of Running Processes	19

3.5.3	Network Interfaces Up & Open	19
3.5.4	Databases	19
3.6	Diagnosis Procedures	20
3.6.1	Resource availability	20
3.6.2	Remote Service Access	20
3.6.3	Resource consumption	20
3.6.4	I/O flows	20
3.7	References	20
4	Security Monitoring / Service Level SIEM - Installation and Administration Guide	22
4.1	Introduction	22
4.1.1	Requirements	22
4.1.1.1	OSSIM	22
4.1.1.2	Java VM	22
4.1.1.3	Database	22
4.1.1.4	Storm Cluster	22
4.1.2	Service Level SIEM Installation steps	23
4.1.2.1	Storm Cluster Installation	23
4.1.2.2	Service Level SIEM Installation	28
4.1.2.3	Supervisor Installation	35
4.2	Administration	36
4.2.1	Storm and Service Level SIEM Topology Administration	36
4.2.2	Supervisor Administration	37
4.3	Sanity check procedures	37
4.3.1	End to End testing	38
4.3.2	List of Running Processes	38
4.3.3	Network interfaces Up & Open	38
4.3.4	Databases	38
4.4	Diagnosis Procedures	40
4.4.1	Resource availability	40
4.4.2	Remote Service Access	40
4.4.3	Resource consumption	40
4.4.4	I/O flows	41
4.5	References	41
5	Security Monitoring / IoT Fuzzer - Installation and Administration Guide	42
5.1	Introduction	42
5.2	Requirements	42
5.2.1	Fuzzing Environment	42
5.2.2	IoT Bridge	43
5.3	Installation	43
5.3.1	Fuzzing Environment	43
5.3.2	IoT Bridge	43
5.4	Sanity check procedures	45
5.4.1	End to End testing	45
5.4.2	List of Running Processes	45
5.4.3	Network interfaces Up & Open	45

5.4.4	Databases	46
5.5	Diagnosis Procedures	46
5.5.1	Resource availability	46
5.5.2	Remote Service Access	46
5.5.3	Resource consumption	46
5.5.4	I/O flows	46
6	Security Monitoring / Android Vulnerability Assessment - Installation and Administration Guide	47
6.1	Introduction	47
6.2	Server Installation	47
6.2.1	Database Installation and Configuration	47
6.2.2	Web Service Installation and Configuration	48
6.3	Android Client Installation	49
6.4	Sanity check procedures	50
6.4.1	End to End testing	50
6.4.2	List of Running Processes	50
6.4.3	Network interfaces Up & Open	51
6.4.4	Databases	51
6.5	Diagnosis Procedures	52
6.5.1	Resource availability	52
6.5.2	Remote Service Access	52
6.5.3	Resource consumption	53
6.5.4	I/O flows	53
7	Security Monitoring / Remediation - Installation and Administration Guide	54
7.1	Introduction	54
7.2	Requirements	54
7.3	Installation	54
7.3.1	Building application	54
7.3.2	Installation instructions	56
7.3.2.1	Configuration files needed by the application	56
7.3.2.2	Installation of the remediation application on Tomcat	57
7.4	Sanity check procedures	57
7.4.1	End to End testing	57
7.4.2	List of Running Processes	58
7.4.3	Network interfaces Up & Open	58
7.4.4	Databases	58
7.5	Diagnosis Procedures	58
7.5.1	Resource availability	58
7.5.2	Remote Service Access	59
7.5.3	Resource consumption	59
7.5.4	I/O flows	59
8	Security Monitoring / MulVAL Attack Paths Engine Web Application - Installation and Administration Guide	60
8.1	Introduction	60

8.2	System Requirements	60
8.3	Installation	60
8.4	Administration	63
8.5	Sanity check procedures	63
8.5.1	End to End testing	64
8.5.2	List of Running Processes	64
8.5.3	Network interfaces Up & Open	64
8.5.4	Databases	64
8.6	Diagnosis Procedures	64
8.6.1	Resource availability	64
8.6.2	Remote Service Access	65
8.6.3	Resource consumption	65
8.6.4	I/O flows	65
8.7	References	65
9	<i>Security Monitoring / Scored Attack Paths - Installation and Administration Guide</i>	66
9.1	Introduction	66
9.2	Requirements	66
9.3	Installation	66
9.3.1	Building application	66
9.3.2	Installation instructions	67
9.3.2.1	Configuration files needed by the application	67
9.3.2.2	Configuration needed by the application	67
9.3.2.3	Installation of the scoring application	67
9.4	Sanity check procedures	67
9.4.1	End to End testing	67
9.4.2	List of Running Processes	68
9.4.3	Network interfaces Up & Open	68
9.4.4	Databases	68
9.5	Diagnosis Procedures	68
9.5.1	Resource availability	69
9.5.2	Remote Service Access	69
9.5.3	Resource consumption	69
9.5.4	I/O flows	69
10	<i>Identity_Management_-_Installation_and_Administration_Guide</i>	70
11	<i>Identity_Management_-_One-IDM_Installation_and_Administration_Guide</i>	71
11.1	Introduction	71
11.2	Installation	71
11.2.1	Software environment	71
11.2.2	IDM provisioning	72
11.3	Administration	72
11.3.1	Network administration	72
11.3.2	Tomcat	72
11.3.3	Apache2	72
11.3.4	IDM	73
11.3.4.1	Circle of trust	73

11.3.4.2	User management	74
11.3.4.3	Administrative tools	75
11.3.4.4	LDAP browser	75
11.4	Sanity Check Procedures	75
11.4.1	End to End testing	76
11.4.2	List of Running Processes	76
11.4.3	Network interfaces Up & Open	76
11.4.4	Databases	76
11.5	Diagnosis Procedures	76
11.5.1	Resource availability	76
11.5.2	Remote Service Access	77
11.5.3	Resource consumption	77
11.5.4	I/O flows	77
12	Identity_Management_-_GCP_Installation_and_Administration_Guide	78
12.1	Introduction	78
12.2	Installation	78
12.2.1	Access to GCP-IDM server	78
12.2.2	Software on demo presentation machine	78
12.2.3	Administrative tools	78
12.2.3.1.1	Administration interfaces	79
12.2.3.1.2	Programming interfaces	79
12.3	Administration	79
12.4	Sanity Check Procedures	84
12.4.1	End to End testing	84
12.4.2	List of Running Processes	84
12.4.3	Network interfaces Up & Open	84
12.4.4	Databases	85
12.5	Diagnosis Procedures	85
12.5.1	Resource availability	85
12.5.2	Remote Service Access	85
12.5.3	Resource consumption	85
12.5.4	I/O flows	85
13	Data Handling GE - Installation and Administration Guide	86
13.1	System Requirements	86
13.1.1	Database	86
13.1.2	Java VM	86
13.1.3	Mozilla Firefox	86
13.1.3.1	PPL PolicyUI: Firefox Plug-in	86
13.2	PPL Installation steps	87
13.3	Sanity Check Procedures	87
13.3.1	End to End testing	87
13.3.2	List of Running Processes	89
13.3.3	Network interfaces Up & Open	89
13.3.4	Databases	89
13.4	Diagnosis Procedures	90

13.4.1	Resource availability	90
13.4.2	Remote Service Access	90
13.4.3	Resource consumption	90
13.4.4	I/O flows	90
14	<i>Access Control - Installation and Administration Guide</i>	91
14.1	Introduction	91
14.2	System Requirements	91
14.3	Installation	91
14.3.1	Operating System Setup	91
14.3.2	Certificate Authority Setup	91
14.3.3	Application Server Setup (Glassfish)	92
14.3.3.1	Server Basic Security	92
14.3.3.2	Server Certificate Setup	92
14.3.3.3	Server TLS Setup	93
14.3.3.4	Server Performance Tuning	94
14.3.4	User Role Management Setup	94
14.3.5	Superadmin Account Setup	94
14.3.5.1	Client Certificate Setup	94
14.3.5.2	User-Role Assignment	95
14.3.6	Authorization Server Application Setup	96
14.3.6.1	Configuration	96
14.3.6.2	Web Application Deployment	96
14.4	Administration	96
14.4.1	Glassfish Webapp Administration	96
14.4.2	Application-Specific Administration	96
14.5	Sanity check procedures	96
14.5.1	End to End testing	97
14.5.2	List of Running Processes	98
14.5.3	Network interfaces Up & Open	98
14.5.4	Databases	98
14.6	Diagnosis Procedures	98
14.6.1	Resource availability	99
14.6.2	Remote Service Access	100
14.6.3	Resource consumption	100
14.6.4	I/O flows	100
15	<i>DB Anonymizer GE - Installation and Administration Guide</i>	101
15.1	Introduction	101
15.2	Installation	101
15.2.1	Database	101
15.2.1.1.1	DB Configuration	101
15.2.2	Application Server	102
15.2.2.1	Application Server configuration	102
15.2.2.2	DB Anonymizer deployment	103
15.2.2.3	Remark: HTTPS/SSL	103
15.3	Sanity Check Procedures	103

15.3.1	End to End testing	103
15.3.2	List of Running Processes	104
15.3.3	Network interfaces Up & Open	104
15.3.4	Databases	104
15.4	Diagnosis Procedures	104
15.4.1	Resource availability	105
15.4.2	Remote Service Access	105
15.4.3	Resource consumption	105
15.4.4	I/O flows	105
16	<i>Malware_Detection_Service_Installation_and_Administration_Guide</i>	106
16.1	Preliminary remark	106
16.2	Resource availability	106
17	<i>Context-based security & compliance - Installation and Administration Guide</i>	107
17.1	Introduction	107
17.2	Installation	107
17.2.1	Requirements	107
17.2.1.1	Operating Systems	107
17.2.1.2	Java VM	107
17.2.1.3	Database	107
17.2.2	PRRS Installation Steps	107
17.3	Administration	111
17.4	Sanity check procedures	112
17.4.1	End to End testing	112
17.4.2	List of Running Processes	112
17.4.3	Network interfaces Up & Open	113
17.4.4	Databases	113
17.5	Diagnosis Procedures	114
17.5.1	Resource availability	114
17.5.2	Remote Service Access	115
17.5.3	Resource consumption	115
17.5.4	I/O flows	115
17.6	References	115
18	<i>Android Flow Monitoring - Installation and Administration Guide</i>	116
18.1	Introduction	116
18.2	Requirements	116
18.2.1	Host Computer	116
18.2.2	Target Device	116
18.2.3	NetFlow Collector	117
18.3	Installation	117
18.4	Administration	117
18.5	Sanity check procedures	117
18.5.1	End to End testing	118
18.5.2	List of Running Processes	118
18.5.3	Network interfaces Up & Open	118
18.5.4	Databases	118

18.6	Diagnosis Procedures	119
18.6.1	Resource availability	119
18.6.2	Remote Service Access	119
18.6.3	Resource consumption	119
18.6.4	I/O flows	119
19	<i>Content-based Security - Installation and Administration Guide</i>	120
19.1	Introduction	120
19.2	Installation	120
19.2.1	Requirements	120
19.2.2	Build the CBS OGE	120
19.2.3	Install the CBS GE	121
19.3	Administration	121
19.3.1	Application Server Configuration	121
19.3.2	Key Generation	121
19.3.3	CBS Provider Configuration	123
19.3.4	CBS Consumer Configuration	124
19.3.5	CBS Broker Configuration	124
19.4	Sanity check procedures	125
19.4.1	End to End testing	125
19.4.2	List of Running Processes	126
19.4.3	Network interfaces Up & Open	126
19.4.4	Databases	126
19.5	Diagnosis Procedures	126
19.5.1	Resource availability	126
19.5.2	Remote Service Access	126
19.5.3	Resource consumption	126
19.5.4	I/O flows	127
19.6	References	127

2 Security_Monitoring_- _Installation_and_Administration_Guide

You can find the content of this chapter as well in the [wiki](#) of fi-ware.

- [Security Monitoring / Visualisation Framework - Installation and Administration Guide](#)
- [Security Monitoring / Service Level SIEM - Installation and Administration Guide](#)
- [Security Monitoring / IoT Fuzzer - Installation and Administration Guide](#)
- [Security Monitoring / Android Vulnerability Assessment - Installation and Administration Guide](#)
- [Security Monitoring / Remediation - Installation and Administration Guide](#)
- [Security Monitoring / MulVAL Attack Paths Engine Web Application - Installation and Administration Guide](#)
- [Security Monitoring / Scored Attack Paths - Installation and Administration Guide](#)

3 Security Monitoring / Visualisation Framework - Installation and Administration Guide

You can find the content of this chapter as well in the [wiki](#) of fi-ware.

3.1 Introduction

The Security Monitoring Visualisation Framework is composed of a Java Web Application, which is packaged in a Web Archive (WAR) file. It requires a Java Application Server. Note that although any application server should work, only Apache Tomcat is supported.

A small utility application (fiware.app.attackgraph) is included that can be used to transfer attack graph data in XML format into the database. This data can then subsequently be visualised in the web application.

3.2 Installation

3.2.1 Requirements

The Visualisation Framework requires the following software to be installed:

- Tomcat 7 [\[1\]](#)
- Maven [\[2\]](#)
- HSQLDB [\[3\]](#)

The Visualisation Framework requires the following Generic Enablers:

- Security Monitoring GE MulVal Attack Paths Engine [\[4\]](#)

3.2.2 Build the Visualisation Framework

The Visualisation Framework can be downloaded from the FI-WARE PPP Restricted Project website [\[5\]](#) as a zip file. The zip file contains two Maven projects, fiware.visualisation and fiware.app.attackgraph, which need to be built using Apache Maven [\[2\]](#), and a local Maven repository containing some of the dependencies needed to build the project.

Once the Maven projects have been unzipped, we need to build them in order to generate the binaries to install. To do so, we run the following commands:

```
$ cd fiware.visualisation
```

```
$ mvn clean package
```

The WAR file is in the target folder of the project.

```
$ cd fiware.app.attackgraph
$ mvn clean package
```

3.2.3 Install the Visualisation Framework

The Visualisation Framework web application is installed by copying the WAR file `com.thalesgroup.uk.trt.fiware.visualisation.WAR` into the "webapps" folder of Apache Tomcat. To install them on other Java Application Servers (e.g. JBoss) please refer to the specific application server guidelines.

3.3 Launching the Visualisation Web Application

In order to run the Visualisation Framework web application the following applications need to be launched:

- HSQLDB Database
 - The database user settings should be left as default:
 - Username: sa
 - Password:
 - The database should be launched with the following parameters:

```
$ cd hsqldb\lib\
$ java -cp hsqldb.jar org.hsqldb.server.Server --database.0
file:fiware.app.attackgraph --dbname.0 fiware.app.attackgraph
```

- Java application server, such as Tomcat

3.4 Administration

The Visualisation Framework web application uses a HSQLDB database to store administrative settings and data to be visualised. This information can be erased via the following steps:

1. Navigate to the web application home page (see [#End to End testing](#))

2. Click the "Data Admin" Button
3. Select which data to remove from the following options:
 1. "Clear the current visualisation layout"
 2. "Delete all data in the Intersection-Visualisation-Model database"

3.5 Sanity check procedures

The Sanity Check Procedures are the steps that a System Administrator will take to verify that an installation is ready to be tested. This is therefore a preliminary set of tests to ensure that obvious or basic malfunctioning is fixed before proceeding to unit tests, integration tests and user validation.

3.5.1 End to End testing

To verify access to the Visualisation Framework, navigate to the home page and check that an HTML page is properly displayed: <http://localhost:<Tomcat HTTP port>/fiware.visualisation/>

3.5.2 List of Running Processes

- Java.exe (Java application server)
- Java.exe (HSQLDB database)

3.5.3 Network Interfaces Up & Open

Port	Protocol	Use
8080	TCP	Tomcat HTTP connector
9001	HSQL	HSQLDB

Note that the Tomcat port can be changed in the Tomcat configuration files.

3.5.4 Databases

When launching the Tomcat application server, the Visualisation web application will automatically attempt to connect to the database. If the connection is unsuccessful, the following error message will be printed to the terminal:

```
Caused by: java.net.ConnectException: Connection refused: connect
```

If this message is shown ensure the database is running and accepting connections using a database data viewer tool such as Squirrel [\[6\]](#).

3.6 Diagnosis Procedures

The Diagnosis Procedures are the first steps that a System Administrator will take to locate the source of an error in a GE. Once the nature of the error is identified with these tests, the system admin will very often have to resort to more concrete and specific testing to pinpoint the exact point of error and a possible solution. Such specific testing is out of the scope of this section.

3.6.1 Resource availability

The resource load of the Visualisation Framework strongly depends on the number of concurrent requests received as well as on the free main memory and disk space. The minimum requirements are:

- Minimum available memory: 256 MB
- Minimum available hard disk space: 256 MB

3.6.2 Remote Service Access

N/A

3.6.3 Resource consumption

Resource consumption strongly depends on the load, especially on the number of concurrent requests. The memory consumption of the Tomcat application server should be between 48MB and 1024MB. These numbers can vary significantly if you use a different application server.

3.6.4 I/O flows

The only expected I/O flow is of type HTTP on the port defined in the Apache Tomcat configuration files, inbound and outbound. Requests interactivity should be low.

3.7 References

1. [↑ http://tomcat.apache.org/tomcat-7.0-doc/setup.html](http://tomcat.apache.org/tomcat-7.0-doc/setup.html)
2. [↑ ^{2.0} ^{2.1} http://maven.apache.org/](http://maven.apache.org/)
3. [↑ http://hsqldb.org](http://hsqldb.org)
4. [↑ http://forge.fi-ware.eu/plugins/mediawiki/wiki/fiware/index.php/MulVal_Attack_Paths_Engine - Installation and Administration Guide](http://forge.fi-ware.eu/plugins/mediawiki/wiki/fiware/index.php/MulVal_Attack_Paths_Engine_-_Installation_and_Administration_Guide)

5. [↑ https://forge.fi-ware.eu/frs/?group_id=23](https://forge.fi-ware.eu/frs/?group_id=23)
6. [↑ http://squirrel-sql.sourceforge.net/](http://squirrel-sql.sourceforge.net/)

4 Security Monitoring / Service Level SIEM - Installation and Administration Guide

You can find the content of this chapter as well in the [wiki](#) of fi-ware.

4.1 Introduction

The Service Level SIEM (SLS) provided for FI-WARE Second Release in the Security Monitoring GE integrates the preconfigured version of the Open Source OSSIM SIEM already released for First Release with a Storm cluster (<http://storm-project.net/>) to provide scalability and distributed real time computation. A set of Java process including Esper libraries (<http://esper.codehaus.org/>) are executed inside that cluster for the processing and correlation at a business perspective of events coming from FI-WARE GE.

4.1.1 Requirements

4.1.1.1 **OSSIM**

OSSIM is distributed as a standalone Debian based Operative System. It must be previously installed and configured in the machine where the Service Level SIEM is going to be running. Detailed instructions can be found in the Installation and Administration Guide provided with the FI-WARE First Release ([https://forge.fi-ware.eu/plugins/mediawiki/wiki/fiware/index.php/Service_Level_SIEM - Installation and Administration Guide](https://forge.fi-ware.eu/plugins/mediawiki/wiki/fiware/index.php/Service_Level_SIEM_-_Installation_and_Administration_Guide)).

4.1.1.2 **Java VM**

The Service Level SIEM requires Java release 1.6 or greater installed.

4.1.1.3 **Database**

The Service Level SIEM requires a MySQL database. It is already installed with the OSSIM distribution.

4.1.1.4 **Storm Cluster**

Storm 0.9.0-wip16 has been used in the Service Level SIEM. It is a free and open source project licensed under the Eclipse Public License (EPL) that can be downloaded from <http://storm-project.net/downloads.html>

NOTE: The stable version v0.8.2 can be used if the Service Level SIEM is going to be used only in Local Cluster. Some bugs have been found during its execution in Remote Cluter mode that are fixed in v0.9.0.

Storm has the following requirements:

- *Apache ZooKeeper Server Package* v3.4.5 (<http://zookeeper.apache.org/>)
- *ZeroMQ* v2.1.7 (<http://www.zeromq.org/>)

If you want to have a supervisory process that manages each of your ZooKeeper/Storm processes, it is required a supervisory process such as *Daemontools* (<http://cr.yip.to/daemontools>).

4.1.2 Service Level SIEM Installation steps

4.1.2.1 **Storm Cluster Installation**

The first step to install the Service Level SIEM is to have a Storm cluster up and running.

1. Definition of roles in the Storm Cluster.

There are two kinds of nodes on a Storm cluster:

- **Master node:** It runs a daemon called *Nimbus* responsible for distributing code around the cluster, assigning task to machines and monitoring for failures.
- **Worker nodes:** Each worker node runs a daemon called *Supervisor* that listens for work assigned to its machine and starts and stops as many worker processes as necessary based on what Nimbus has assigned to it.

So, our first step is to decide which host in the cluster will have installed the master node and which ones the worker nodes. For simplicity, in this Installation Guide we are going to assume we have only one Virtual Machine and consequently, only one worker node installed in the same host that the master node. In a real situation with a cluster of hosts, the installation process shown below must be done in each host in the cluster.

- master node: 192.168.215.118
- worker node: 192.168.215.118

2. Set up a ZooKeeper Cluster.

Storm uses Apache ZooKeeper for coordinating the cluster. For reliable ZooKeeper service, three ZooKeeper servers running on separate machines is the minimum recommended size for what is known as an *ensemble* (a cluster of ZooKeeper servers). And in any case, it is best to use an odd number of machines to handle with failures because the service will be available as long as a majority of the ensemble are up.

To install and configure the cluster, just follow the steps provided in <http://zookeeper.apache.org/> that can be summarized in:

- Set the Java heap size to avoid swapping and consequently have a good performance.

- Download and unzip the ZooKeeper Server Package (v3.4.5) in a folder (e.g. /home/ServiceLevelSIEM/zookeeper-3.4.5)
- Create the configuration file ./zookeeper-3.4.5/conf/zoo.cfg with the name of your ZooKeeper servers. For example:

```
tickTime=2000
dataDir=/var/zookeeper/
clientPort=2181
initLimit=5
syncLimit=2
server.1=zoo1:2888:3888
server.2=zoo2:2888:3888
server.3=zoo3:2888:3888
```

NOTE: In the case of a standalone operation (a single ZooKeeper server for example for developing purposes), the following configuration file should be used:

```
tickTime=2000
dataDir=/var/zookeeper
clientPort=2181
```

- Every machine that is part of the ZooKeeper ensemble should know about every other machine in the ensemble. A *myid* file with a single line containing only the test of that machine's id must be in the directory configured in the dataDir parameter. The id must be unique within the ensemble and should have a value between 1 and 255. For example, the myid of server 1 should contain only the text "1".
- Start the Zookeeper server:

```
# bin/zkServer.sh start
```

- Test your deployment by connecting to the hosts:

```
# bin/zkCli.sh -server 127.0.0.1:2181
```


It is important to remark that a ZooKeeper server will not remove old snapshots and log files, so it is the responsibility of the administrator to set up a cron to deal with this issue.

And since ZooKeeper is fail-fast, it will exit the process if it encounters any error case. Consequently, it is critical to run ZooKeeper under supervision (see how to install and configure a Supervisor below).

3. Install dependencies on Master and Worker machines.

Most of the dependencies such as Java, Python or Unzip are already installed with the OSSIM Debian distribution, but you must download and install the following ones by executing the commands shown below:

* ZeroMQ v2.1.7 (<http://www.zeromq.org/area:download>)

```
# apt-get install autoconf automake libtool
# apt-get install uuid-dev
# apt-get install build-essential
# tar -zxvf zeromq-2.2.0.tar.gz
# cd zeromq-2.2.0
# ./configure
# make
# make install
# ldconfig
```

* JZMQ (<https://github.com/zeromq/jzmq>)

```
# unzip jzmq-master.zip
# cd jzmq-master
# ./autogen.sh
# ./configure
# make
# make install
# ldconfig
```

4. Download and extract a Storm release

Download and extract the Storm software (v0.9.0) into Master and Worker machines and add the `storm-0.9.0-wip16/bin` directory to your `PATH` variable in the `~/.bashrc` file.

5. Configure the Storm Cluster

The file at `./storm-0.9.0-wip16/conf/storm.yaml` is used to configure the Storm daemons. It must include the following parameters in each host in the cluster:

- *storm.zookeeper.servers*: This is a list of the hosts in the Zookeeper cluster for your Storm cluster. It should look something like:

```
storm.zookeeper.servers:  
- "192.168.215.118"  
- "555.666.777.888"
```

NOTE: If the port that your Zookeeper cluster uses is different than the default, you should set the parameter *storm.zookeeper.port* as well.

- *storm.local.dir*: The Nimbus and Supervisor daemons require a directory on the local disk to store small amounts of state (like jars, configs, and things like that). You should create that directory on each machine, give it proper permissions, and then fill in the directory location using this config. For example:

```
storm.local.dir: "/mnt/storm"
```

- *java.library.path*: This is the load path for the native libraries that Storm uses (ZeroMQ and JZMQ). The default of `"/usr/local/lib:/opt/local/lib:/usr/lib"` should be fine for most installations, so you probably do not need to set this config.
- *nimbus.host*: The worker nodes need to know which machine is the master in order to download topology jars and configs. For example:

```
nimbus.host: "192.168.215.118"
```

- *supervisor.slots.ports*: For each worker machine, you configure how many workers run on that machine with this config. Each worker uses a single port for receiving messages, and this setting defines which ports are open for use. For example, if you define five ports here, then Storm will allocate up to five workers to run on this machine. By default, this setting is configured to run 4 workers on the ports 6700, 6701, 6702, and 6703. For example:

```
supervisor.slots.ports:
    - 6700
    - 6701
    - 6702
    - 6703
```

7. Configure remote mode of operation

Storm has two modes of operation:

- *local mode*
- *remote mode*

In local mode, you can develop and test topologies completely in process on your local machine. In remote mode, you submit topologies for execution on a cluster of machines. To be able to start and stop topologies on a remote cluster, it is necessary to put the host address of the Master node in the file `~/storm/storm.yaml`

```
nimbus.host: "192.168.215.118"
```

8. Run all the Storm daemons

NOTE: It is critical that you run each of these daemons under supervision. Storm is a fail-fast system which means the processes will halt whenever an unexpected error is encountered. Storm is designed so that it can safely halt at any point and recover correctly when the process is restarted. This is why Storm keeps no state in-process but if Nimbus or the Supervisors restart, the running topologies are unaffected.

- *Nimbus*: Run the command "storm nimbus &" under supervision on the master machine.
- *Supervisor*: Run the command "storm supervisor &" under supervision on each worker machine. The supervisor daemon is responsible for starting and stopping worker processes on that machine.
- *UI*: Run the Storm UI (a site you can access from the browser that gives diagnostics on the cluster and topologies) by running the command "storm ui &" under supervision. The UI can be accessed by navigating your web browser to <http://{nimbus host}:8080>.

The daemons will log to the logs/ directory in wherever {Storm_install} directory where you extracted the Storm release.

NOTE: More detail information about setting up a Storm cluster can be found in <https://github.com/nathanmarz/storm/wiki/Setting-up-a-Storm-cluster>

4.1.2.2 **Service Level SIEM Installation**

In order to add the Service Level SIEM functionalities to the OSSIM core engine released in FI-WARE First Release it is necessary:

1. Install the Service Level SIEM files in the Storm cluster:

All the required files used in the Storm cluster for the Service Level SIEM are included in the **ServiceLevelSIEM-2.2.3.tar.gz** file. It must be unzipped under the folder with the Storm installation (we will refer in this guide to this directory with *{Storm_install}*).

2. Add patch to the OSSIM Agent installed by default with the OSSIM Debian distribution:

Once you have unzipped the Service Level SIEM files, go to the created *ossim_patch* folder and execute the script **install_SLS_patch.sh**. This script will make a backup of the */usr/share/ossim-agent/ossim_agent* and */etc/ossim* directories provided with the OSSIM default distribution and will copy the new ones. Once executed, the folder *ossim_patch* can be removed.

```
# tar -zxvf ossim_SLS_patch.tar.gz
# cd ossim_patch
# ./install_SLS_patch.sh
# cd ..
# rm -rf ossim_patch
```

Make sure the following configuration files include the new plugin for detecting events coming from FI-WARE GE:

*** */etc/ossim/ossim_setup.conf***

It must contain "fiware" in the parameter *detectors* in the section for *[sensor]* and in the parameter *server_plugins* included in the section *[server]*

```
...
[sensor]
detectors=snare, syslog, fiware
...
```

```
[server]

...

server_plugins=osiris, pam_unix, ssh, snare, sudo, fiware

...
```

*** /etc/ossim/agent**

You must include the following section and configure the ip where your Nimbus is running in the Storm cluster. The default port 41000/tcp can be modified but in that case it should be modified also in the Storm configuration.

```
[output-storm]

enable=True

ip=192.168.215.118

port=41000

send_events=True
```

3. Configure the Service Level SIEM files in the Storm cluster:

All the configuration files are located under the {Storm_install}/conf directory. The main configuration file is the **ServiceLevelSIEM.conf**. This file includes different sections to configure: the listening port for the events coming from the ossim-agent, the files with the schema of events and alarms, the properties to be used in the correlation engine, the information to be used in the Storm topology for the different parallel processing and the data to access OSSIM Database in order to store the generated Service Level Alarms.

```
//AGENT OSSIM PROPERTIES

spout_port = 41000


//EVENT SCHEMA FILE

eventSchema = /conf/schemas/eventSchema


//ALARM SCHEMA FILE

alarmSchema = /conf/schemas/alarmSchema
```

```
//CORRELATION PARAMETERS

EPLVariables = /conf/EPLVariables.conf
EPLStatements = /conf/EPLStatements.conf

correlation_rule = select AlarmID, AlarmMSG, a.event_id as
FirstEvent, c.event_id as LastEvent

    from pattern [every (a=PRRSEventA) ->
b=PRRSEventB(client_ip=a.client_ip and sec_enabler=a.sec_enabler)

    -> c=PRRSEventC(client_ip=a.client_ip and
securitySpecs=a.securitySpecs)

        where timer:within(var_timeout sec)]

alarmMSGToEmit = AlarmMSG
alarmIDToEmit = AlarmID
eventToEmit1 = FirstEvent
eventToEmit2 = LastEvent

// TOPOLOGY PARAMETERS

topologyName = "topologyPATTERN"

TOPOLOGY_WORKERS = 5

spoutParallelism = 1

ge_filterParallelism = 1

filterParallelism = 1

geParallelism = 1

activateCorrelation = 1

corrParallelism = 1

activateDBWriter = 1

dbWriterParallelism = 1
```

```
//OSSIM DATABASE

databaseIP = localhost

databasePort = 3306

databaseAlarms = alienvault

databaseEvents = alienvault_siem

userName = fiware

pwd = flwar3
```

The Jar file with the topology to be started in the Storm cluster for the Service Level SIEM is called *ServiceLevelSIEM-0.0.1-SNAPSHOT-jar-with-dependencies.jar* and it is stored in a folder called *topologies*.

4. Database configuration

A new user for FI-WARE must be created in the OSSIM database with permissions for the *alienvault* and *alienvault_siem* tables. Once created, the data for this new user must be configured in the *ServiceLevelSIEM.conf* file.

```
mysql> create user 'fiware'@'localhost' identified by 'flwar3';

mysql> grant all privileges on *.* to 'fiware'@'localhost';
```

Finally, in order to perform the required modifications in the OSSIM Database, the script **db/slSIEM.sql** (included in the *ServiceLevelSIEM.tar.gz* file) must be executed.

```
mysql> source slSIEM.sql
```

With that script, a new table called **sls_alarm** is created to store the new Service Level SIEM Alarms. This type of alarms are not the common and known network risks such as port scanning or brute force attacks (that will be detected with the OSSIM core engine already release in FI-WARE First Release and stored in the *alarm* table) but defined from a business perspective depending the client. For this reason, a new table called **sls_rule** is created to store the ids assigned to them and a short description. For example, for the FI-WARE Context-based Security & Compliance GE we have defined the following rule:

```
70001: "FI-WARE additional security level compromised in client
environment"
```

The events that will generate these alarms will arrive to the Service Level SIEM through the Syslog Server. Due to this fact, all of them will have plugin = 4007 (the one for syslog events), but with a new plugin_sid. For our example with the FI-WARE Context-based Security & Compliance GE we have defined the following one in the PLUGIN_SID table:

```
2: "syslog: FI-WARE Context-based Security & Compliance event"
```

Under the folder conf/db there are other two files called 'sqlsAlarmsDB.conf' and 'sqlsEventsDB.conf' (also included in the ServiceLevelSIEM.tar.gz file) where it is defined the SQL commands used in the Service Level SIEM to store the new alarms and recover the information required.

The fields included in the new table *sls_alarm* must be defined in the *conf/schemas/alarmSchema*. Currently, that table includes the following ones but they could be adapted depending the business requirements and the information that the user wants to be shown in the Visualization Framework:

```
sls_alarm_id, rule_id, msg, timestamp, firstEvent_id, lastEvent_id
```

5. Correlation configuration

The Service Level SIEM includes an Esper correlation engine to process rules written in 'Event Processing Language (EPL)' language. The following files located under the folder *{Storm_install}/conf* are used to configure it:

*** EPLVariables.conf** This file is used to define variables to be used in the EPL statements included in the correlation. They depends on the correlation to be performed to arise alarms from a service level or business perspective. In our example, where the correlation rule is a pattern of events, it includes the windows timeout for the reception of the events once the first of them has been detected and before discarding it. It also includes the alarm ID and description that will be sent when then pattern is detected.

```
var_timeout, integer, 10

AlarmMSG, string, FI-WARE additional security level compromised in
client environment

AlarmID, string, 70001
```

*** EPLStatements.conf** This file is used to define statements in EPL language to be used in the Esper process included in the topology for the correlation. In our example, they include the definition of the events included in the correlation pattern for the case of events coming from the implementation called *PRRS* of the FI-WARE Context-based security & compliance GE.

```
insert into PRRSEventA select * from filterBolt_default where
(FIWARE_GE = "prrs") and (status = "non-compliance")
```



```
insert into PRRSEventB select * from filterBolt_default where
(FIWARE_GE = "prrs") and (status = "undeployed")

insert into PRRSEventC select * from filterBolt_default where
(FIWARE_GE = "prrs") and (status = "not-found")

insert into PRRSEventD select * from filterBolt_default where
(FIWARE_GE = "prrs") and (status = "deployed")
```

* Correlation rule (ServiceLevelSIEM.conf)

The correlation rule, included in the ServiceLevelSIEM.conf file, is the EPL statement that will have added a listener in the correlator. This means that when the condition included in it happens, a tuple will be emitted by this process included in the Storm topology.

In our example, this correlation rule is a pattern of events but it could have any other rule depending the client and the service level alarm that wants to be generated. It is important to remark that this correlation rule is the key to have alarms from a business perspective. It will use the events and variables defined in the EPLStatements.conf and EPLVariables.conf respectively.

```
correlation_rule = select AlarmID, AlarmMSG, a.event_id as
FirstEvent, c.event_id as LastEvent from pattern

    [every (a=PRRSEventA)

        -> b=PRRSEventB(client_ip=a.client_ip and
sec_enabler=a.sec_enabler)

        -> c=PRRSEventC(client_ip=a.client_ip and
securitySpecs=a.securitySpecs)

        where timer:within(var_timeout sec)]
```

6. Running the Service Level SIEM Topology in the Storm Cluster

To run the topology that includes the processes for the Service Level SIEM in the Storm Cluster, you only need to execute the script **startSLSTopology.sh** included directly in the {Storm_install} folder.

```
# ./startSLSTopology.sh
```

It is important to remark that although the Storm processes can be started with any user, the Service Level SIEM topology must be executed with the root user. Otherwise, the connection with the OSSIM core engine (running with root user) does not work correctly.

The log files for the Storm processes are located under the folder {Storm_install}/log. Besides the Nimbus.log and Supervisor.log files, a file is created by each worker process with the number of the port assigned to it. For example:

```
~/storm-0.9.0-wipl6/logs$ ls -l
total 1116
-rw-r--r-- 1 storm storm 236268 Apr 22 14:16 nimbus.log
-rw-r--r-- 1 storm storm 572728 Apr 22 13:27 supervisor.log
-rw-r--r-- 1 storm storm 60695 Apr 22 14:24 worker-6701.log
-rw-r--r-- 1 storm storm 59208 Apr 22 14:24 worker-6702.log
-rw-r--r-- 1 storm storm 53764 Apr 22 14:24 worker-6703.log
-rw-r--r-- 1 storm storm 56107 Apr 22 14:24 worker-6704.log
-rw-r--r-- 1 storm storm 64210 Apr 22 14:24 worker-6705.log
~/storm-0.9.0-wipl6/logs$
```

Once those files are created and the connections between them and the supervisor process take place, you can see the port 41000/tcp is listening with the following command:

```
~/storm-0.9.0-wipl6/logs$ netstat -an |grep 41000
tcp        0      0 0.0.0.0:41000          0.0.0.0:*
LISTEN
tcp        0      0 192.168.215.118:48566 192.168.215.118:41000
ESTABLISHED
tcp        0      0 192.168.215.118:41000 192.168.215.118:48566
ESTABLISHED
~/storm-0.9.0-wipl6/logs$
```

NOTE: To run the Service Level SIEM in local mode (without Nimbus and Supervisor daemons), the following command must be executed:

```
# storm jar topologies/ServiceLevelSIEM-0.0.1-SNAPSHOT-jar-with-
dependencies.jar storm.sls.main.SLSTopologyPattern
```

4.1.2.3 **Supervisor Installation**

This step is only necessary in case you want to have a supervisory process to manage the ZooKeeper and Storm clusters. One of the most known supervisors is the 'daemontools' (<http://cr.yp.to/daemontools.html>), a collection of tools for managing UNIX services.

Once the **daemontools-0.76.tar.gz** has been downloaded and unzipped, it is required to apply the patch from <http://blog.tonycode.com/tech-stuff/setting-up-djbdns-on-linux>, adding “-include /usr/include/errno.h” to src/conf-cc and executing

```
# cd daemontools-0.76/package
# ./install
```

Add the following line at the end of the */etc/inittab* file to start the **svscan** process in the */service* directory:

```
SV:123456:respawn:/command/svscanboot
```

The process *svscan* starts one supervise process for each subdirectory in that */service* directory. Consequently, to supervise the Zookeeper it is necessary to create a directory called *zookeeper* and include inside the following run script (substituting *myuser* by your username and the installation directory):

```
#!/bin/sh

exec su -l myuser -c "/home/myuser/SLS/zookeeper-3.4.5 &&
./bin/zkServer.sh start-foreground"
```

And the same for the storm processes *Nimbus* and *Supervisor* (depending the host in the cluster):

```
#!/bin/sh

exec su -l myuser -c "cd /home/myuser/SLS/storm-0.9.0-wip16 &&
./bin/storm nimbus"

#!/bin/sh

exec su -l myuser -c "cd /home/myuser/SLS/storm-0.9.0-wip16 &&
./bin/storm supervisor"
```

4.2 Administration

4.2.1 Storm and Service Level SIEM Topology Administration

We can differentiate two situations:

'1.' We want to test a topology in a **local cluster** (in this case, it is not necessary to have a cluster of hosts with the Nimbus and Supervisors daemons running). In this case we have to execute the storm command without the final argument with the name we want to register the topology:

```
#storm jar ServiceLevelSIEM-0.0.1-SNAPSHOT-jar-with-dependencies.jar
storm.sls.main.SLSTopologyPattern | tee -ai output.log
```

The last part of the command will allow to write all the logs shown in the console in a file for later analysis.

'2.' We have a **remote cluster of hosts** with the Nimbus and Supervisors correctly configured and running. In this case, we need to provide a name to register the topology in the cluster.

```
#storm jar ServiceLevelSIEM-0.0.1-SNAPSHOT-jar-with-dependencies.jar
storm.sls.main.SLSTopologyPattern ServiceLevelSIEM
```

In this situation, we can list all the active topologies in the cluster with the following command:

```
# storm list

0      [main] INFO  backtype.storm.thrift  - Connecting to Nimbus at
192.168.215.118:6627

Topology_name      Status      Num_tasks  Num_workers  Uptime_secs
-----
ServiceLevelSIEM   ACTIVE      6           3             68
```

Finally, the following command must be executed to kill an active topology in the cluster:

```
# storm kill {TopologyName}
```

NOTE: Storm will not kill the topology immediately. Instead, it deactivates all the Spouts so that they do not emit any more tuples and then Storm waits *Config.TOPOLOGY_MESSAGE_TIMEOUT_SECS* seconds before destroying all the workers. This gives the topology enough time to complete any tuples it was processing when it got killed.

To work in an easiest way, the following admin scripts are provided with the Service Level SIEM under the folder `./{Storm_install}` to do it automatically:

- `startSLSTopology.sh`
- `stopSLSTopology.sh`
- `restartSLSTopology.sh`

All available options to be used with Storm are shown with the command: **storm help**

4.2.2 Supervisor Administration

- The command **svstat** is used to see the status of a supervised service. For example:

```
# cd /service
# svstat storm-supervisor
```

- The command **svc** is used to stop a supervised service. For example:

```
#cd /service
#svc -p storm-supervisor
```

Other options to be executed with the **svc** command are:

- **-u:** Up. If the service is not running, start it. If the service stops, restart it.
- **-d:** Down. If the service is running, send it a TERM signal and then a CONT signal. After it stops, do not restart it.
- **-o:** Once. If the service is not running, start it. Do not restart it if it stops.
- **-k:** Kill. Send the service a KILL signal.

4.3 Sanity check procedures

The Sanity Check Procedures are the steps that a System Administrator will take to verify that an installation is ready to be tested. This is therefore a preliminary set of tests to ensure that obvious or basic malfunctioning is fixed before proceeding to unit tests, integration tests and user validation.

4.3.1 End to End testing

To test quickly that everything is up and running, it is only necessary to connect to the master node and execute the command *storm list* to see the active topologies. It must show the topology *slsTopologyPattern* in ACTIVE state.

```
# storm list

0      [main] INFO  backtype.storm.thrift  - Connecting to Nimbus at
192.168.215.118:6627

Topology_name      Status      Num_tasks  Num_workers  Uptime_secs
-----
ServiceLevelSIEM   ACTIVE      6           3             68
```

4.3.2 List of Running Processes

The following processes should be up and running:

- *ossim-server*
- *ossim-agent*
- *zookeeper*
- *storm nimbus* (in the master node)
- *storm supervisor* (in the worker nodes)

NOTE: The processes *zookeeper*, *storm nimbus* and *storm supervisor* can be running also within a *supervise* process.

4.3.3 Network interfaces Up & Open

The following ports must be listening for connections in the master node:

- port 40001/tcp - *ossim-server*
- port 41000/tcp (or the one configured in the *ServiceLevelSIEM.conf*) - spout to pass events to the Storm cluster
- port 514/udp - to receive events from the FI-WARE Generic Enablers through the Syslog Server

4.3.4 Databases

The following OSSIM databases must be up and accept queries:

- alienvault
- alienvault_siem

The connection details are included in the `conf/ServiceLevelSIEM.conf` file.

Test query:

```
# mysql -u fiware -p
# mysql> use alienvault;
# mysql> select count(*) from sls_rule;
+-----+
| count(*) |
+-----+
|          1 |
+-----+
1 row in set (0.05 sec)

mysql> use alienvault_siem;

mysql> select count(*) from device;
+-----+
| count(*) |
+-----+
|          1 |
+-----+
1 row in set (0.13 sec)
```

4.4 Diagnosis Procedures

The Diagnosis Procedures are the first steps that a System Administrator will take to locate the source of an error in a GE. Once the nature of the error is identified with these tests, the system admin will very often have to resort to more concrete and specific testing to pinpoint the exact point of error and a possible solution. Such specific testing is out of the scope of this section.

4.4.1 Resource availability

For the ZooKeeper Server is necessary to set the Java heap size. This is very important to avoid swapping, which will seriously degrade ZooKeeper performance. To determine the correct value, use load tests, and make sure you are well below the usage limit that would cause you to swap. Be conservative and use a maximum heap size of 3GB for a 4GB machine.

Then the minimum requirements:

3GB of RAM

And comfortable use is:

8 GB of RAM

4.4.2 Remote Service Access

The system must be connected to the network in order to detect security events.

There are several ways to remotely access the system and its components:

- Web based interface
- SSH login
- SFTP server
- MySQL database
- Syslog server

4.4.3 Resource consumption

The CPU consumption is rather complex to determinate, because it not only depends on the number of events sent to the Service Level SIEM, but also on the complexity of the active correlation rules.

It is recommended to minimize the arithmetic operations applied in the correlation rules because these are the most resource consuming procedures and they are meant to be performed millions of times per second. Also, if there is a point with too much network activity and it constantly overloads

a host, it is recommended to split the network and put one more host in the Storm Cluster and an OSSIM-Agent instance on each subnetwork in order to balance the load.

Generally, it consumes 3GB upto 8 GB of RAM. Two CPUs are required for the use of this GE.

4.4.4 I/O flows

I/O flow on HTTPS & SSH standard ports and inputs to 514/udp port (Syslog).

The OSSIM Agent will communicate with the port 41000/tcp in the Master node. That flow must be ensured in case the OSSIM Agent are installed in different machines.

4.5 References

- OSSIM SIEM: <http://www.ossim.net>
- Storm Tutorial: <http://storm-project.net/documentation.html>
- Esper Tutorial: <http://esper.codehaus.org/tutorials/tutorials.html>
- ZooKeeper Tutorial: <http://zookeeper.apache.org/doc/r3.4.5/>
- Daemons Tools: <http://cr.yp.to/daemontools.html>

5 Security Monitoring / IoT Fuzzer - Installation and Administration Guide

You can find the content of this chapter as well in the [wiki](#) of fi-ware.

5.1 Introduction

6LowFuzzer, the reference implementation of the IoT Fuzzer is an application written in python, supported by the [scapy](#) packet manipulation library, and communicating with IoT devices through an [Atmel RZUSBstick](#). This document will go through the necessary steps in order to install 6LowFuzzer.

5.2 Requirements

To be able to install and use 6LowFuzzer, there are a few requirements, for the fuzzing tool itself, as well as for the installation of a firmware on the RZUSBstick.

5.2.1 Fuzzing Environment

The supported environment is based on Ubuntu 12.04 LTS, for x86-64 architecture.

A few dependencies have to be present on the system. Most of them can be obtained as packages:

```
user@host:~$ sudo aptitude install python mercurial python-pypcap  
python-tk
```

Scapy does not include our contributed 6LoWPAN module in a stable version yet. So, scapy has to be installed by fetching its sources from their repository:

```
user@host:~$ hg clone http://hg.secdev.org/scapy-com scapy  
destination directory: scapy  
requesting all changes  
adding changesets  
adding manifests  
adding file changes  
added 1564 changesets with 2317 changes to 187 files (+2 heads)  
updating to branch default
```

```
169 files updated, 0 files merged, 0 files removed, 0 files
unresolved
```

Then, it can be installed using the following commands:

```
user@host:~$ cd scapy
user@host:~/scapy$ sudo python setup.py install
```

5.2.2 IoT Bridge

To be able to flash 6LowFuzzer's firmware onto the RZUSBstick, one will require an Atmel AVR development environment, composed of:

- one On-chip programming platform, like the Atmel JTAGICE mkII;
- one PC running Windows, with:
 - AVR Studio 4.14 or later;
 - Windows drivers installed for the JTAGICE mkII.

5.3 Installation

5.3.1 Fuzzing Environment

Once all dependencies have been installed, the only missing step to be able to run the fuzzer is to extract it:

```
user@host:~$ tar zxvf 6lowfuzzer-2.2.0.tgz
```

5.3.2 IoT Bridge

In the 6lowfuzzer-2.2.0.tgz, there is a "firmware/ravenusbstick.elf" file, that contains the firmware that has to be flashed onto the RZUSBstick.

Detailed instructions can be found in the Contiki Documentation (section Modules/Tutorials/Running Contiki with uIPv6 and SICSLOWPAN support on Atmel RAVEN hardware), which can be read online [here](#).

Here are the required steps to perform the installation:

- launch AVR Studio and exit any popup window;

- connect the JTAG pins of the JTAGICE into the JTAG connector of the RZUSBstick;
- in AVR Studio, in the "Tools" menu, select "Program AVR/Auto Connect";
- go to the "Main" tab;
- in the "Programming Mode and Target Tettings" list, select "JTAG";
- select "AT90USB1287" in the "Device and Signature Bytes" list, then click "Read Signature". If the Device signature is read properly, it means AVR Studio is properly connected to the RZUSBstick;
- go to the "Program" tab;
- in the "ELF Production file format" section, browse to the firmware binary, then click the "Program" button.

LEDs on the JTAGICE should start to flicker, and text should appear in the text field at the window, similar to:

```
Validating ELF input file.. OK!
Reading FLASH input..OK!
Reading EEPROM input..OK!
Reading FUSE input..OK!
Reading LOCKBIT input..OK!
Reading SIGNATURE input..OK!
Entering programming mode.. OK!
Erasing device.. OK!
Programming FLASH ..      OK!
Reading FLASH ..          OK!
Leaving programming mode.. OK!
```

The RZUSBstick is now ready to be used with 6LowFuzzer, and it can be disconnected.

5.4 Sanity check procedures

The Sanity Check Procedures are the steps that a System Administrator will take to verify that an installation is ready to be tested. This is therefore a preliminary set of tests to ensure that obvious or basic malfunctioning is fixed before proceeding to unit tests, integration tests and user validation.

5.4.1 End to End testing

To test that 6LowFuzzer and all its dependencies have been properly installed, one can simply run it:

```
user@host:~$ cd 6lowfuzzer-2.2.0
user@host:~/6lowfuzzer-2.2.0$ python executor.py -h
Usage: executor.py [options]

Options:
  -h, --help            show this help message and exit
  -s SCENARIO_FILE, --scenario=SCENARIO_FILE
                        name of the file Scenario to play with
  [...]
```

If no exception show up instead of the above help message, it means that everything is fine.

Similarly, to test that all dependencies for the message building interface are properly installed, one can simply run it:

```
madynes@vbox:~/6lowfuzzer-2.2.0$ python building.py
```

If the graphical interface shows up, it means that all is fine.

5.4.2 List of Running Processes

While the fuzzer is executing, there should be only one process running "executor.py".

5.4.3 Network interfaces Up & Open

When the RZUSBstick is plugged in, it should appear as a network interface, usually "usb0".

5.4.4 Databases

N/A

5.5 Diagnosis Procedures

The Diagnosis Procedures are the first steps that a System Administrator will take to locate the source of an error in a GE. Once the nature of the error is identified with these tests, the system admin will very often have to resort to more concrete and specific testing to pinpoint the exact point of error and a possible solution. Such specific testing is out of the scope of this section.

5.5.1 Resource availability

6LowFuzzer does not depend on heavy RAM or storage usage, and should run fine in normal system conditions.

5.5.2 Remote Service Access

6LowFuzzer works by sending malformed 6LoWPAN packets to a target, in order to assert its robustness. So, the target IoT device has to be in range of the fuzzing environment, to be able to communicate with it.

5.5.3 Resource consumption

CPU and RAM usage of 6LowFuzzer should remain low at all times. An increased level of CPU or RAM consumption might indicate that the fuzzed device started to behave erratically, and it flooding the 6LoWPAN network.

5.5.4 I/O flows

There are no input / output flows.

6 Security Monitoring / Android Vulnerability Assessment - Installation and Administration Guide

You can find the content of this chapter as well in the [wiki](#) of fi-ware.

6.1 Introduction

Ovaldroid, the reference implementation of the Android Vulnerability Assessment tool, is an Android application written in Java, connected to a remote database through a web service, that stores OVAL definitions and results. This document will go through the necessary steps in order to install and configure Ovaldroid.

6.2 Server Installation

Before starting the installation, the archive has to be extracted, using the following command lines:

```
user@host:~$ tar zxvf Ovaldroid-2.2.0.tgz
user@host:~$ cd Ovaldroid-2.2.0
```

6.2.1 Database Installation and Configuration

The database (named ANDROVAL) is used to store both OVAL definitions and results (those ones are sent by Android device) in two different tables (namely OVALDEF and OVALRES).

To install and configure the database, just follow these steps:

- install the required mysql packages:

```
user@host:~/Ovaldroid-2.2.0$ sudo aptitude install mysql-server
mysql-client
```

- create the database:

```
user@host:~/Ovaldroid-2.2.0$ sudo mysql -p <
scripts/install_db.sql
```

With **install_db.sql** containing:

```
CREATE DATABASE ANDROVAL;

GRANT ALL PRIVILEGES ON ANDROVAL.* TO 'androval'@'localhost'
IDENTIFIED BY 'androval';
```

```
USE ANDROVAL

create table OVALDEF
(
    id_def                INTEGER        not null,
    id_cve                 VARCHAR(56) ,
    upload_date            TIMESTAMP     not null,
    path                   VARCHAR(128)   not null,
    tags                   VARCHAR(256) ,
    primary key (id_def)
);

create table OVALRES
(
    imei                   VARCHAR(56)    not null,
    tstamp                 TIMESTAMP ,
    path                    VARCHAR(256)   not null,
    primary key (imei, path)
);
```

Please note that the "androval" user – used by the web services to interact with the db – uses a password (default is "androval") to connect to the database.

Note: on Ubuntu 12.04 systems, the mysql daemon is automatically started after boot by default, hence there is no need to type special command-line to start it.

6.2.2 Web Service Installation and Configuration

The provider WS is used to store and redistribute OVAL definitions to the Android devices, while the reporter WS allows to store the assessments results of the devices. Both of them are REST-based web services.

First, Apache Tomcat needs to be installed:


```
user@host:~/Ovaldroid-2.2.0$ sudo aptitude install tomcat7
```

After the installation, web services can be deployed into Apache-Tomcat webserver using their WAR files:

```
user@host:~/Ovaldroid-2.2.0$ sudo cp -a webservices_war/*.war  
/var/lib/tomcat7/webapps/
```

Finally, the last task to perform on the server is to create the directories that will be used by the web services, and give them the correct rights. In a shell, type:

```
user@host:~/Ovaldroid-2.2.0$ sudo mkdir -p /var/androval/oval_def  
/var/androval/oval_res  
  
user@host:~/Ovaldroid-2.2.0$ sudo chown -R tomcat7:tomcat7  
/var/androval
```

The first directory is used to store the OVAL definitions on the server file system, while the second one is used to store the received results. It is mandatory to create those directories, especially the oval_res (otherwise no result will be stored on the server file system).

6.3 Android Client Installation

Installation of the client application is performed using a computer connected to the client device.

- The [Android SDK Tools](#) must be installed on the host, and its Android SDK Platform-tools package must be installed as well, in order to use the Android Debug Bridge tool (ADB – part of the Platform-tools package). Although this isn't absolutely required, as there are alternative ways to go through the installation, it is the tool that will be used in this document.
- The target device must be connected to the host, in a way that it is accessible through the Android Debug Bridge tool. This is usually done using the USB cable bundled with the device by its vendor.

Once the computer is set up, the Android application can be installed on the client device:

- On the device, go to Menu -> Settings -> Applications and check the box marked “Unknown Sources”;
- go to Menu -> Settings -> Applications -> Development and check the box marked “USB debugging”;
- to install the application on the device, run

```
user@host:~/Ovaldroid-2.2.0$ adb install
andro_client/apk/fr.inria.madynes.androval.apk
```

Finally, a last – and mandatory – step is needed to finish the installation: copy the 'xovaldi' directory from your computer to the Android device SD card:

```
user@host:~/Ovaldroid-2.2.0$ adb push andro_client/xovaldi/
/sdcard/xovaldi
```

6.4 Sanity check procedures

The Sanity Check Procedures are the steps that a System Administrator will take to verify that an installation is ready to be tested. This is therefore a preliminary set of tests to ensure that obvious or basic malfunctioning is fixed before proceeding to unit tests, integration tests and user validation.

6.4.1 End to End testing

After the WS WAR files have been deployed into Apache-Tomcat, to check if the WS have adequately started, you can open a web browser on the computer and go to the following URL : <http://localhost:8080/fr.inria.madynes.androval.providerWS/hello> ('Hello Updater' provider test) and <http://localhost:8080/fr.inria.madynes.androval.reporterWS/hello> ('Hello Reporter' reporter test).

Regarding the client, to check if the application is correctly installed, just run it (Vuln. Manager app) on the device and click on the UI-button “Start services”. A small pop-up window should warn you that services have started.

6.4.2 List of Running Processes

Once everything has been installed and started, a tomcat and a mysql processes should be running:

```
user@host:~$ ps axu | grep -e '\(tomcat\|mysql\)'

mysql          985   0.0  2.1 492512 43116 ?        Ssl  15:33   0:01
/usr/sbin/mysqld

tomcat7       1198   0.9  7.0 1507004 144220 ?        Sl   15:33   0:12
/usr/lib/jvm/default-java/bin/java

-
Djava.util.logging.config.file=/var/lib/tomcat7/conf/logging.properties
-Djava.awt.headless=true

-Xmx128m -XX:+UseConcMarkSweepGC -
Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager
```

```
-Djava.endorsed.dirs=/usr/share/tomcat7/endorsed

-classpath
/usr/share/tomcat7/bin/bootstrap.jar:/usr/share/tomcat7/bin/tomcat-
juli.jar

-Dcatalina.base=/var/lib/tomcat7 -
Dcatalina.home=/usr/share/tomcat7

-Djava.io.tmpdir=/tmp/tomcat7-tomcat7-tmp
org.apache.catalina.startup.Bootstrap start
```

6.4.3 Network interfaces Up & Open

In order for the web service to be reachable, the tcp port 8080, used by tomcat, should be accessible from the Android devices.

6.4.4 Databases

To verify the connection between the provider web service and the database, you can direct your browser to the following URL:

http://localhost:8080/fr.inria.madynes.androval.providerWS/fetch_defs/get_all. The answer should look like this:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<OVAL_ProviderWS_Answer>
  <OVAL_ProviderWS_AnswerHeader>
    <response_code>0</response_code>
    <details>Definition not found.</details>
    <nb_def_found>0</nb_def_found>
  </OVAL_ProviderWS_AnswerHeader>
  <Definitions/>
</OVAL_ProviderWS_Answer>
```

The "response_code" field can take several values:

- 0 (connection is OK, but no def was found);
- 1 (connection is OK and some def were found);

- 3 (connection is not OK, database error).

To verify that the reporter web service is well connected to the MySQL database, you can direct your browser to the following URL:

http://localhost:8080/fr.inria.madynes.androval.reporterWS/list_results. If you see an error message like "Server database error.", there is a connection error between the reporter WS and the database. Otherwise (message "No result was found." or "OVAL results listing:"), the connection is OK.

On the server, to access the database locally from a shell, use the default mysql login/password : androval – androval. Once you are identified, you can access the ANDROVAL database by typing "USE ANDROVAL;" in the sql shell.

```
user@host:~$ mysql -u androval -p
(enter the pass)
mysql> USE ANDROVAL;
mysql> DESC OVALDEF;
mysql> DESC OVALRES;
```

6.5 Diagnosis Procedures

The Diagnosis Procedures are the first steps that a System Administrator will take to locate the source of an error in a GE. Once the nature of the error is identified with these tests, the system admin will very often have to resort to more concrete and specific testing to pinpoint the exact point of error and a possible solution. Such specific testing is out of the scope of this section.

6.5.1 Resource availability

The minimum requirements can be qualified such as:

Minimum available memory: 256 MB

Minimum available hard disk space: 20 GB

6.5.2 Remote Service Access

Interaction between the Android device and the web services is done through HTTP requests:

- in order to get the latest definitions, client devices do a set of HTTP GET requests to the provider web service on port 8080:
"http://IP_PROVIDER_WS:8080/fr.inria.madynes.androval.providerWS/fetch_defs/by_date/{date}", where {date} (YYYYMMDDhhmmss) is a cookie date representing the last time that

the device probed the provider. In order to get all definitions since the beginning, it is possible to set the value of the cookie to a quite old date, like 20000101010101:

(http://IP_PROVIDER_WS:8080/fr.inria.madynes.androval.providerWS/fetch_defs/by_date/20000101010101);

- to download definitions, the clients then send:
"http://IP_PROVIDER_WS:8080/fr.inria.madynes.androval.providerWS/download/{id}",
where {id} is the download id (given in the previous provider answer) of the definition they want to download. When it receives that HTTP request, the provider then answers by sending the definition content as the HTTP response.

There is only one request that a client can use to communicate with the reporter WS: when an assessment is finished on the client-side, the client sends the result of that assessment using HTTP POST request http://IP_REPORTER_WS/fr.inria.madynes.androval.reporterWS/upload. The result content is passed as POST data.

6.5.3 Resource consumption

On the Android device, the application should remain completely idle most of the time, only becoming active when a system configuration change is detected, or when a new OVAL definition has been downloaded. During an analysis, (RAM usage) Memory consumption will be between 10 MB and 20 MB (depending on the complexity and number of selected tests), and CPU usage might be around 70% to 80% for the duration of the tests. This duration will increase linearly with the number of tests.

On the server, the activity per client is not really noticeable; but activity will increase with the number of clients, and a lot of simultaneous requests from different clients might cause a peak.

On average, there should be less than ten flows per minute between a client and the server (except when new definitions have to be synchronized).

6.5.4 I/O flows

The I/O flow of Web Service uses HTTP, on standard port 80.

7 Security Monitoring / Remediation - Installation and Administration Guide

You can find the content of this chapter as well in the [wiki](#) of fi-ware.

7.1 Introduction

The remediation application provides tools to security operators for proposing cost-sensitive remediations to attack paths. This application is written in Java and has a web-interface provided by the Vaadin library. This document will explain how to install this application.

7.2 Requirements

To be able to install and use the remediation application, there are few requirements, that will be described in this part.

The remediation application has been developed under Mac OS X and successfully tested under Ubuntu 10.04 and CentOS 5. The minimum requirement for the operating system is that the execution of all software components listed here is possible on the OS.

In order to successfully deploy the vulnerability assessment and remediation prototype, the software listed below is required:

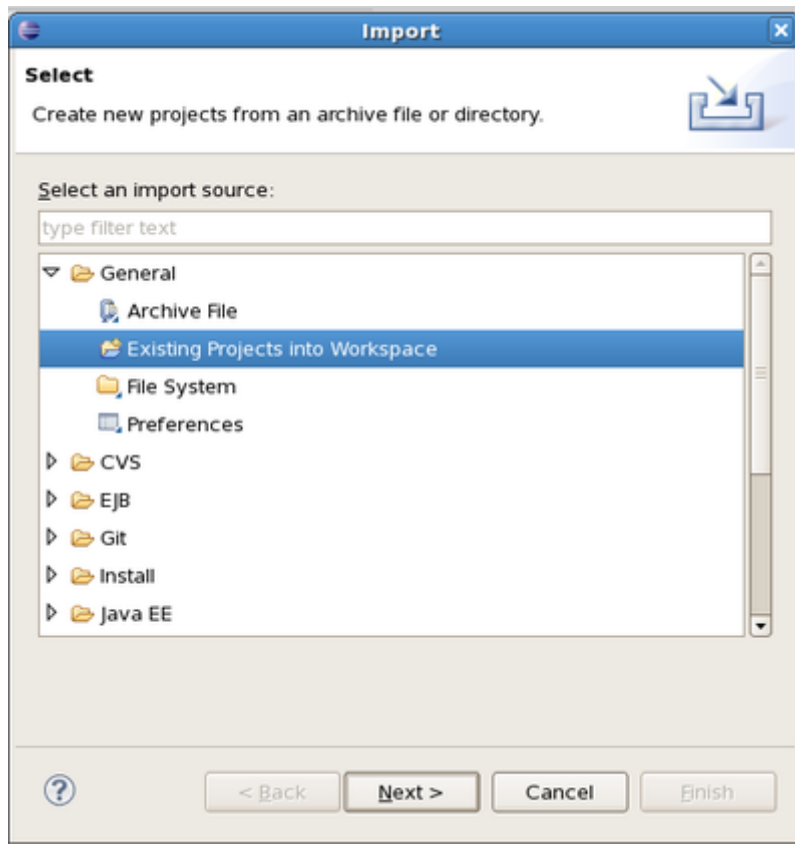
- Eclipse : Juno with Java EE development plugin (to build the application and export the WAR file)
- Java: JRE Open JDK 6 has been used during the build process.
- Apache Tomcat v6 or higher
- Chromium 18.0 or any web browser that provides the same feature set.

7.3 Installation

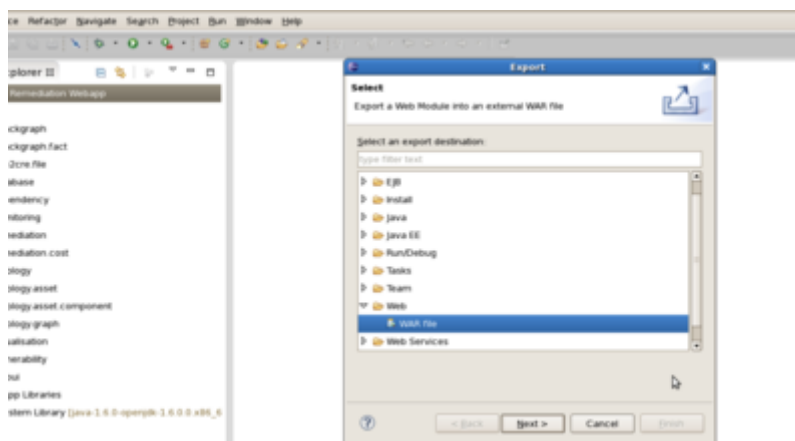
7.3.1 Building application

The build procedure of the remediation application executable (.war) consists of two steps. In the first step, the source of the application is imported into Eclipse and then a .war file for deployment is created. The source code of the remediation application prototype contains a ready to use Eclipse project and class-path descriptor file (.project .classpath). Therefore, to build the remediation application prototype, the entire project has to be imported to Eclipse. In Eclipse, choose File → Import and select "Existing Projects into Workspace" as depicted the following screenshot. In the

next step enter the path to the remediation application sources and select the project for import. Then follow the on-screen instructions to continue the import of the project.



The next step is to create a deployable WAR-File by selecting File → Export (cf screenshot below). In the next step, specify the correct web project, choose the correct Tomcat version and select a proper destination for the WAR file. If the build process has succeeded, the .war file can be found in the directory chosen in step two.



7.3.2 Installation instructions

In this section, the installation of the remediation application is described. Although, the screenshots show the installation process in an CentOS 5 environment, the installation on Windows and other platforms which are capable of executing the Java platform and Tomcat follows the same logic.

7.3.2.1 **Configuration files needed by the application**

The first mandatory step to install the remediation application is to set the appropriate configuration file in the good directory. The resources files mentioned in this section can be found in the `./configuration-files/` folder near the `.war` file. The only requirement is that the configuration file `"config.properties"` must be in the directory `"~/remediation/"`. This file contains the configuration parameters about all the other files needed by the application. Be careful, the home directory (`~`) is for the user launching the servlet. If the servlet is deployed directly with tomcat, it is generally `"var/www"`. In the `"config.properties"` file, several parameters needs to be defined (an example file can be found in `"./configuration/config.properties"`):

- `xsbs-path` The installation path for the `/bin/` directory of XSB (don't forget the `/bin/` at the end if necessary)
- `output-path` The path where the temporary information (for example the attack graph) will be stored. This directory must be both readable and writable by the webserver.
- `mulval-path` The installation path of MulVAL.
- `cost-parameters` The path of the folder where are stored the cost-parameters. This directory must be both readable and writable by the webserver.
- `database-path` The path to the remediation database.
- `topology-path` Path to the test topology used while interaction with CMDB are not functional.

The easier to do is to put all the files and folder referenced in `"config.properties"` in the `"~/remediation/"` folder (`"./cost-parameters/"` `"./database.db"` `"./topology-vuln.xml"` and `"dependencyGraph.xml"`) This can be done with the following command :

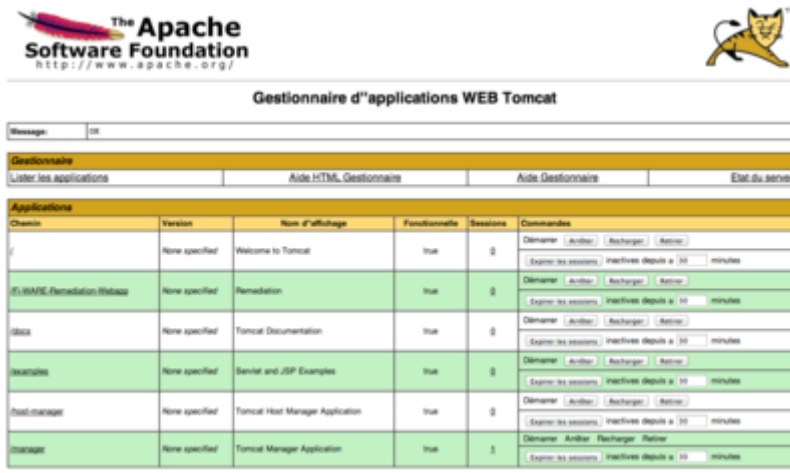
```
mkdir /root /. remediation
cp ./configuration-files/* /root/.remediation/
```

Then, check that all the files and folders paths in the configuration file `"~/remediation/config.properties"` are correct.

7.3.2.2 *Installation of the remediation application on Tomcat*

The remediation application can be deployed directly on Tomcat as a .war file. To do that, please follow the official guidelines for deploying applications to Tomcat :

<http://tomcat.apache.org/tomcat-6.0-doc/manager-howto.html>. You can see in screenshot below screenshots of the deployment of the war file on Tomcat 7.

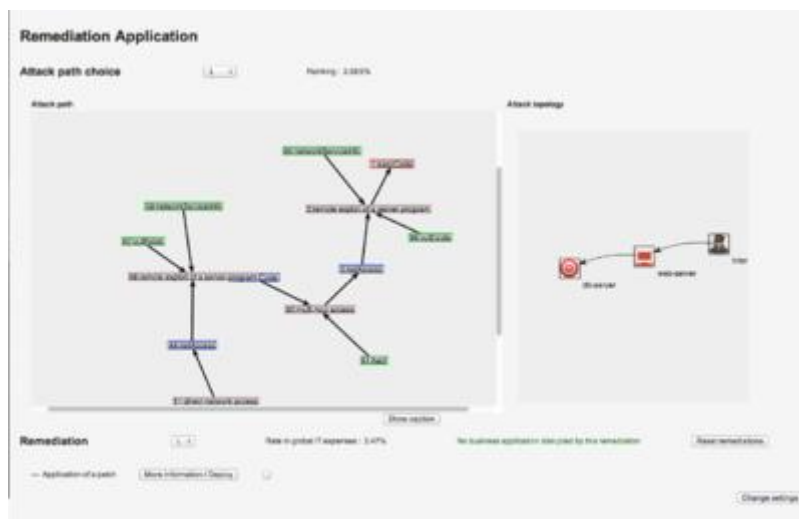


7.4 Sanity check procedures

The Sanity Check Procedures are the steps that a System Administrator will take to verify that an installation is ready to be tested. This is therefore a preliminary set of tests to ensure that obvious or basic malfunctioning is fixed before proceeding to unit tests, integration tests and user validation.

7.4.1 End to End testing

The successful installation of the remediation application can be verified by accessing the URL of the remediation application with the internet browser. This can be done by accessing the correct url (<http://my-ip.address/Fi-Ware-Remediation-Webapp/>). If the installation was successful, the page as depicted in screenshot below is visible and no error messages are shown.



7.4.2 List of Running Processes

"tomcat" process should be running.

7.4.3 Network interfaces Up & Open

Ports used by tomcat (generally 8080 or 80).

7.4.4 Databases

The remediation sqlite database should be located in `/root/.remediation/database.db`. It can be viewed with a software like "SQLite Database Browser" or by using SQL queries such as "SELECT * FROM patches"

7.5 Diagnosis Procedures

In our case, the Diagnosis Procedures are based on Tomcat Apache software. The diagnosis of this server is out of the scope of this project.

7.5.1 Resource availability

The needed resource depends on the number of concurrent requests received on the web server. The minimum requirements can be qualified such as:

Minimum available memory: 512 MB

Minimum available hard disk space: 5 GB

7.5.2 Remote Service Access

N/A

7.5.3 Resource consumption

Resource consumption strongly depends on the load, especially on the number of concurrent requests. The memory consumption of the Tomcat application server should be between 48MB and 1024MB. These numbers can vary significantly if you use a different application server.

7.5.4 I/O flows

The I/O flow uses HTTP, on standard port 80.

8 Security Monitoring / MulVAL Attack Paths Engine Web Application - Installation and Administration Guide

You can find the content of this chapter as well in the [wiki](#) of fi-ware.

8.1 Introduction

The the following guide will give us how to install and administrate web application which is coupled with Mulval Attack Path Engine.

8.2 System Requirements

- This web application is only working with a complete installation of Mulval on CentOS or Ubuntu (32 or 64).
- This web application needs to be hosted on a web server. To do that, you can install the Tomcat Apache latest version downloaded at <http://tomcat.apache.org/>
- You don't need to modify code source when using this web application, the modification of `attackpath.properties` (found inside of `AttackPathEngine.tar.gz`) is sufficient. The required properties are :

`ipaddress = <the_server_name>:<port>`, ex: `secmonitoring.lab.fi-ware.eu:8080`

`rootMulval = <where your Mulval Attack Path is installed>`, ex: `/opt/mulval_v1.1`

`rootApache = <where your Tomcat Apache is installed>`, ex: `/opt/apache-tomcat-7.0.32`

- In order to store the result, we need to create the following folders:

`temp` in `<rootApache>/webapps/<name_of_webapp>/docs/`

`attackgraph` in `/opt/`

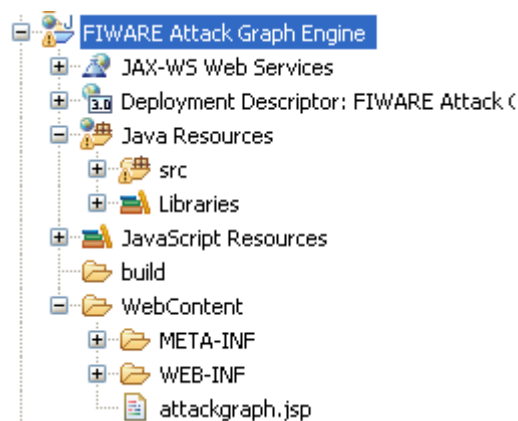
8.3 Installation

The intstallation of this web application required a development environment as Eclipse IDE. It exists other methods to deploy / create a WAR file. According to our test, we recommend the following methods:

Presently, you can install this development environment. You just need to download an Eclipse from <http://www.eclipse.org/downloads/>. The Eclipse IDE for Java EE Developers is good.

And now you need to download AttackPathEngine.tar.gz from FI-WARE Forge and decompress it to the workspace. A workspace is folder which is required by Eclipse in order to store the current development project.

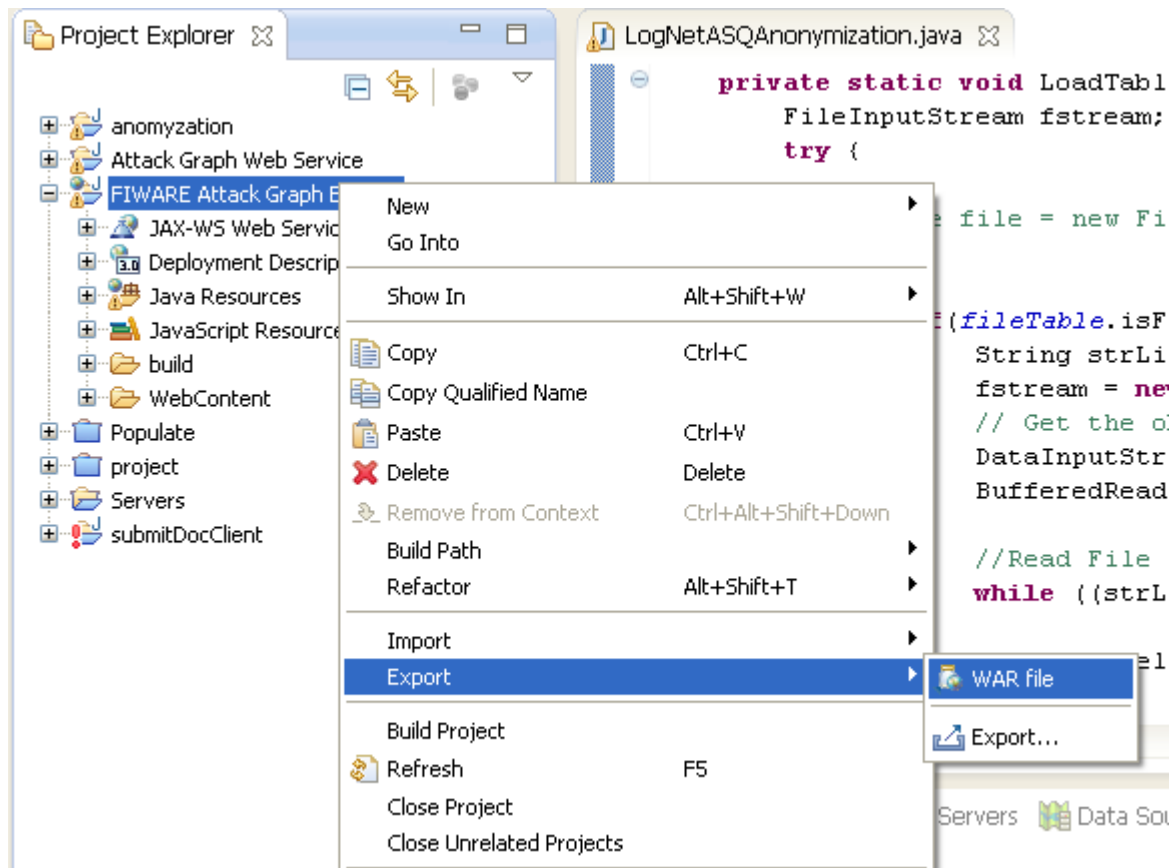
Once the web application project is imported, the files structure looks like:



files structure of web application project

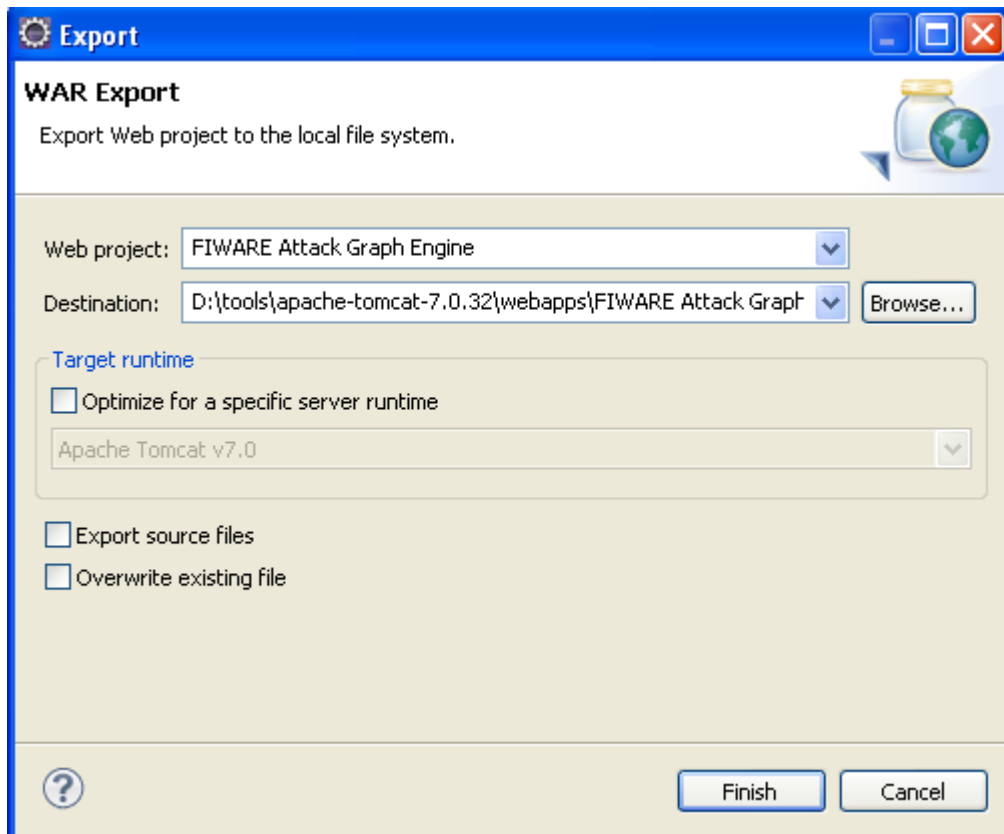
After importing the project, you can export a WAR file. The process of exporting this WAR file is given in following pictures.

You need to select the current project and right click on this project to make appear a menu. Click on Export=>WAR File.



Exporting a WAR file of the current project

After clicking on WAR file, the below menu appears. You must to set where to deploy the WAR File. In general, you must indicate where is the "webapp" folder. This folder is found on "Tomcat Apache Root folder"



Setting where to deploy the WAR File

Click on Finish to deploy the WAR.

8.4 Administration

The administration of Tomcat Apache is given on <http://tomcat.apache.org/tomcat-7.0-doc/>

In our case, we only need how to start and stop the Tomcat Apache server. It's sufficient for the web application to work.

To start the server: `/"installation folder of Tomcat Apache"/bin/catalina.sh start`

And to stop the server: `/"installation folder of Tomcat Apache"/bin/catalina.sh stop`

or just launch `catalina.sh`. By executing this command, the help menu is appeared in order to give us all possibles commands.

8.5 Sanity check procedures

Before connecting this web application, we can check if the web server is running or not.

We can launch the following command to check it:

<installation of Tomcat Apache>/bin/catalina.sh status

8.5.1 End to End testing

Use your browser such as Firefox <http://secmonitoring.lab.fi-ware.eu/AttackPathEngine/attackgraph.jsp>

Upload a NESSUS file (after scanning your environment with NESSUS Vulnerability Scanner. And export your result as file).

Get the result back directly on your web browser.

8.5.2 List of Running Processes

This web application is based on web server which is developed on Java language. The used processes are: apache java

8.5.3 Network interfaces Up & Open

The protocol used is HTTP on port 80.

The eth0 interface must be up in order to be visible outside of the server machine.

8.5.4 Databases

N/A

8.6 Diagnosis Procedures

The Diagnosis Procedures are the first steps that a System Administrator will take to locate the source of an error in a GE. In our case, the Diagnosis Procedures are based on Tomcat Apache software. The diagnosis of this server is out of the scope of this project.

8.6.1 Resource availability

The needed resource depends on the number of concurrent requests received on the web server. The minimum requirements can be qualified such as:

Minimum available memory: 512 MB

Minimum available hard disk space: 256 MB

8.6.2 Remote Service Access

N/A

8.6.3 Resource consumption

Resource consumption strongly depends on the load, especially on the number of concurrent requests. The memory consumption of the Tomcat application server should be between 48MB and 1024MB. These numbers can vary significantly if you use a different application server.

8.6.4 I/O flows

The I/O flow uses HTTP, on standard port 80.

8.7 References

<http://tomcat.apache.org/tomcat-7.0-doc/>

<http://tomcat.apache.org/>

<http://www.eclipse.org/downloads/>

9 Security Monitoring / Scored Attack Paths - Installation and Administration Guide

You can find the content of this chapter as well in the [wiki](#) of fi-ware.

9.1 Introduction

The remediation application provides tools to security operators for proposing cost-sensitive remediations to attack paths. This application is written in Java and has a web-interface provided by the Vaadin library. This document will explain how to install this application.

9.2 Requirements

To be able to install and use the remediation application, there are few requirements, that will be described in this part.

The Scored Attack Paths application has been developed and successfully tested under Windows XP. The minimum requirement for the operating system is that the execution of all software components listed here is possible on the OS.

In order to successfully deploy the vulnerability assessment and remediation prototype, the software listed below is required:

- Eclipse : Juno
- Java: JRE Open JDK 7 has been used during the build process.
- Berkeley DB XML 2.5.16
- Apache Commons Library
- JDom

9.3 Installation

9.3.1 Building application

The build procedure of the scoring application classes requires the jar of the application to be imported into Eclipse. The jar has to be imported to Eclipse.

9.3.2 Installation instructions

In this section, the installation of the remediation application is described. Although, the screenshots show the installation process in an CentOS 5 environment, the installation on Windows and other platforms which are capable of executing the Java platform and Tomcat follows the same logic.

9.3.2.1 *Configuration files needed by the application*

No particular configuration files are needed by the application. For configuration parameters that need to be set, please refer to the following sub-section.

9.3.2.2 *Configuration needed by the application*

There is only one configuration needed, and that is setting the path for the database location. In order to do this, please set the 'BDBXML' string in the Launch class.

9.3.2.3 *Installation of the scoring application*

Since the scoring application comes in the form of an Eclipse project, there is no installation procedure to be followed. Please follow the build procedure described previously.

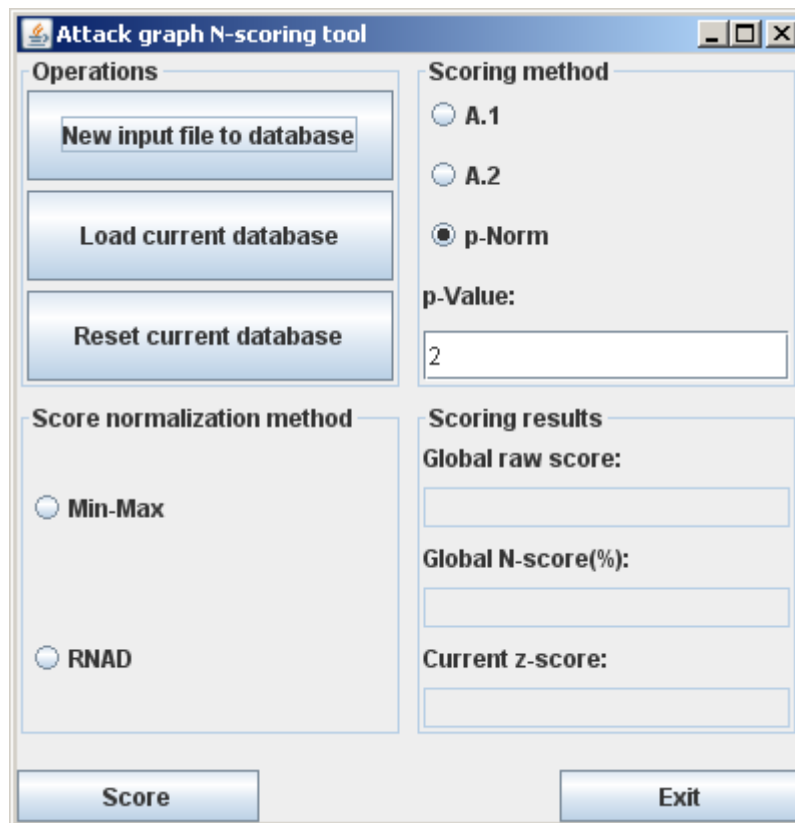
9.4 Sanity check procedures

The Sanity Check Procedures are the steps that a System Administrator will take to verify that an installation is ready to be tested. This is therefore a preliminary set of tests to ensure that obvious or basic malfunctioning is fixed before proceeding to unit tests, integration tests and user validation.

9.4.1 End to End testing

The successful installation of the scoring application can be verified by simply launching the Launch class of the project.

The following control interface then appears:



Main control interface of the scoring application.

9.4.2 List of Running Processes

"Berkeley DB XML" embedded database must be installed. This is not generally considered to be a running process, yet needs to be considered.

9.4.3 Network interfaces Up & Open

No network interfaces are required for this application.

9.4.4 Databases

Berkeley DB XML needs to be installed. The installation path must be added to the path environment variable. This is generally proposed as a choice to the user during the installation of the database.

9.5 Diagnosis Procedures

In the case of Scored Attack Paths, the Diagnosis Procedures are based on the Berkeley DB XML API tests, as well as on the Apache Commons Maths library. The diagnosis of this database and the Commons library is out of the scope of this project.

9.5.1 Resource availability

The needed resource depends on the number of concurrent requests received on the database. The minimum requirements can be qualified such as:

Minimum available memory: 512 MB

Minimum available hard disk space: 5 GB

As for the database parameters, the value of the number of allowed lockers and mutexes is set to 45000 for both.

9.5.2 Remote Service Access

N/A

9.5.3 Resource consumption

Resource consumption strongly depends on the load, especially on the number of concurrent requests. Recommended values for the memory consumption of the Berkeley DB XML database should be between 32MB and 1024MB.

9.5.4 I/O flows

The I/O flow uses disk access.

10 Identity_Management_-_Installation_and_Administration_Guide

You can find the content of this chapter as well in the [wiki](#) of fi-ware.

- [Identity Management - One-IDM Installation and Administration Guide](#)
- [Identity Management - GCP Installation and Administration Guide](#)

11 Identity_Management_-_One-IDM_Installation_and_Administration_Guide

You can find the content of this chapter as well in the [wiki](#) of fi-ware.

11.1 Introduction

This guide provides a description of the installation and administration of the Identity Management (IDM) Generic Enabler in NSN premises. The IDM GE software is provided as a service and not running in the FI-WARE testbed. Therefore the following guide is just an overview about the necessary installation and administration steps and is not intended for testbed application. The IDM software can be remotely accessed from the Internet for collaboration purposes in FI-WARE project. To use an instance of the One-IDM system in your own project you have to contact the representatives for the project.

This is for management issues:

- Seidl, Robert (NSN - DE/Munich) <robert.seidl@nsn.com>

and for installation issues:

- Meyer, Gerald (NSN - DE/Munich) <gerald.meyer@nsn.com>

11.2 Installation

11.2.1 Software environment

The IDM is provided as a servlet for Tomcat 6.0.20. The complete SW stack used in the NSN IDM lab consists of

- Tomcat 6.0.20,
- Sun JDK 1.6.0_16,
- openLDAP2,
- Apache2 web server,
- MySQL database,
- openSUSE 11.3 (64 bit) with KDE GUI.

running as a VirtualBox virtual machine on another (equal) openSUSE fundament. The LDAP directory is used for the complete user management and configuration. It can be managed by tools like JXplorer or other LDAP browsers.

The Apache web server hosts the demo shop which accesses its configuration and warehouse data in the MySQL database. Tomcat is installed in a user directory and is running under this user account. It is started and stopped with the usual startup.sh and shutdown.sh scripts.

11.2.2 IDM provisioning

The .war file for the IDM is copied as usual into the "webapps" directory of Tomcat. After the next restart of Tomcat the IDM will be operational.

11.3 Administration

11.3.1 Network administration

The server running IDM's tomcat is operating behind a NAT router. The router translates the public IP address 85.183.197.168 into the address of the system in the local lab network. Ports 80 and 8443 are opened to the public side of the network without further translation.

In order to be able to access the IDM it should be made sure that these ports are not blocked by a firewall in your organization. Every user should be able to access the URLs below from the public internet (if your "hosts" file is not yet adapted, you might get incomplete pages, but this quick test shows whether the IDM system is basically reachable).

- [Simple web shop as demo application](#),
- [Portal page of IDM](#).

11.3.2 Tomcat

Tomcat is configured to listen to port 443. This is the default configuration. If not, add the following lines to the **server.xml** file:

```
<!-- Internal and External Connector for http(s) -->  
  
<Connector port="443" protocol="HTTP/1.1" connectionTimeout="20000"  
/>
```

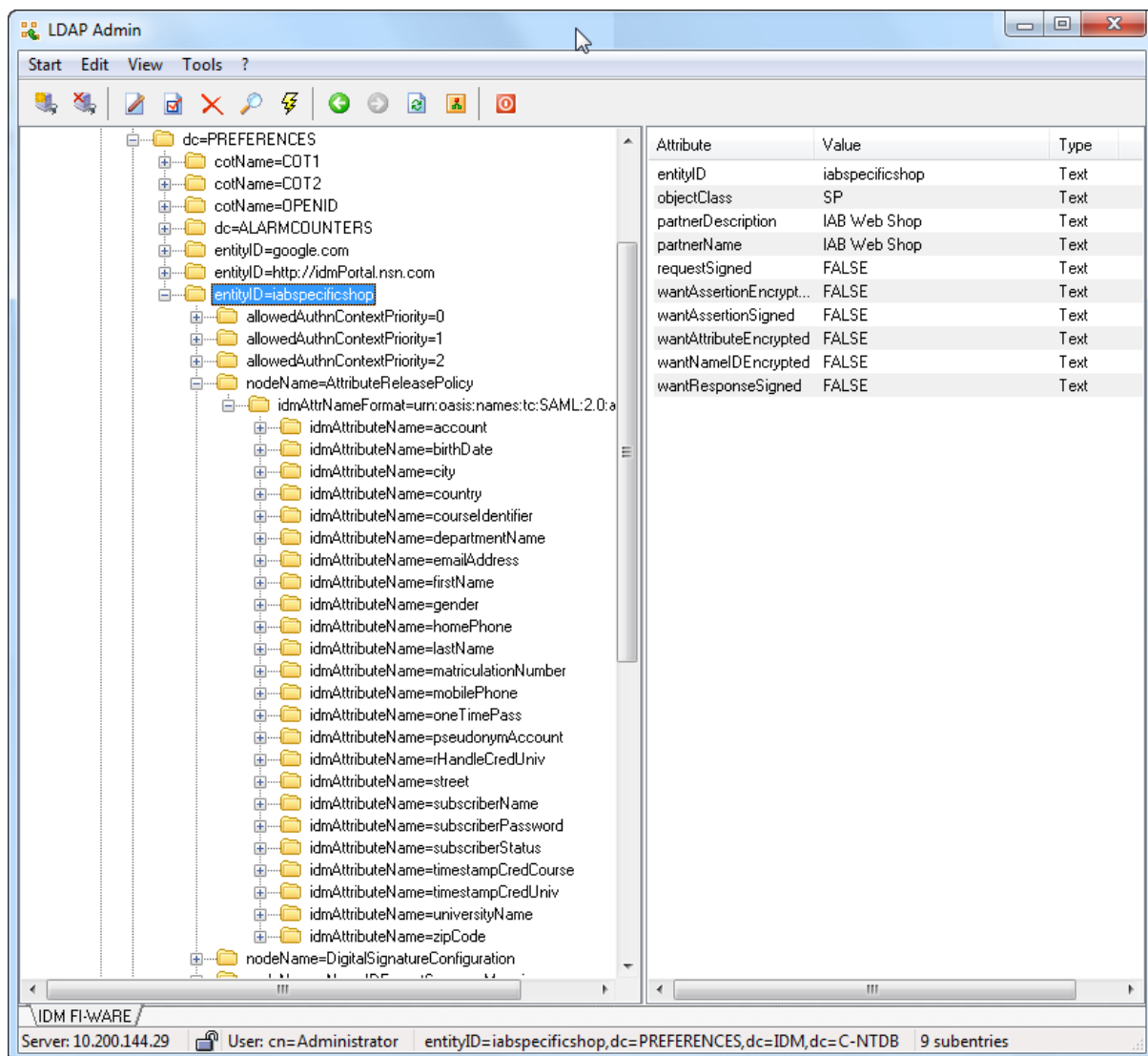
11.3.3 Apache2

In the lab network Apache is listening to ports 80. This is the standard configuration.

11.3.4 IDM

11.3.4.1 *Circle of trust*

In order to establish a trusted relationship between a SAML service and the IDM a suitable entry in the IDM's LDAP directory has to be created. The following screenshot shows a typical result; the service of interest is named "iabspecificshop". This is the key identifier of the service which will later be presented to the IDM in every (encrypted) SAML authentication request.



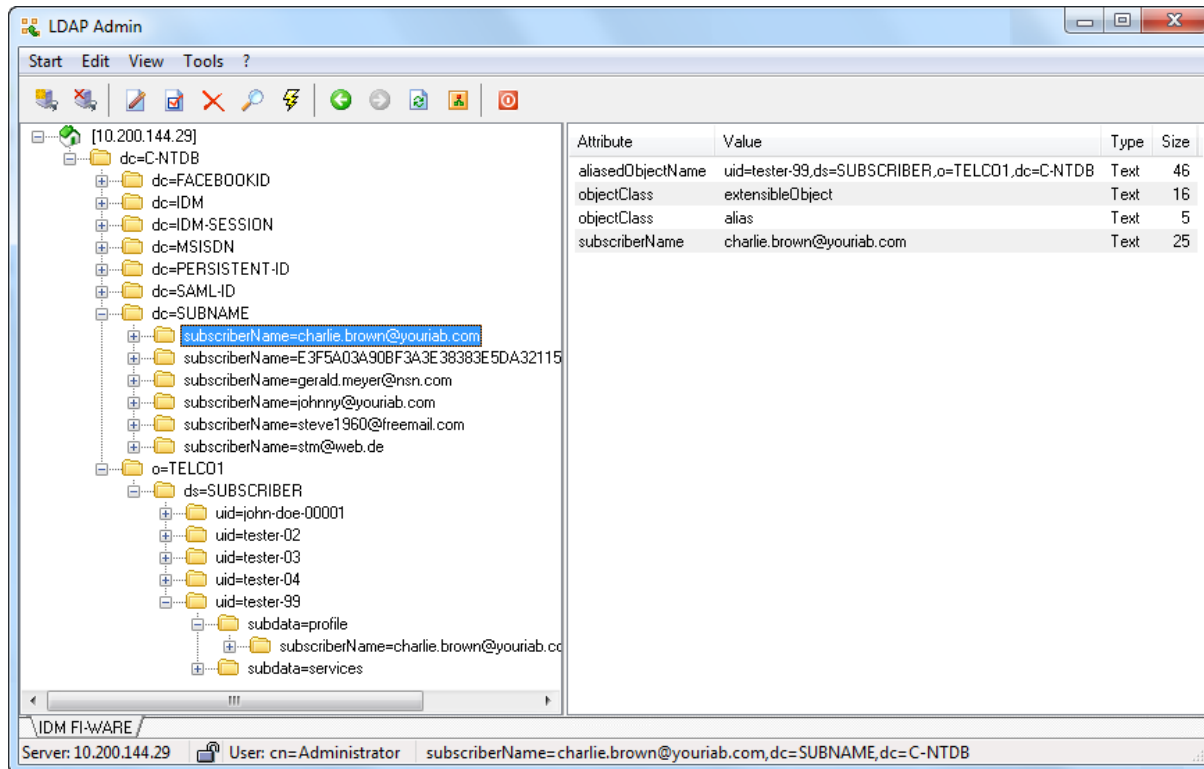
The screenshot shows the LDAP Admin interface. On the left, a tree view displays the directory structure. The 'entityID=iabspecificshop' entry is selected under the 'dc=ALARMCOUNTERS' container. The right pane shows the attributes and values for this entry.

Attribute	Value	Type
entityID	iabspecificshop	Text
objectClass	SP	Text
partnerDescription	IAB Web Shop	Text
partnerName	IAB Web Shop	Text
requestSigned	FALSE	Text
wantAssertionEncrypt...	FALSE	Text
wantAssertionSigned	FALSE	Text
wantAttributeEncrypted	FALSE	Text
wantNameIDEncrypted	FALSE	Text
wantResponseSigned	FALSE	Text

The status bar at the bottom indicates: Server: 10.200.144.29, User: cn=Administrator, entityID=iabspecificshop,dc=ALARMCOUNTERS,dc=IDM,dc=C-NTDB, 9 subentries.

11.3.4.2 *User management*

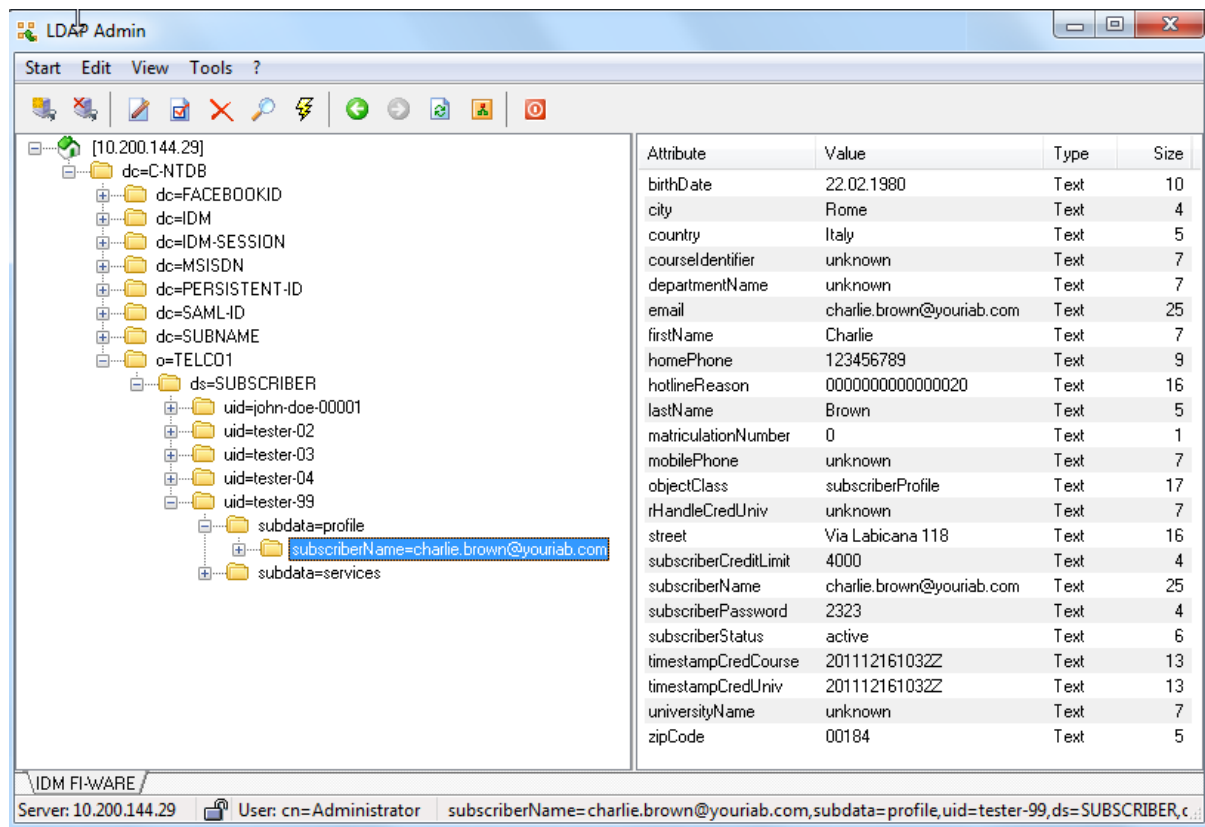
Every user who shall be known to the IDM must be configured in the IDM's LDAP directory. The following two screenshots show in detail the location in the directory tree where a user entry has to be placed and the related user attribute data.



The screenshot shows the LDAP Admin tool interface. On the left, the directory tree is expanded to show the entry `subscriberName=charlie.brown@yourlab.com` under the `dc=SUBNAME` container. On the right, the attributes and values for this entry are displayed in a table.

Attribute	Value	Type	Size
aliasedObjectName	uid=tester-99,ds=SUBSCRIBER,o=TELCO1,dc=C-NTDB	Text	46
objectClass	extensibleObject	Text	16
objectClass	alias	Text	5
subscriberName	charlie.brown@yourlab.com	Text	25

The status bar at the bottom indicates the server is `10.200.144.29`, the user is `cn=Administrator`, and the selected entry is `subscriberName=charlie.brown@yourlab.com,dc=SUBNAME,dc=C-NTDB`.



Attribute	Value	Type	Size
birthDate	22.02.1980	Text	10
city	Rome	Text	4
country	Italy	Text	5
courseIdentifier	unknown	Text	7
departmentName	unknown	Text	7
email	charlie.brown@youriab.com	Text	25
firstName	Charlie	Text	7
homePhone	123456789	Text	9
hotlineReason	0000000000000020	Text	16
lastName	Brown	Text	5
matriculationNumber	0	Text	1
mobilePhone	unknown	Text	7
objectClass	subscriberProfile	Text	17
rHandleCredUniv	unknown	Text	7
street	Via Labicana 118	Text	16
subscriberCreditLimit	4000	Text	4
subscriberName	charlie.brown@youriab.com	Text	25
subscriberPassword	2323	Text	4
subscriberStatus	active	Text	6
timestampCredCourse	201112161032Z	Text	13
timestampCredUniv	201112161032Z	Text	13
universityName	unknown	Text	7
zipCode	00184	Text	5

Server: 10.200.144.29 User: cn=Administrator subscriberName=charlie.brown@youriab.com,subdata=profile,uid=tester-99,ds=SUBSCRIBER,c...

11.3.4.3 **Administrative tools**

The tools mentioned below are usable only with local access to One-IDM, i.e. they are only accessible for operators at NSN premises.

11.3.4.4 **LDAP browser**

Tool for manual subscriber administration in the LDAP directory used by the One-IDM. Good experiences were made with the freely available tools "LDAP admin" and "JXplorer".

11.4 Sanity Check Procedures

The Sanity Check Procedures are the steps that a System Administrator will take to verify that an installation is ready to be tested. This is therefore a preliminary set of tests to ensure that obvious or basic malfunctioning is fixed before proceeding to unit tests, integration tests and user validation.

The following sections have to be filled in with the information or an "N/A" ("Not Applicable") where needed. Do not delete section titles in any case.

11.4.1 End to End testing

To verify that the Identity Management system is running on the application server, make sure that your "hosts" file contains the following entry:

```
217.110.102.211 idm.nsn.com
```

(the file is located at /etc/hosts on Linux systems and at c:\system32\windows\drivers\etc\hosts on Windows). Then direct a web browser to the following URL:

- [Portal page of IDM.](#)

and log in with username "charlie.brown@youriab.com" and password "2323".

11.4.2 List of Running Processes

N/A

11.4.3 Network interfaces Up & Open

To verify that the network interface of the Identity Management server is responding:

using the PING utility, check whether

```
ping 217.110.102.211
```

will result in messages beginning with "64 bytes from 217.110.102.211";

11.4.4 Databases

N/A

11.5 Diagnosis Procedures

The Diagnosis Procedures are the first steps that a System Administrator will take to locate the source of an error in a GE. Once the nature of the error is identified with these tests, the system admin will very often have to resort to more concrete and specific testing to pinpoint the exact point of error and a possible solution. Such specific testing is out of the scope of this section.

The following sections have to be filled in with the information or an "N/A" ("Not Applicable") where needed. Do not delete section titles in any case.

11.5.1 Resource availability

This is identical to the Sanity Check Procedures. If there is any malfunction, intervention of an administrator for the One-IDM system is required.

11.5.2 Remote Service Access

N/A

11.5.3 Resource consumption

N/A

11.5.4 I/O flows

N/A

12 Identity_Management_-_GCP_Installation_and_Administration_Guide

You can find the content of this chapter as well in the [wiki](#) of fi-ware.

12.1 Introduction

This document describes what has to be done if someone intends to use the Global Customer Platform (GCP) identity management system for his service. Currently for the GCP-IDM system there is a demo shop accessible, to understand how the system looks and feels. To use an instance of the GCP system in your own project you have to contact the representative for the project (Wolfgang Steigerwald (T-Labs – DE/Berlin) wolfgang.steigerwald@telekom.de), he will configure an tenant instance on the platform.

12.2 Installation

GCP is a cloud service so there is no installation required. If a project partner wants to use GCP, a tenant instance will be configured by Deutsche Telekom. When the instance is available a link to the instance will be provided.

12.2.1 Access to GCP-IDM server

Access to the GCP is only possible over HTTPS and the above mentioned link, it should be made sure that the port (443) is not blocked by a firewall in your organization. There are three IDM demo systems to get an impression how it works.

- <https://logint2.idm.toon.sul.t-online.de/media-store>
- <https://logint2.idm.toon.sul.t-online.de/music-service>
- <https://logint2.idm.toon.sul.t-online.de/video-service>

How the GCP can be used in the own system is described in the [Onboarding.pdf](#)

12.2.2 Software on demo presentation machine

The local machine must be equipped with an up-to-date web browser (tested with Firefox, Internet Explorer).

12.2.3 Administrative tools

The administration of the GCP instance will be done by Deutsche Telekom.

12.2.3.1.1 Administration interfaces

Deutsche Telekom currently don't provide an administration interface, all administrative work will be done by Deutsche Telekom.

12.2.3.1.2 Programming interfaces

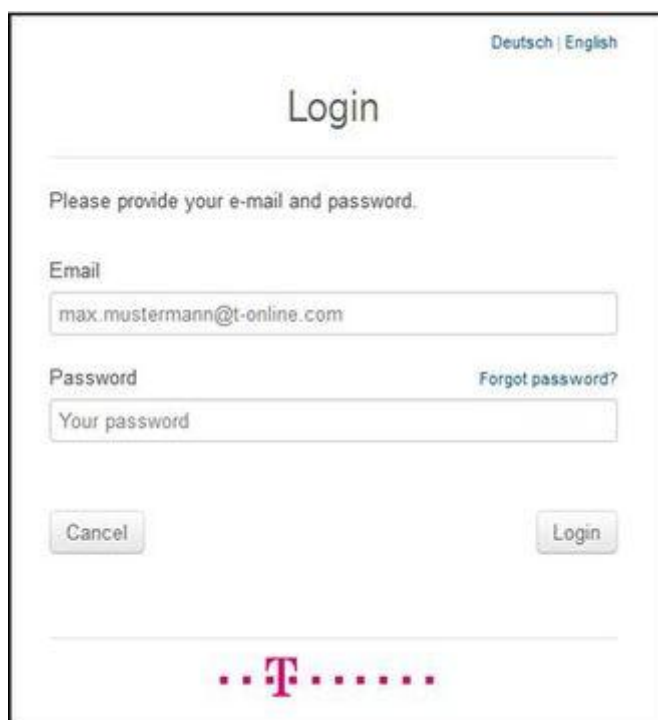
For the GCP-IDM solution we will provide later a REST API for user management.

12.3 Administration

The GCP is a cloud solution and provides an Administration UI for browsers (tested with Firefox and Internet Explorer). Here the basic configuration steps are described. Further administration information will be provided if an instance is requested.

1. Login on the Admin UI

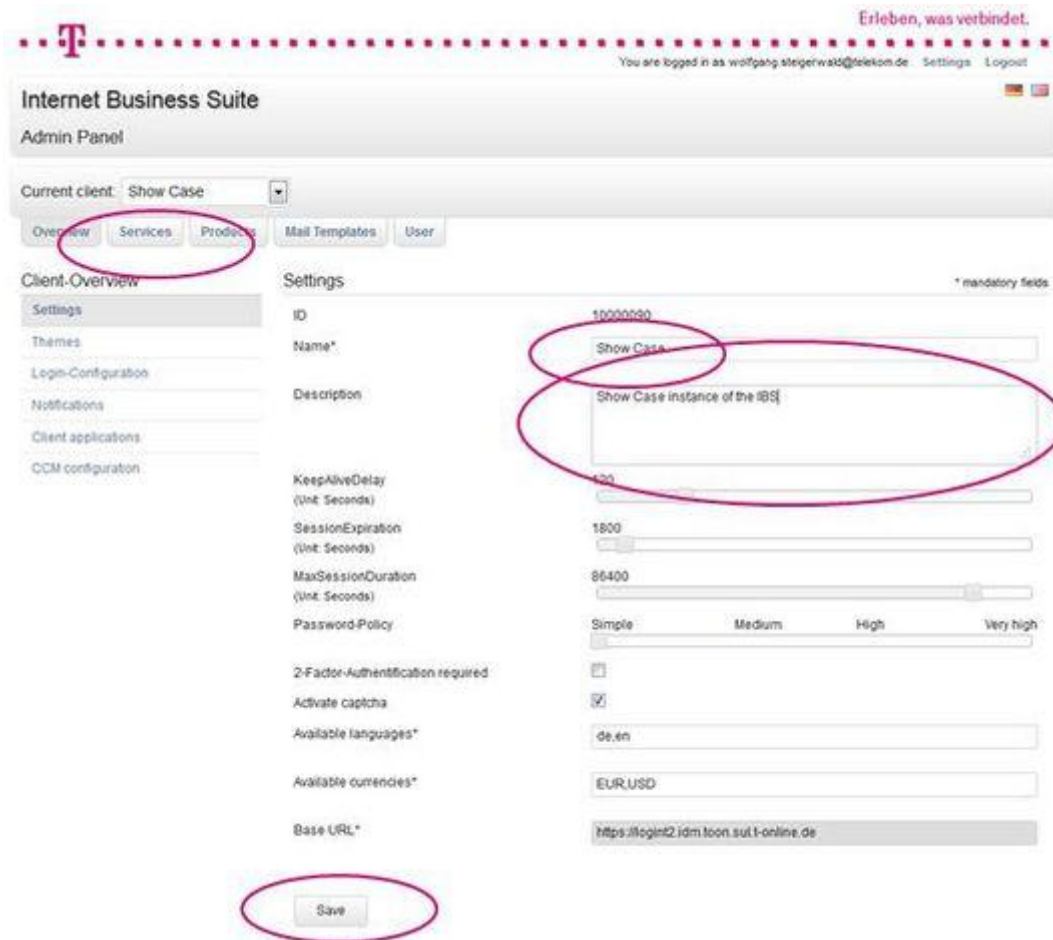
- Open your Browser
- Navigate to <https://logint2.idm.toon.sul.t-online.de/gcp-web-admin>
- Login



2. On the "Overview" tab

- Name your instance
- Provide a short description
- Save your changes

- Navigate to the “Services” tab



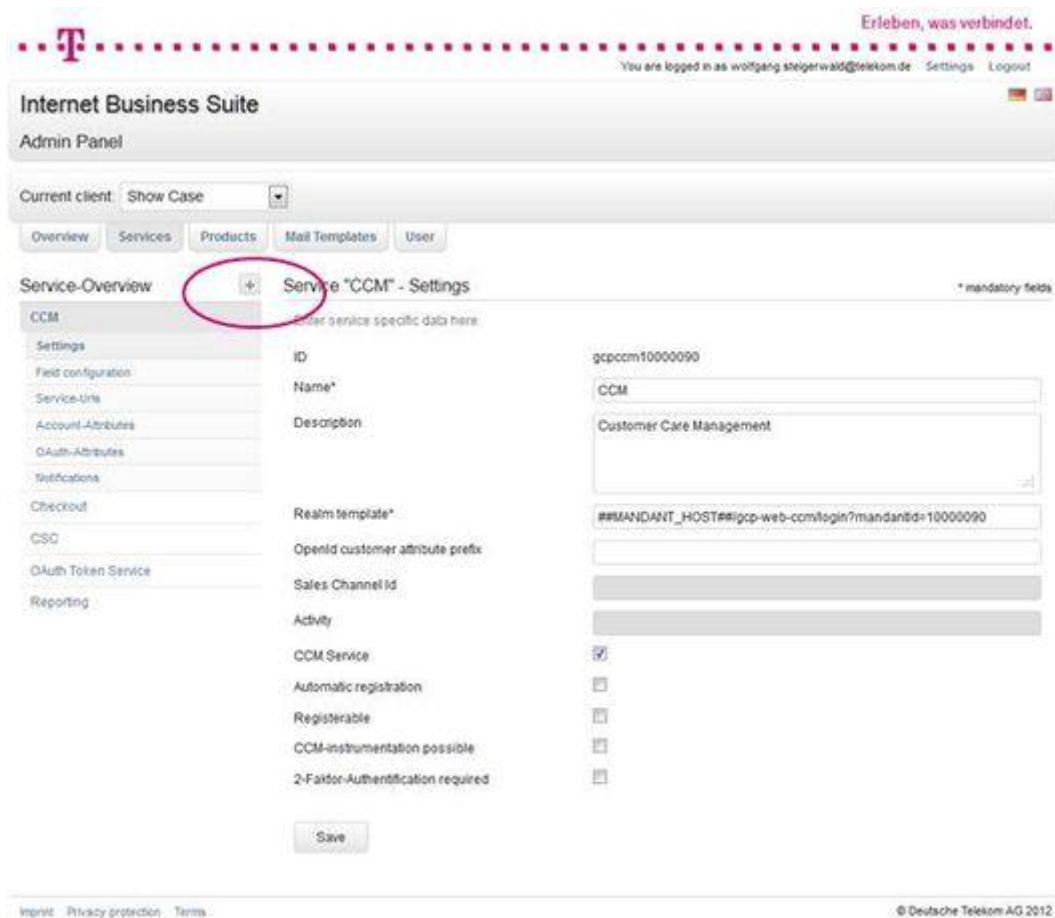
The screenshot shows the 'Internet Business Suite Admin Panel' with the 'Services' tab selected. The 'Current client' is 'Show Case'. The 'Settings' section for this client includes the following fields:

- ID:** 100000000
- Name*:** Show Case
- Description:** Show Case instance of the IBS
- KeepAliveDelay (Unit: Seconds):** 600
- SessionExpiration (Unit: Seconds):** 1800
- MaxSessionDuration (Unit: Seconds):** 86400
- Password-Policy:** Simple
- 2-Factor-Authentication required:** ☐
- Activate captcha:** ☒
- Available languages*:** de,en
- Available currencies*:** EUR,USD
- Base URL*:** https://login2.idm.toon.sut.t-online.de

The 'Save' button at the bottom is highlighted with a red circle.

3. On the “Services” tab

- Add a new service



The screenshot shows the 'Internet Business Suite Admin Panel' for a user logged in as 'wolfgang.steigerwald@telekom.de'. The 'Services' tab is active, and the 'Service "CCM" - Settings' form is displayed. A red circle highlights the '+' icon next to 'Service-Overview' in the left sidebar. The form contains fields for ID, Name, Description, Realm template, Openid customer attribute prefix, Sales Channel Id, and Activity. It also has checkboxes for 'CCM Service', 'Automatic registration', 'Registerable', 'CCM-instrumentation possible', and '2-Factor-Authentication required'. A 'Save' button is at the bottom.

Erleben, was verbindet.

You are logged in as wolfgang.steigerwald@telekom.de Settings Logout

Internet Business Suite

Admin Panel

Current client: Show Case

Overview Services Products Mail Templates User

Service-Overview + Service "CCM" - Settings * mandatory fields

CCM

Settings

Field configuration

Service-URLs

Account-Attributes

OAuth-Attributes

Notifications

Checkout

CSQ

OAuth Token Service

Reporting

Enter service specific data here:

ID gpcpcm10000090

Name* CCM

Description Customer Care Management

Realm template* ##MANDANT_HOST##gcp-web-com/login?mandantid=10000090

Openid customer attribute prefix

Sales Channel Id

Activity

CCM Service ☒

Automatic registration ☐

Registerable ☐

CCM-instrumentation possible ☐

2-Factor-Authentication required ☐

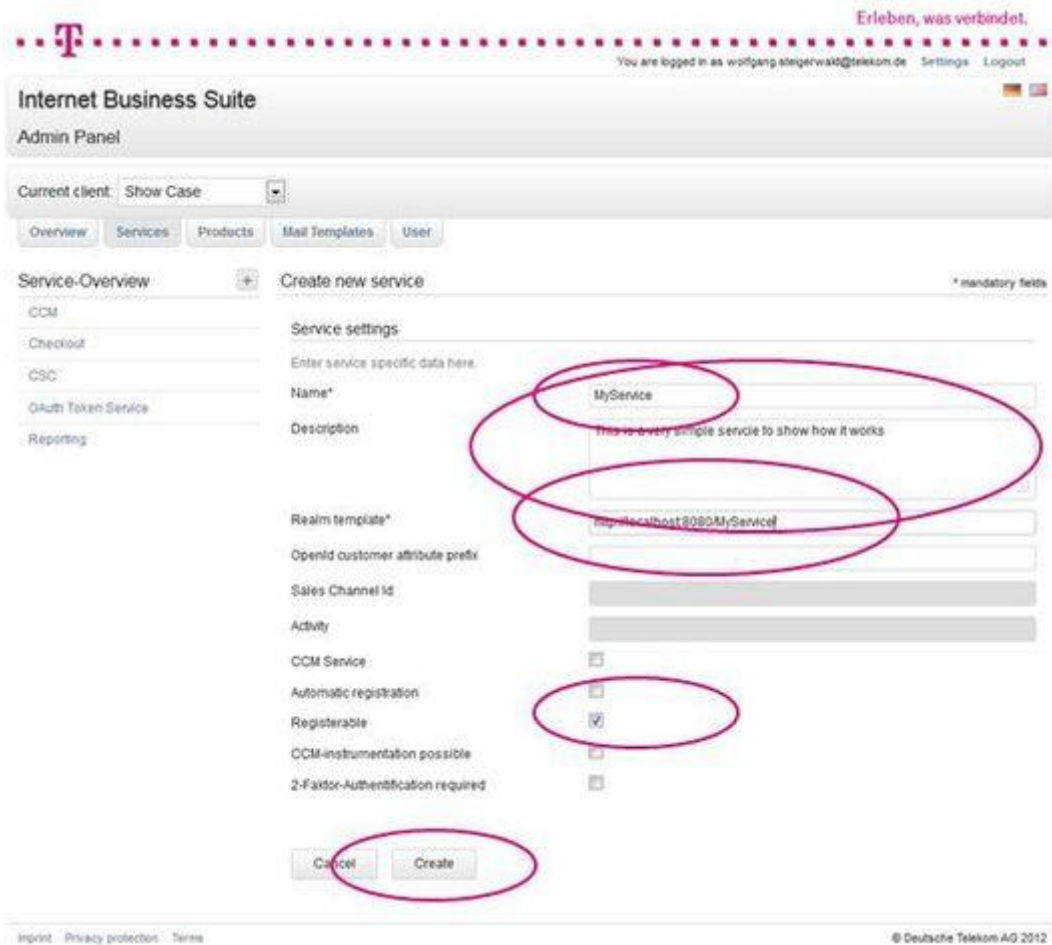
Save

Imprint Privacy protection Terms

© Deutsche Telekom AG 2012

4. On the "Services" tab

- Name your service
- Provide a short description
- Provide the link to your service
- Make the service registerable
- Press „Create“



Erleben, was verbindet.

You are logged in as wolfgang.steigerwald@telekom.de Settings Logout

Internet Business Suite

Admin Panel

Current client: Show Case

Overview Services Products Mail Templates User

Service-Overview

- CCM
- Checkout
- CSC
- OAuth Token Service
- Reporting

Create new service

* mandatory fields

Service settings

Enter service specific data here.

Name* MyService

Description This is a very simple service to show how it works

Realm template* realm:cbhost3080MyService

Openid customer attribute prefix

Sales Channel id

Activity

CCM Service ☐

Automatic registration ☐

Registerable ☒

CCM-instrumentation possible ☐

2-Faktor-Authentification required ☐

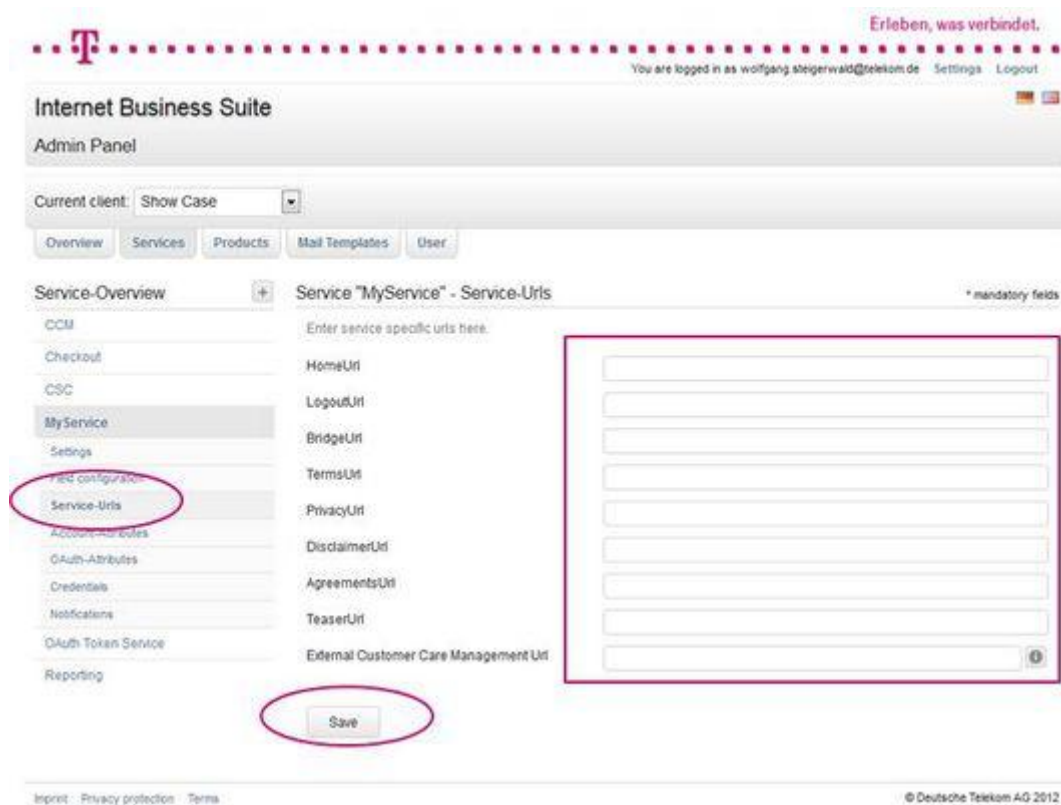
Cancel Create

Imprint Privacy protection Terms

© Deutsche Telekom AG 2012

5. On the "Services" tab

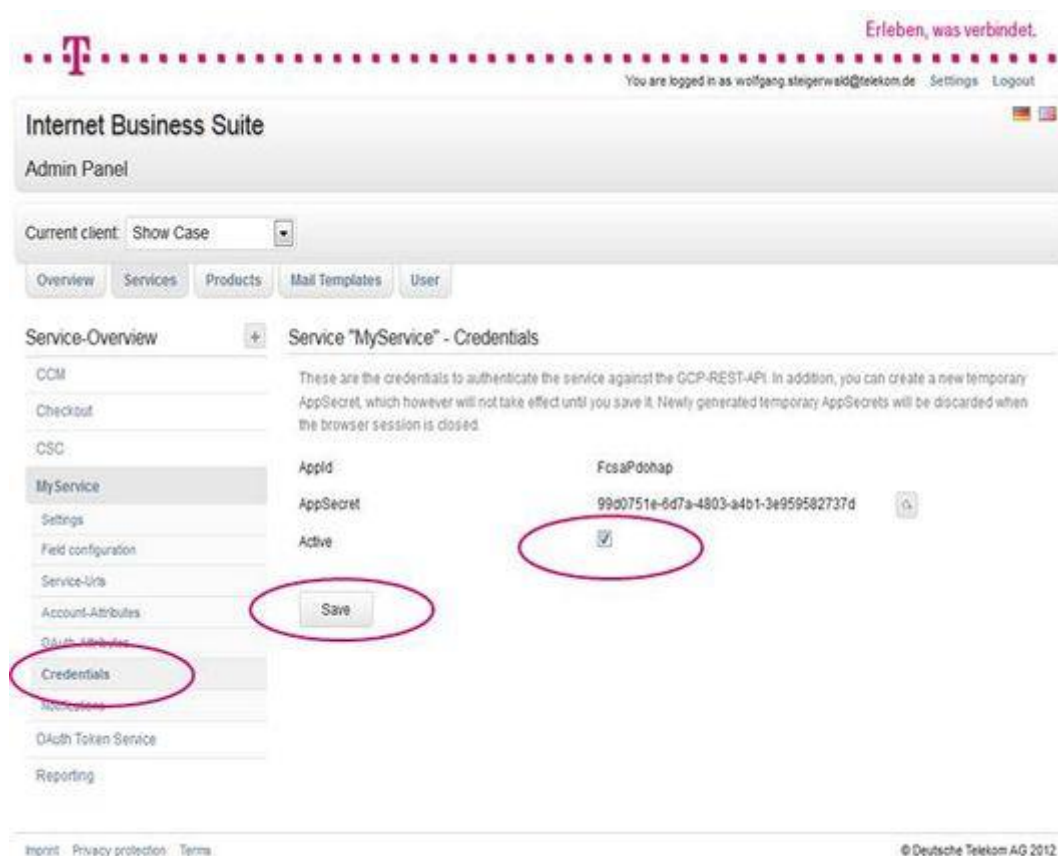
- Select „Service-URLs“
- Provide the required URLs
- Save your changes



The screenshot shows the 'Internet Business Suite Admin Panel' for 'MyService'. The left sidebar contains a 'Service-Overview' menu with 'Service-Urls' highlighted. The main area is titled 'Service "MyService" - Service-Urls' and contains a list of service URLs to be configured. A red box highlights the 'Save' button at the bottom. Another red box highlights the 'Service-Urls' menu item in the sidebar. The 'Service-Urls' list includes: HomeUrl, LogoutUrl, BridgeUrl, TermsUrl, PrivacyUrl, DisclaimerUrl, AgreementsUrl, TeaserUrl, and External Customer Care Management Url. Each item has a corresponding input field for the URL. The 'Save' button is located at the bottom of the configuration area.

6. On the "Services" tab Optional if you later want to use the API

- Select „Credentials“
- Set the checkbox
- Save your changes



12.4 Sanity Check Procedures

The Sanity Check Procedures are the steps that a System Administrator will take to verify that an installation is ready to be tested. This is therefore a preliminary set of tests to ensure that obvious or basic malfunctioning is fixed before proceeding to unit tests, integration tests and user validation.

12.4.1 End to End testing

The test can be done by using the provided link to the GCP instance (see section "Installation")

Example:

```
https://destination/gcp/login?tenant-Id=xyz
```

12.4.2 List of Running Processes

N/A

12.4.3 Network interfaces Up & Open

N/A

12.4.4 Databases

N/A

12.5 Diagnosis Procedures

The Diagnosis Procedures are the first steps that a System Administrator will take to locate the source of an error in a GE. Once the nature of the error is identified with these tests, the system admin will very often have to resort to more concrete and specific testing to pinpoint the exact point of error and a possible solution. Such specific testing is out of the scope of this section.

The following sections have to be filled in with the information or an “N/A” (“Not Applicable”) where needed. Do not delete section titles in any case.

12.5.1 Resource availability

If the service is reachable through the provided link all resources are available.

12.5.2 Remote Service Access

N/A

12.5.3 Resource consumption

N/A

12.5.4 I/O flows

The only expected I/O flow is of type HTTPS, on standard port 443.

13 Data Handling GE - Installation and Administration Guide

You can find the content of this chapter as well in the [wiki](#) of fi-ware.

13.1 System Requirements

First of all copy the installer zip file , extract the archive on your local folder that we will call \$root. All the install files of these software are copied in \$root\softwares

13.1.1 Database

We used the MySQL 5.1.43 database. In order to manage this database we used the EasyPHP 5.3.5 package that includes the Apache HTTP web server, PHP, MySQL and phpMyAdmin. The install including MySQL and PHP can be found here \$root\softwares\EasyPHP-5.3.5.0-setup.exe

13.1.2 Java VM

The executable for the PPL engine requires Java version 1.6. In order to install it you can use the JDK file in \$root\softwares\jdk-6u25-windows-i586.exe. 6. In order to configure correctly 16 your calsspath you can add a new variable PATH with the following value: \Program_Files\Java\jdk1.6.0_21\bin

13.1.3 Mozilla Firefox

The demo is web-based and the user interface should be displayed in a browser. The user interface is a Firefox Plug-in that is compatible with the Firefox 4 Beta 3 version. For this reason this version of the browser must be installed and the setup file can be found in: \$root\softwares\Firefox Setup 4.0 RC 1.exe

13.1.3.1 ***PPL PolicyUI: Firefox Plug-in***

This plug-in is only available in the Install directory under the name PrimeLifePolicyUI.xpi. the file can be found in \$root\partner_software\firefox_plugins\PrimeLifePolicyUI.xpi. In order to install the plug-in you just have to drag and drop this file into the Firefox browser. In order to configure it please set Set the PPL Engine URL to : <http://localhost:9477/api/> and make sure that the suffix is empty. The default is .php

13.2 PPL Installation steps

The PPL engine is composed of three entities, a data subject, a data controller and a third party. Each entity uses its own database. They have to be created before resetting the database and launching the three entities. Here is the list of databases that need to be created:

- ppl
- ppl-dc
- ppl-dc-test
- ppl-ds-test

Once these databases are created you can either reset all databases with `reset-productive.bat` or for a specific entity with `reset-productive-<entity>.bat`.

After resetting the database each entity can be launched with its respective bat file: `dc.bat` and `ds.bat`.

How to manage an entity: Each entity runs its own webserver providing a web interface to manage it. Below is the url for each of them

- Data subject: <http://localhost:9477/>
- Data controller: <http://localhost:8082/>

13.3 Sanity Check Procedures

The Sanity Check Procedures are the steps that a System Administrator will take to verify that an installation is ready to be tested. This is therefore a preliminary set of tests to ensure that obvious or basic malfunctioning is fixed before proceeding to unit tests, integration tests and user validation.

13.3.1 End to End testing

To verify that the Data Handling GE service was correctly deployed on the application server:

- Data subject: <http://localhost:9477/> an HTML frontend with the FIWARE logo is properly displayed
 - Another test can be to navigate with a web browser to URL:
<http://localhost:9477/api/pii>

Execution log using [cURL](#):

```
* About to connect() to <server> port 9477 (#0)
*   Trying <server>... connected
* Connected to <server> (<server> IP) port 9477 (#0)
> GET /api/pii HTTP/1.1
> User-Agent: curl/7.21.3 (i686-pc-linux-gnu) libcurl/7.21.3
OpenSSL/0.9.8o zlib/1.2.3.4 libidn/1.18
> Host: <server>:9477
> Accept: */*
>
< HTTP/1.1 200 OK
< Content-Type: text/xml; charset=utf-8
< Transfer-Encoding: chunked
< Server: Jetty(6.1.x)
<
* Connection #0 to host <server> left intact
* Closing connection #0
<piis>[a list of <pii> nodes, or nothing if no PIIs are
inserted]</piis>
```

- Data controller: <http://localhost:8082> an HTML frontend with the FIWARE logo is properly displayed
 - Another test can be to navigate with a web browser to URL:
<http://localhost:9477/api/pii>

Execution log using [cURL](#):

```
* About to connect() to <server> port 8082 (#0)
*   Trying <server>... connected
* Connected to <server> (<server> IP) port 8082 (#0)
> GET /api/pii HTTP/1.1
```



```
> User-Agent: curl/7.21.3 (i686-pc-linux-gnu) libcurl/7.21.3
OpenSSL/0.9.8o zlib/1.2.3.4 libidn/1.18

> Host: <server>:8082

> Accept: */*

>

< HTTP/1.1 501 Not Implemented
< Content-Type: text/html; charset=utf-8
< Transfer-Encoding: chunked
< Server: Jetty(6.1.x)
<

* Connection #0 to <server> left intact
* Closing connection #0

["status":501]
```

This test actually ensures that the internal implementation correctly parses the incoming request, while at the same time producing a meaningful error message as result of the internal computation (the "["status":501]"). In case of deployment issues, a different error message is returned.

13.3.2 List of Running Processes

- cmd.exe
- java.exe
- mysqld.exe
- apache.exe
- easyphp.exe

13.3.3 Network interfaces Up & Open

N/A

13.3.4 Databases

Check whether MySQL instance is up and running, and reachable from the Application Server;

When using Easy PHP the Admin console is reachable at : <http://127.0.0.1/home/mysql/>

Check if the three DB are initiated as described in the previous section

13.4 Diagnosis Procedures

The Diagnosis Procedures are the first steps that a System Administrator will take to locate the source of an error in a GE. Once the nature of the error is identified with these tests, the system admin will very often have to resort to more concrete and specific testing to pinpoint the exact point of error and a possible solution. Such specific testing is out of the scope of this section.

13.4.1 Resource availability

It is important that the deployment (virtual/physical) machine has at least 2 CPUs, in order to manage effectively concurrent requests to different servers (Data Handling GE components + RDBMS backend). For optimal performances, a system should also have at least 4 GB RAM, and 5 GB storage to be dedicated to Data Handling + RDBMS needs.

Minimal requirements are 1 CPU, 500 MB RAM and 500 MB of application-dedicated storage.

13.4.2 Remote Service Access

N/A

13.4.3 Resource consumption

Resource consumption strongly depends on inputs. In our tests, a 8-core i5 CPU system with 16 GB RAM and 500 GB storage was able to deal with 4 concurrent requests

13.4.4 I/O flows

The only expected I/O flow is of type HTTPS (or HTTP), on standard ports, and inbound only. Requests interactivity should be low, even if a polling mechanism on an API method (getPolicyResult) could involve several requests to be executed from clients.

14 Access Control - Installation and Administration Guide

You can find the content of this chapter as well in the [wiki](#) of fi-ware.

14.1 Introduction

The Access Control GE consists of Identity Manager GE(s) and Thales Access Control Asset. As a preliminary requirement, we assume you have already followed the [Identity Management - GCP Unit Installation and Administration Guide](#). The present document will explain how to install the Access Control Asset and then how to integrate with the IdM GE. In the rest of document, the Thales Access Control Asset will be simply referred to as 'Authorization Server'.

14.2 System Requirements

- Operating System: Linux (Ubuntu Server 12.04 LTS recommended) i686
- Java Platform: JDK 7 (JRE is not enough), Glassfish Server 3.1.2 or later, OpenDJ 2.5-Xpress1 or later
- RAM: 4GB min
- CPU: 2.6 GHz min
- Disk space: 10 GB min
- [FIWARE.OpenSpecification.Security.Identity_Management_Generic_Enabler](#), only DTAG IdM GE (Global Customer Platform) tested so far.

14.3 Installation

14.3.1 Operating System Setup

It is strongly recommended that the system clock be synchronized with the same time server as the IdM GE or with a close statum if using NTP. This is critical for checking the validity of OAuth access tokens generated by the IdM GE.

14.3.2 Certificate Authority Setup

If you have a Certificate Authority already, you can skip this section. This section is about creating a local Certificate Authority (CA) for issuing certificates of the Authorization Server and clients, for authentication, integrity and confidentiality purposes. (For the sake of simplicity, we do not use a subordinate CA, although you should for production, see [command example](#), use the pathlen parameter to restrict number of subordinate CA, pathlen=0 means no subordinate.)

1. Generate the CA keypair and certificate on the platform where the Authorization Server is to be deployed (change the validity argument to your security requirements, example here is 365 days):

```
$ keytool -genkeypair -keystore taz-ca-keystore.jks -alias taz-ca -dname "CN=Thales AuthzForce CA, O=FIWARE" -keyalg RSA -keysize 2048 -validity 365 -ext bc:c="ca:true,pathlen:0"
```

2. Export the CA certificate to PEM format for easier distribution to clients.

```
$ keytool -keystore taz-ca-keystore.jks -alias taz-ca -exportcert -rfc > taz-ca.pem
```

14.3.3 Application Server Setup (Glassfish)

First run the following command on the Glassfish domain where the Authorization Server webapp will be deployed:

```
$ asadmin create-jvm-options -Dcom.sun.enterprise.overrideablejavaxpackages=javax.ws.rs,javax.ws.rs.core,javax.ws.rs.ext
```

14.3.3.1 **Server Basic Security**

Follow the *Security Guide* of your respective Glassfish version. In particular, do the following:

- Change the master and admin password of the Glassfish domain (where the Authorization Server webapp is to be deployed) if not changed from the default values ('changeit' for Glassfish)

```
$ change-master-password domain1
```

- Enable secure admin
- Disable XPoweredBy header on all HTTP listeners
- Change Server header

```
$ asadmin create-jvm-options -Dproduct.name="FIWARE Authorization Server"
$ asadmin restart-domain
```

14.3.3.2 **Server Certificate Setup**

1. For all keystores created in this section, use the same password as in the **change-master-password** command run in the previous section.

2. On the Authorization Server platform, generate the Authorization Server keypair, you must change the `dname` parameter to the domain name that all users will use to access the server over TLS transport:

```
$ keytool -genkeypair -keystore taz-server-keystore.jks -alias  
taz-server -dname "CN=taz.example.com, O=FIWARE" -keyalg RSA -  
keysize 2048
```

3. Make the server certificate signed by the CA, typically by sending a CSR (Certificate Signing Request) to the CA. For example, with the CA created in previous section, do as follows (change the password according to what you entered previously when creating the server keystore, and change the validity period according to your security requirements):

```
$ keytool -storepass password -keystore taz-server-keystore.jks  
-certreq -alias taz-server | keytool -gencert -storepass  
password -keystore taz-ca-keystore.jks -alias taz-ca -validity  
365 -ext ku:c=dig,keyEncipherment -rfc > taz-server-cert.pem
```

4. Import the CA certificate into the server truststore

```
$ keytool -importcert -keystore taz-server-keystore.jks -file  
taz-ca-cert.pem -alias taz-ca -noprompt
```

5. Import server certificate signed by the CA into the server keystore:

```
$ keytool -importcert -keystore taz-server-keystore.jks -file  
taz-server-cert.pem -alias taz-server
```

6. Create the server truststore that will be used to validate certificates of clients accessing the server (CA certificate only, truststore is created on-the-fly by the keytool command):

```
$ keytool -importcert -noprompt -alias taz-ca-cert -file taz-  
ca-cert.pem -keystore taz-server-truststore.jks
```

14.3.3.3 **Server TLS Setup**

1. Backup the `keystore.jks` file in the Glassfish domain directory.
2. Stop Glassfish domain (replace `s1as` cert with new one), from Glassfish domain config directory (e.g. `${GLASSFISH_HOME}/domains/domain1/config`), run (taz-server-keystore.jks has been created in the previous section):

```
keytool -importkeystore -srckeystore taz-server-keystore.jks -  
destkeystore keystore.jks -srcalias taz-server -destalias s1as
```

3. For the truststore, follow the [6 tutorial - Working with Digital Certificates](#), i.e. set it via JVM option `javax.net.ssl.truststore` and restart the Glassfish domain.
4. Configure group assignment of Clients authenticated by certificates: edit the Certificate Realm in Glassfish admin console and set the **Assign-Groups** property to *authenticated*.

14.3.3.4 **Server Performance Tuning**

Follow the *Performance Tuning Guide* of your respective Glassfish version.

14.3.4 User Role Management Setup

This section is about configuring OpenDJ directory for storing user-role assignments required to access the Authorization Server interface.

1. If you do not have a base DN setup in OpenDJ directory, create one, such as 'dc=fiware,dc=eu'.
2. Create a user branch (organizationalUnit objectClass), such as 'ou=users', if you do not have one already, in the above directory base.
3. Create a role branch (organizationalUnit objectClass), such as 'ou=roles' in the same base.
4. Configure OpenDJ to enforce user mail uniqueness on the user branch

```
$ dsconfig create-plugin --port 4444 --hostname localhost --bindDN "cn=Directory Manager" --bindPassword password --plugin-name "Unique mail" --type unique-attribute --set base-dn:ou=users,dc=example,dc=com --set type:mail --set enabled:true --trustAll --no-prompt
```

14.3.5 Superadmin Account Setup

This section is about creating credentials for a client (e.g. end-user) of the Authorization Server, and assigning him/her the Superadmin role. This role is necessary to start managing the server with the API.

14.3.5.1 **Client Certificate Setup**

Superadmin user - like any other client accessing the Authorization Server - is required to authenticate with a certificate to access certain endpoints of the API. Follow these steps to create a such a valid client certificate:

1. On the client host, e.g. some John Doe user's desktop, generate the client keypair (the `dname` argument does not matter as it will be overridden by the CA when issuing the certificate, yet you must provide one at this step):

```
$ keytool -genkeypair -keystore johndoe-taz-client-keystore.jks
-alias johndoe-taz-client -keyalg RSA -keysize 2048 -dname
"CN=John Doe" > johndoe-taz-client.csr
```

2. On the CA/Authorization Server host, make the client certificate signed by the CA using the above generated client CSR; make the subject DN unique by means of a UUID, for example with OSSP UUID command-line tool (UUID version 1 is recommended for one-time globally unique identifier, with the downside that the generated UUID may disclose information about the host MAC address)

```
$ keytool -storepass password -keystore johndoe-taz-client-
keystore.jks -certreq -alias johndoe-taz-client | keytool -
storepass password -keystore taz-ca-keystore.jks -gencert -
alias taz-ca -dname "CN=$(/usr/bin/uuid -v1), OU=Thales
AuthzForce Users, O=FIWARE" -validity 365 -rfc > johndoe-taz-
client-cert.pem
```

3. On the client host, import the CA certificate into the client keystore:

```
$ keytool -importcert -keystore johndoe-taz-client-keystore.jks
-file taz-ca-cert.pem -alias taz-ca -noprompt
```

4. Import the signed client certificate into the client keystore:

```
$ keytool -importcert -keystore johndoe-taz-client-keystore.jks
-file johndoe-taz-client-cert.pem -alias johndoe-taz-client
```

5. For clients that do not have native support for JKS (Java keystore), such as most common web browsers, you may export the JKS to PKCS12 format:

```
$ keytool -importkeystore -srckeystore johndoe-taz-client-
keystore.jks -srcalias johndoe-taz-client -deststoretype PKCS12
-destkeystore johndoe-taz-client-keystore.p12
```

14.3.5.2 **User-Role Assignment**

After creating the certificate of your first Superadmin user, say John Doe, assign him the Superadmin role:

1. Create user entry (inetorgperson objectClass) with DN 'uid=\${CN}' in the user branch created in section 'User Role Management Setup', where \${CN} is the value of the CN attribute in the user certificate subject DN, as defined in previous section 'Client Certificate Setup'.
2. Create entry 'cn=Superadmin' with objectClass group.

14.3.6 Authorization Server Application Setup

1. Download the release package from the FI-WARE PPP Restricted Project website, package "Security-AccessControlGE".

14.3.6.1 **Configuration**

1. Copy all files from *authforce-config* directory in the package to the Glassfish domain config directory as follows: `${GLASSFISH_DOMAIN_DIR}/config/authzforce/`.
2. In the *attributeFinderModule* with attribute `class="com.thalesgroup.authzforce.finder.LdapSubjectAttributeFinder"`, replace *url* with your OpenDJ directory server URL; *bindDn*, *bindPassword* with the credentials of an OpenDJ admin account that has read permissions on the user and role branch set up in section *User Role Management Setup*; *baseDn* with the user branch DN, and all occurrences of `ou=roles,dc=example,dc=com` with the actual role branch DN.

14.3.6.2 **Web Application Deployment**

1. Deploy the war file from the release package to Glassfish server as a web application.

14.4 Administration

14.4.1 Glassfish Webapp Administration

Most administration of the Authorization Server webapp, security and monitoring can be done with standard Glassfish admin console or CLI operations.

14.4.2 Application-Specific Administration

For application-specific configuration, refer to *Configuration* section in Installation. Application error logs are located in *taz.log* file in glassfish domain *logs* directory. Application access logs are located in *access/taz.log* file in glassfish domain *logs* directory.

Policy administration is part of the Authorization Server API and is addressed in the *User Guide*.

14.5 Sanity check procedures

The Sanity Check Procedures are the steps that a System Administrator will take to verify that the installation is ready to be tested. This is therefore a preliminary set of tests to ensure that obvious or basic malfunctioning is fixed before proceeding to unit tests, integration tests and user validation.

14.5.1 End to End testing

To check the proper deployment and operation of the Authorization Server, perform the following steps:

1. Configure your favorite HTTP client to use the Superadmin certificate created during installation phase, as client certificate (for authentication). Examples of free HTTP clients are curl, SoapUI, your favorite web browser. If you choose the latter, use one of the browser's REST client plugins and import the certificate in PKCS12 format into your browser's personal certificates.
2. Get the list of policy administration domains by doing the following HTTP request, replacing *host* with the Glassfish server hostname that must match the CN of the server certificate set up during installation phase, and *port* with the HTTPS port of the server, and *contextRoot* with the context root of the webapp deployed during installation:
 - Method: GET
 - URL:

`https://${host}:${port}/${contextRoot}/pap/domains`
 - Accept header: application/xml
3. If you get a security warning about an untrusted certificate, and you want to get rid of it definitively, import the CA certificate generated during installation phase, into the client's trusted CAs, and send the request again.
4. Check the response which should have the following headers and body (there may be more headers which do not require checking here):
 - Status Code: 200 OK
 - Content-Type: application/xml
 - Body:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<ns2:resources xmlns:ns2="http://thalesgroup.com/authzforce/model"
xmlns:ns3="http://www.w3.org/2005/Atom">

    ... list of links to policy domains omitted here...

</ns2:resources>
```

You can check the exact body format in the representation element of response code 200 for method *getDomains* in the WADL (Web Application Description Language) document that describes the RESTful API of the service and is available at the following URL:

```
https://${host}:${port}/${contextRoot}/?_wadl
```

14.5.2 List of Running Processes

- One or more `java.exe` processes for Glassfish;
- One or more `java.exe` process for OpenDJ.

14.5.3 Network interfaces Up & Open

- TCP 22
- TCP 80
- TCP 389
- TCP 443
- TCP 636
- TCP 4848
- TCP 8443
- TCP 9009

14.5.4 Databases

To check the OpenDJ directory server status, run OpenDJ control panel via `control-panel.sh` command in OpenDJ directory.

14.6 Diagnosis Procedures

1. Perform the test described in *End to End Testing*:

```
HTTP GET https://${host}:${port}/${contextRoot}/pap/domains
```

2. If you get an error 404 Not Found, make sure the webapp is deployed and enabled in Glassfish
 1. In the admin WUI, if the webapp is not listed in Applications tab, deploy it.
 2. In the admin WUI, if webapp is listed as disabled, select it and click the *Enable* button.

3. In both cases, if the enable or deployment action fails, check the `${GLASSFISH_DOMAIN_ROOT}/logs/server.log`.
3. If you get an error 401 or 403, see the Authorization Server logs as described in section *Application-Specific Administration*. In particular, if you see JNDI LDAP connection error, check that the OpenDJ server status by running OpenDJ control panel locally. Check the Superadmin role is still properly assigned for your certificate as done during installation. For more information, see [OpenDJ directory server troubleshooting](#).
4. If you get a Connection Refused, check that the Glassfish server is running by checking the server status in Glassfish admin WUI (Web User Interface)
 1. If you are unable to connect to the admin WUI, check that the Glassfish domain is started locally with command `asadmin list-domains`. If not, start the domain:
`asadmin start-domain`
 2. If the command fails, check the domain logs in `${GLASSFISH_DOMAIN_ROOT}/logs/server.log`. For more information, see [Glassfish 3.1 Troubleshooting Guide](#).
 3. If the command succeeds, or the domain is already started, and you still get a Connection Refused, check the server status via Glassfish admin WUI for the domain, and start the server if it is not started. Check you are using the right HTTP port in the list of HTTP ports given in the server *General Information* of the admin WUI.
 4. If you are able to connect locally but not remotely, check the network link, i.e. whether any network filtering is in place on the host or on the access network, or other network issue: network interface status, DNS/IP address resolution, routing, etc.

For other status codes, see the Authorization Server logs as described in section *Application-Specific Administration*.

14.6.1 Resource availability

To have a healthy enabler, the resource requirements listed in *System Requirements* must be satisfied, in particular:

- Minimum RAM: 4GB
- Minimum CPU: 2.6 GHz
- Minimum Disk space: 10 GB

14.6.2 Remote Service Access

The Access Control GE may communicate with Identity Management GEs on HTTPS (port 443), in particular the OAuth Token Endpoint and User Attribute Management API endpoint.

14.6.3 Resource consumption

The resource consumption strongly depends on the number of users registered in the system, the number of concurrent users and requests per user, the number of policy domains (a.k.a. tenants in this context) managed by the Authorization Server, and the complexity of the policies defined by administrators of each domain.

The memory consumption shall remain under 80% of allocated RAM. See *System Requirements* for the minimum required RAM.

The CPU usage shall remain under 80% of allocated CPU. See *System Requirements* for the minimum required CPU.

As for disk usage, at any time, there should be 1GB free space left on the disk.

14.6.4 I/O flows

- HTTPS flows with possibly large XML payloads to port 443, 8443, 4848
- HTTP flow to port 80
- LDAPS flows to 636.

15 DB Anonymizer GE - Installation and Administration Guide

You can find the content of this chapter as well in the [wiki](#) of fi-ware.

15.1 Introduction

DB Anonymizer is a Java Web Application, packaged in a WAR file. It uses a MySQL database in order to evaluate the adequateness of a specific anonymization policy, to be used to disclose a dataset. The dataset (in the form of a MySQL table SQL dump) will be passed to DB Anonymizer through its ReSTful API, together with an anonymization policy to analyse. At that point, DB Anonymizer will upload in its MySQL instance the received dataset, and perform the policy analysis.

As said, DB Anonymizer needs to receive a dump of a MySQL table, containing all data, together with a disclosure policy. Both inputs are mandatory to let the service's algorithm to be able to evaluate the effectiveness of the disclosure policy. Once the policy is evaluated, the table is dropped from the DB and the file dump is erased. The application server encapsulation model permits a complete isolation of each user's data, and any intermediate result created during algorithm's execution is deleted immediately at the end of the computation.

DB Anonimizer uses:

1. a MySQL instance
2. a Java Application Server (any should work, but only Apache Tomcat is supported)

In the remainder of this section, the configuration of these two elements is presented.

15.2 Installation

15.2.1 Database

DB Anonymizer has been tested using MySQL v5.5.X.

15.2.1.1.1 DB Configuration

DB Anonymizer expects to be able to access to a MySQL schema, **obligatorily called "census"**, using an account that has full grants on that specific schema. This is necessary to be able to create and drop tables, views, indexes and so on.

The DB schema should be empty, except for a table, that can be created using the following SQL statements:

```
DROP TABLE IF EXISTS `results`;
```

```
CREATE TABLE `results` (  
  `idresults` int(11) NOT NULL AUTO_INCREMENT,  
  `GID` bigint(20) NOT NULL,  
  `result` text NOT NULL,  
  `computed` tinyint(1) NOT NULL,  
  PRIMARY KEY (`idresults`)  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

15.2.2 Application Server

While no specific Application Server functionalities are used, the application has been tested with Apache Tomcat 6.X and 7.X.

15.2.2.1 *Application Server configuration*

It is necessary to provide DB Anonymizer with a JDBC MySQL datasource, identified by a JNDI resource, that will be used in its computation processes.

To this extent, for Apache Tomcat, it is necessary to add a file named "context.xml" (or to edit it, if already existing) in its configuration directory (generally "conf").

It should contain the following information:

```
<?xml version="1.0" encoding="UTF-8"?>  
<Context>  
  <!-- Specify a JDBC datasource -->  
  
  <Resource name="jdbc/censusdb" auth="Container"  
    type="javax.sql.DataSource" username="<username for MySQL>"  
password="<password for MySQL>"  
    driverClassName="com.mysql.jdbc.Driver"  
    url="jdbc:mysql://<address of MySQL DBMS >:3306/<database  
schema name>?autoReconnect=true"  
    maxActive="100" maxIdle="30" />
```

</Context>

Attributes *maxActive* and *maxIdle* are needed in order to enable the creation of a connection pooling to improve DB Anonymizer performances. These values should be fine tuned according to the characteristics of the deployment server. Please refer to [this link](#) for more information.

15.2.2.2 **DB Anonymizer deployment**

DB Anonymizer WAR package can be installed by copying it into "webapp" folder in Apache Tomcat. To install it on other Java Application Servers, please refer to their specific application server guidelines.

15.2.2.3 **Remark: HTTPS/SSL**

It is highly recommended to allow incoming connection to DB Anonymizer only through HTTPS. This can be achieved by using a front-end HTTPS server that will proxy all requests to DB Anonymizer, or by configuring the Application Server in order to accept only HTTPS/SSL connection. In the latter case, please refer to [this link](#) for more information.

15.3 Sanity Check Procedures

The Sanity Check Procedures are the steps that a System Administrator will take to verify that an installation is ready to be tested. This is therefore a preliminary set of tests to ensure that obvious or basic malfunctioning is fixed before proceeding to unit tests, integration tests and user validation.

15.3.1 End to End testing

To verify that the DB Anonymizer application was correctly deployed on the application server:

- check whether, at the URL `http(s)://<server address>:<server port>/<any additional path that could have been set by application server administrator>` an HTML frontend with the FIWARE logo is properly displayed;
- check if, at the URL `http(s)://<server address>:<server port>/<any additional path that could have been set by application server administrator>/DBA?_wadl` a WADL file describing the ReST API is available.

To verify that DB Anonymizer works correctly:

- verify that visiting with a web browser URL: `<application path>/DBA/getPolicyResult?gid=<random number>` would return the following message:

Error in retrieving the requested result

This would ensure that DB connection is properly set. Any other message could represent a possible deployment issue. Return codes are described [in the API specification](#).

- launch the unit test suite as explained in [DB Anonymizer Generic Enabler - Unit Testing Plan](#); again, if misbehaviours would happen, this could mean a deployment issue; return codes are described [in the API specification](#). However, a redeployment is in general the best solution, as well as a re-creation of the DB schema.

15.3.2 List of Running Processes

- there should be a number of Java processes, depending on the Application Server configuration;
- mysql server process(es).

15.3.3 Network interfaces Up & Open

N/A

15.3.4 Databases

- check whether MySQL instance is up and running, and reachable from the Application Server;
- check whether the required MySQL database schema "census" is created, and the username supplied has granted all rights on it;
- check whether the required MySQL table is properly created into "census" with the query:

```
SELECT * FROM census.results;
```

It should return no results if the query is executed before the first DB Anonymizer execution.

15.4 Diagnosis Procedures

The Diagnosis Procedures are the first steps that a System Administrator will take to locate the source of an error in a GE. Once the nature of the error is identified with these tests, the system admin will very often have to resort to more concrete and specific testing to pinpoint the exact point of error and a possible solution. Such specific testing is out of the scope of this section.

The following sections have to be filled in with the information or an "N/A" ("Not Applicable") where needed. Do not delete section titles in any case.

15.4.1 Resource availability

Resource load of DB Anonymizer strongly depends on the size of incoming DB dump, and from the number of concurrent requests received. There are different aspects to consider, on the Application Server and on the MySQL DB.

- with respect to the Application Server:
 - DB Dumps can potentially contain significant amount of data. During the testing phase, dumps of GBs have been transferred and processed. Despite DB dumps are expected to be sent in zipped format, they are nevertheless uncompressed on the local Application Server storage, in order to be pre-parsed and then sent to the MySQL DB. Even if these dumps are deleted just after their analysis, it is recommended that several GBs of space are available, in order to sustain a significant number of concurrent requests.
- with respect to the MySQL DB:
 - the work load on the MySQL DB can be significant, therefore it is recommended to have a number of GBs of RAM and storage dedicated to MySQL, for the same reasons expressed for the Application Server. A number of CPU cores could be assigned to MySQL DBMS as well.

15.4.2 Remote Service Access

N/A

15.4.3 Resource consumption

Resource consumption strongly depends on inputs. In our tests, a 8-core i5 CPU system with 16 GB RAM and 500 GB storage was able to deal with 4 concurrent requests, with MySQL dumps of ~3 GB (uncompressed). These parameters did not affected dramatically server's performance, therefore they should considered safe values for a standard/heavy-load deployment.

15.4.4 I/O flows

The only expected I/O flow is of type HTTPS (or HTTP), on standard ports, and inbound only. Requests interactivity should be low, even if a polling mechanism on an API method (getPolicyResult) could involve several requests to be executed from clients.

16 Malware_Detection_Service _Installation_and_Administration_Guide

You can find the content of this chapter as well in the [wiki](#) of fi-ware.

16.1 Preliminary remark

Morphus System, the Malware Detection Service Generic Enabler GE, will be offered as a service by the GE owner. Besides, the software release of the GE and the accompanying Installation and Administration guide are of PP dissemination level.

In accordance with the privacy restrictions requested by GE owner, both the software (binaries) and the Installation and Administration Guides are not subject to review by the FI-WARE coordinator or the Work Package leader and the responsibility for the quality and appropriateness of this part of the deliverables remain solely with the partner who is the GE owner.

The software (binaries) and the Installation and Administration Guides are accessible to the EC for reviewing purposes. It will be disclaimed upon explicit request to the project coordinator. The partner who is the GE owner will govern this access.

16.2 Resource availability

You must referred to ["User and Programmers Guide"](#) for implementing consumer.

17 Context-based security & compliance - Installation and Administration Guide

You can find the content of this chapter as well in the [wiki](#) of fi-ware.

17.1 Introduction

The PRRS is the Atos implementation of the Context-based Security & Compliance (CBS&C) GE which implements a security layer that adds context-aware capabilities to support additional security requirements through the optional security enablers developed in FI-WARE. This GE helps user to include the appropriate solution for his concrete context and manage those security solutions with a user interface that makes easy the installation and monitors potential problems.

17.2 Installation

17.2.1 Requirements

17.2.1.1 **Operating Systems**

The current version only supports MS Window O.S. Later versions will support GNU/Linux too.

17.2.1.2 **Java VM**

The PRRS needs a JRE to run, it has been tested in a JRE 1.7, the compatibility with previous versions is not proved yet. It needs at least version 1.5.

17.2.1.3 **Database**

This GE is using MySQL database version 5.5.8, the compatibility with previous versions has not been tested.

17.2.2 PRRS Installation Steps

The PRRS has been developed using OSGi technology, so the bundles that implements the G.E. needs to be included in an OSGi container. We are using Apache Karaf as a lightweight OSGi container to provide a standalone distribution of this Context-based security & compliance GE.

Consequently, PRRS is released as a zip file including all the required dependencies and folders structure to be easily installed and run within an Apache Karaf container in Windows.

1. Once we have unzip the file **PRRS-Karaf-2.2.0.zip**, we will have all the required files for Karaf and PRRS in a directory (e.g. D:\PRRS_v1.0.0). From now on we will refer to this root directory like **{PRRS_Karaf}**.
2. Database installation.

Two databases are used in the PRRS Context-based Security & Compliance GE:

- **Context Manager:** It records the data related to the context such as the PRRS configuration, active services, event history, etc. It will be used by the PRRS to select the most suitable security solution for a received request
- **Security Library:** It is a local repository for available security services. It is updated with the information recovered from the FI-WARE Marketplace GE and it is used as a cache for a faster and more efficient search of services.

In order to create those databases and the required tables for the PRRS, the following scripts (located under the folder **{PRRS_Karaf}/PRRS_installation/databases**) must be executed in the MySQL Server:

```
mysql> source ContextManager.sql
mysql> source SecurityRepository.sql
```

In the following screenshots it is shown the schemas created:



Tabla	Acción	Filas	Tipo	Cotejamiento	Tamaño	Residuo a depurar
activepatternslist	Examinar Estructura Buscar Insertar Vaciar Eliminar	0	InnoDB	latin1_swedish_ci	16 KB	-
configuration	Examinar Estructura Buscar Insertar Vaciar Eliminar	1	InnoDB	latin1_swedish_ci	16 KB	-
eventshistory	Examinar Estructura Buscar Insertar Vaciar Eliminar	0	InnoDB	latin1_swedish_ci	16 KB	-
execcomponentsmonitoringservices	Examinar Estructura Buscar Insertar Vaciar Eliminar	0	InnoDB	latin1_swedish_ci	40 KB	-
monitoringservices	Examinar Estructura Buscar Insertar Vaciar Eliminar	1	InnoDB	latin1_swedish_ci	16 KB	-
parameters	Examinar Estructura Buscar Insertar Vaciar Eliminar	0	InnoDB	latin1_swedish_ci	32 KB	-
patternpreferencerules	Examinar Estructura Buscar Insertar Vaciar Eliminar	1	InnoDB	latin1_swedish_ci	16 KB	-
requests	Examinar Estructura Buscar Insertar Vaciar Eliminar	0	InnoDB	latin1_swedish_ci	16 KB	-
8 tablas	Número de filas	3	InnoDB	latin1_swedish_ci	176 KB	0 B

PRRS ContextManager Database Schema



Tabla	Acción	Filas	Tipo	Cotejamiento	Tamaño	Residuo a depurar
creator	Examinar Estructura Buscar Insertar Vaciar Eliminar	1	InnoDB	utf8_general_ci	16 KB	-
rulesgoals	Examinar Estructura Buscar Insertar Vaciar Eliminar	2	InnoDB	utf8_general_ci	32 KB	-
rulesmechanisms	Examinar Estructura Buscar Insertar Vaciar Eliminar	1	InnoDB	utf8_general_ci	32 KB	-
securitygoal	Examinar Estructura Buscar Insertar Vaciar Eliminar	14	InnoDB	utf8_general_ci	32 KB	-
securityrule	Examinar Estructura Buscar Insertar Vaciar Eliminar	29	InnoDB	utf8_general_ci	16 KB	-
securitytechnology	Examinar Estructura Buscar Insertar Vaciar Eliminar	1	InnoDB	utf8_general_ci	48 KB	-
serviceimplementation	Examinar Estructura Buscar Insertar Vaciar Eliminar	1	InnoDB	utf8_general_ci	48 KB	-
8 tablas	Número de filas	60	InnoDB	latin1_swedish_ci	240 KB	0 B

PRRS Security Repository Database Schema

Once we have the required databases, a new user for FI-WARE must be created in the MySQL server with permissions for the ContextManager and SecurityLibrary tables. For example:

```
mysql> create user 'fiware'@'localhost' identified by 'fiware';  
  
mysql> grant all privileges on ContextManager.* to  
      'fiware'@'localhost';  
  
mysql> grant all privileges on SecurityRepository.* to  
      'fiware'@'localhost';
```

The configuration files for the PRRS are located in **{PRRS_Karaf}/PRRS_installation/conf**. It is necessary to review the following ones include the right user and password to access the databases:

- *ContextDB.conf*
- *securityrepository.conf*

3. Karaf configuration

The information about how to access to those databases and that user must be also configured in the PRRS bundle configuration file with the following parameters:

```
database.user=fiware  
  
database.password=fiware  
  
database.contextdb.url=jdbc:mysql://localhost:3306/contextmanager  
  
database.secrepo.url=jdbc:mysql://localhost:3306/securityrepository
```

On the other hand, the PRRS will use the user data to access the FI-WARE Marketplace in order to recover the implementation of the optional security enablers which are available for him. Consequently, the following parameters must be configured in the PRRS bundle configuration:

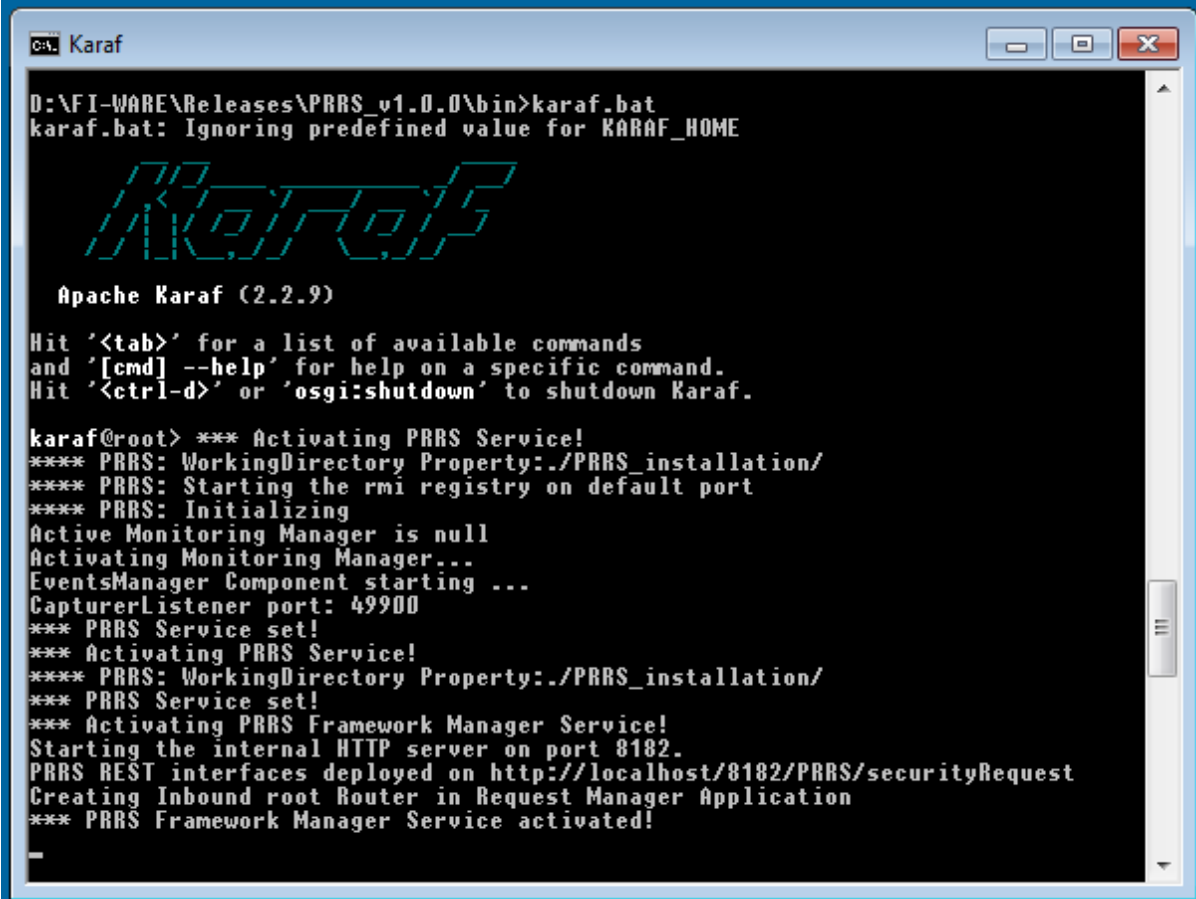
```
marketplace.url=http://appsnserv.lab.fi-ware.eu/FiwareMarketplace/v1  
  
collection.name=offering/store/security/offerings  
  
marketplace.user=integrationUser123
```

```
marketplace.passwd=integrationUser123
```

NOTE: In this example, we are assuming all the available security services are offered in the FI-WARE Marketplace in a store called *security* (*collection.name=offering/store/security/offerings*)

4. Start Karaf

To execute the PRRS it is only required to executed the following de script **karaf.bat** located in the *{PRRS_Karaf}/bin* directory. The result is the following where the last line shows that the PRRS Framework has been activated:

A screenshot of a Windows command prompt window titled "C:\> Karaf". The command "D:\FI-WARE\Releases\PRRS_v1.0.0\bin>karaf.bat" has been executed. The output shows the Karaf logo, version 2.2.9, and various initialization messages. The final message is "*** PRRS Framework Manager Service activated!".

```
C:\> Karaf
D:\FI-WARE\Releases\PRRS_v1.0.0\bin>karaf.bat
karaf.bat: Ignoring predefined value for KARAF_HOME

  Apache Karaf (2.2.9)

Hit '<tab>' for a list of available commands
and '[cmd] --help' for help on a specific command.
Hit '<ctrl-d>' or 'osgi:shutdown' to shutdown Karaf.

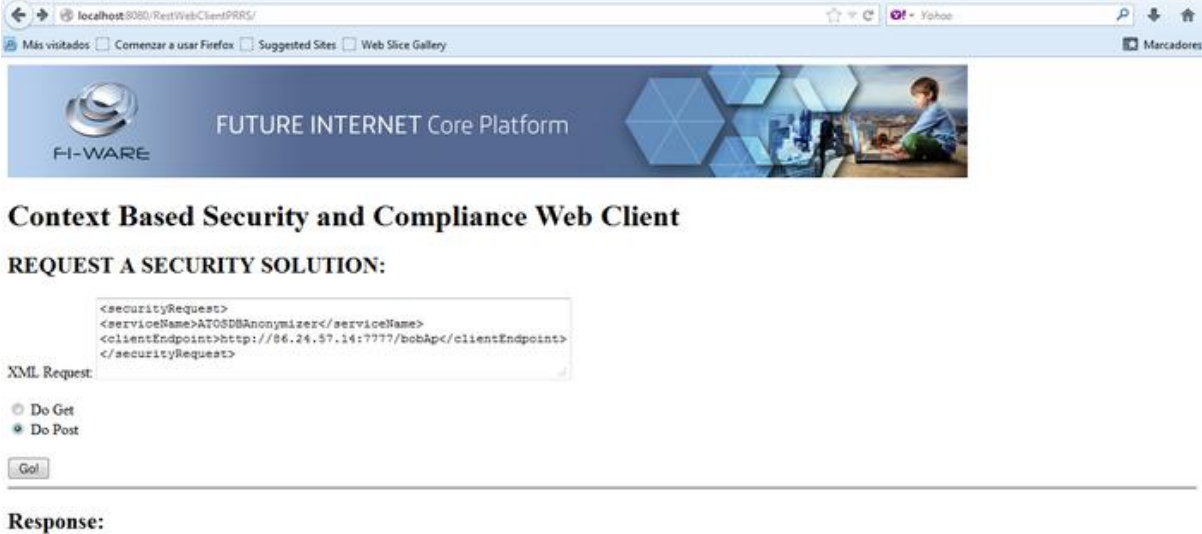
karaf@root> *** Activating PRRS Service!
**** PRRS: WorkingDirectory Property:./PRRS_installation/
**** PRRS: Starting the rmi registry on default port
**** PRRS: Initializing
Active Monitoring Manager is null
Activating Monitoring Manager...
EventManager Component starting ...
CapturerListener port: 49900
*** PRRS Service set!
*** Activating PRRS Service!
**** PRRS: WorkingDirectory Property:./PRRS_installation/
*** PRRS Service set!
*** Activating PRRS Framework Manager Service!
Starting the internal HTTP server on port 8182.
PRRS REST interfaces deployed on http://localhost:8182/PRRS/securityRequest
Creating Inbound root Router in Request Manager Application
*** PRRS Framework Manager Service activated!
```

Running PRRS with Karaf

5. Install the PRRS Client Dashboard

The PRRS Client Dashboard to send security requests fo the PRRS Framework is provided as the **RestWebClientPRRS.war** File. Copy this war file in your Tomcat installation under the webapps. Then, it is only necessary to restart Tomcat to deploy it.

The following dashboard will be accessible in <http://localhost:8080/RestWebClientPRRS>



The screenshot shows a web browser window with the URL `localhost:8080/RestWebClientPRRS/`. The page header features the FI-WARE logo and the text "FUTURE INTERNET Core Platform". Below the header, the title "Context Based Security and Compliance Web Client" is displayed. The main section is titled "REQUEST A SECURITY SOLUTION:" and contains an "XML Request" field with the following content:

```
<securityRequest>
<serviceName>ATOSDBAnonymizer</serviceName>
<clientEndpoint>http://86.24.57.14:7777/bobApp</clientEndpoint>
</securityRequest>
```

Below the XML field, there are two radio buttons: "Do Get" and "Do Post", with "Do Post" selected. A "Go!" button is located at the bottom of the form. Below the form, the "Response:" section is visible but empty.

PRRS Client Dashboard

17.3 Administration

- To start the PRRS it is only necessary to run Karaf with the following command:

```
D:\FI-WARE\PRRS_v1.0.0\bin> karaf.bat
```

- To list the bundles active in Karaf the following command must be run in the Karaf console:

```
karaf@root> osgi:ls
```

- To stop PRRS the following command, run in the Karaf console:

```
karaf@root> osgi:shutdown
```

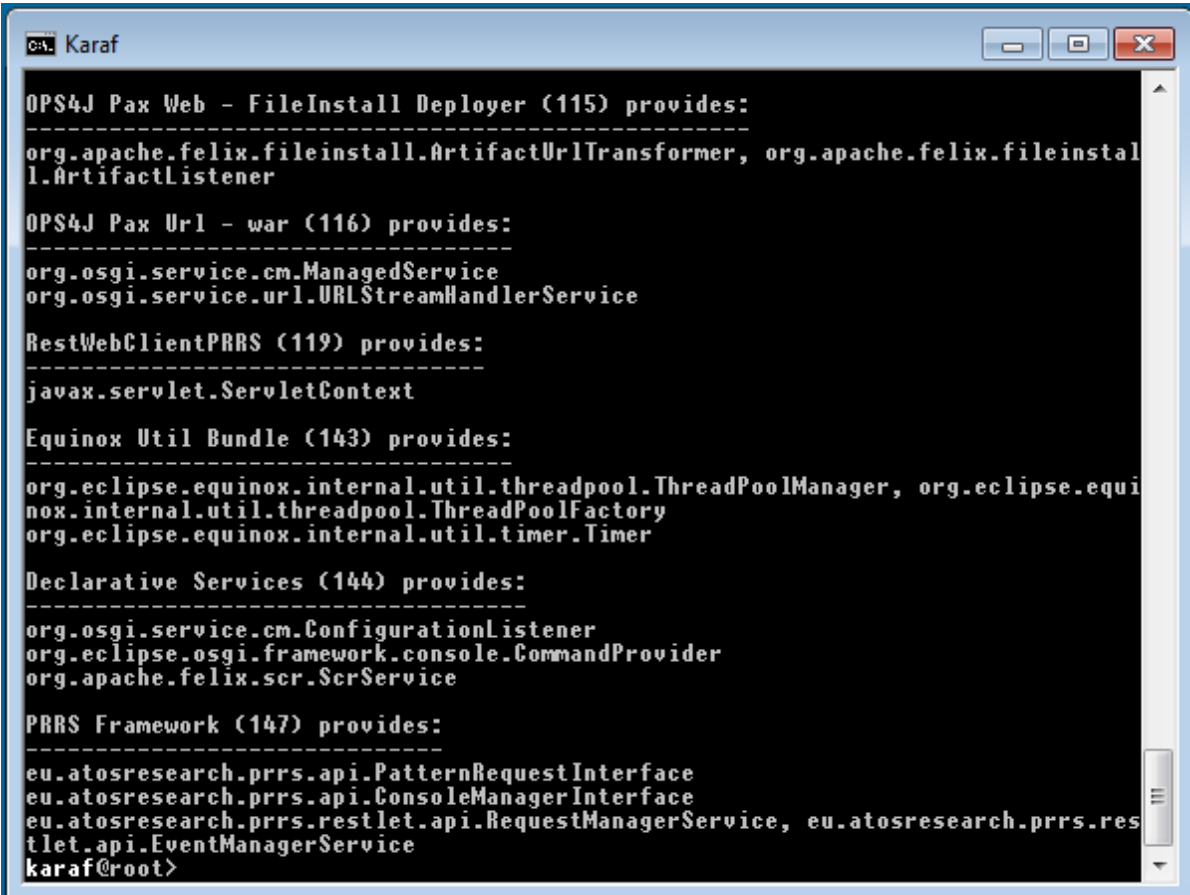
And confirm with yest to shutdown the Karaf instance.

17.4 Sanity check procedures

The Sanity Check Procedures are the steps that a System Administrator will take to verify that an installation is ready to be tested. This is therefore a preliminary set of tests to ensure that obvious or basic malfunctioning is fixed before proceeding to unit tests, integration tests and user validation.

17.4.1 End to End testing

The quick testing to check that everything is up and running can be done with the Karaf command "osgi:ls". A console like the following one should be shown:



```

Karaf

OPS4J Pax Web - FileInstall Deployer (115) provides:
-----
org.apache.felix.fileinstall.ArtifactUrlTransformer, org.apache.felix.fileinstall.ArtifactListener

OPS4J Pax Url - war (116) provides:
-----
org.osgi.service.cm.ManagedService
org.osgi.service.url.URLStreamHandlerService

RestWebClientPRRS (119) provides:
-----
javax.servlet.ServletContext

Equinox Util Bundle (143) provides:
-----
org.eclipse.equinox.internal.util.threadpool.ThreadPoolManager, org.eclipse.equinox.internal.util.threadpool.ThreadPoolFactory
org.eclipse.equinox.internal.util.timer.Timer

Declarative Services (144) provides:
-----
org.osgi.service.cm.ConfigurationListener
org.eclipse.osgi.framework.console.CommandProvider
org.apache.felix.scr.ScrService

PRRS Framework (147) provides:
-----
eu.atosresearch.prrs.api.PatternRequestInterface
eu.atosresearch.prrs.api.ConsoleManagerInterface
eu.atosresearch.prrs.restlet.api.RequestManagerService, eu.atosresearch.prrs.restlet.api.EventManagerService
karaf@root>

```

PRRS Bundles Running

17.4.2 List of Running Processes

The following processes should be up and running:

- Apache Tomcat (java.exe)

- Apache Karaf (java.exe)

17.4.3 Network interfaces Up & Open

The following ports should be open and in use:

- 8080/tcp - HTTP Server (Apache Tomcat)
- 5050/tcp - RMI Server
- 8182/tcp - PRRS Server (API REST to receive security requests and events)

17.4.4 Databases

The following databases must be up and running:

- ContextManager
- SecurityRepository

The connection details are included in the files *ContextDB.conf* and *securityrepository.conf* located in the directory {PRRS_Karaf}/PRRS_installation/conf.

Test query:

```
D:\FI-WARE\Releases\PRRS_v1.0.0> mysql -u fiware -p
mysql> use contextmanager;
Database changed
mysql> select count(*) from configuration;
+-----+
| count(*) |
+-----+
|          1 |
+-----+
1 row in set (0.00 sec)

mysql> use securityrepository;
```

```
Database changed
mysql> select count(*) from securitygoal;
+-----+
| count(*) |
+-----+
|      14 |
+-----+
1 row in set (0.00 sec)

mysql>
```

17.5 Diagnosis Procedures

The Diagnosis Procedures are the first steps that a System Administrator will take to locate the source of an error in a GE. Once the nature of the error is identified with these tests, the system admin will very often have to resort to more concrete and specific testing to pinpoint the exact point of error and a possible solution. Such specific testing is out of the scope of this section.

17.5.1 Resource availability

Resource load of the PRRS depends on the number of available services offered in the FI-WARE Marketplace for the end-user. The main aspects to be considered are:

- When the PRRS is started, it is synchronized with the FI-WARE Marketplace in order to recover available services and store relevant information in a local PRRS database (*SecurityRepository*) for a more efficient subsequent searches.
- When a security request is received in the PRRS, a search process is started with the information available in the local *SecurityRepository* database to select the implementation of the optional security enablers that better match the user requirements.

Consequently, the work load on the MySQL DB can be significant and therefore it could be recommended to have a number of GBs of RAM and storage dedicated to MySQL. But currently, there are very few available services for testing purpose so this should not be a problem.

17.5.2 Remote Service Access

The system must be connected to the network in order to receive HTTP requests. There are several ways to remotely access the system and its components:

- Web based interface
- MySQL database

17.5.3 Resource consumption

Resource consumption depends on the number of available services for the user and the work load on the MySQL Database. The more available services, the more resource consumption.

17.5.4 I/O flows

I/O flow on HTTPS (or HTTP) standard ports. Requests interactivity should be low because they represents only additional security requirements and only the url where the selected services is deployed is returned. This flow must be ensured in case the requests sent by the end-user come from a different host. There is also this I/O flow between the FI-WARE Marketplace GE and the PRRS to search the available security services.

There could be output to 514/udp port (Syslog) in case the PRRS is configured to send logs to the Syslog Server installed in the host where the FI-WARE Security Monitoring GE is running. That flow must be ensured between the host with the PRRS and the host with the FI-WARE Security Monitoring GE.

17.6 References

- Apache Karaf: <http://karaf.apache.org/>

18 Android Flow Monitoring - Installation and Administration Guide

You can find the content of this chapter as well in the [wiki](#) of fi-ware.

18.1 Introduction

Flowoid, the reference implementation of the Android Flow Monitoring GE, is an Android application written in Java, supported by a native linux daemon that handles capturing the network traffic, and monitoring the flows. This document will go through the necessary steps in order to install and configure Flowoid.

18.2 Requirements

To be able to install Flowoid to an Android device, there are a few requirements, both on the device itself, and on the host.

As this GE only contains the exporter part of a NetFlow architecture, a collector is also required, in order to collect, store, and analyse the flow records.

18.2.1 Host Computer

- The [Android SDK Tools](#) must be installed on the host, and its Android SDK Platform-tools package must be installed as well, in order to use the Android Debug Bridge tool (ADB – part of the Platform-tools package). Although this isn't absolutely required, as there are alternative ways to go through the installation, it is the tool that will be used in this document.
- The target device must be connected to the host, in a way that it is accessible through the Android Debug Bridge tool. This is usually done using the USB cable bundled with the device by its vendor.

18.2.2 Target Device

- To be able to run the application, the device must run Android 2.2 (API 8) or above.
- To be accessible through the Android Debug Bridge tool, the "USB Debugging" setting must be enabled. This option can be found in the "Developer Options", near the bottom of the device's "Settings".
- To be able to run the native capture daemon, the device must use the ARM architecture.
- To be able to start the actual packet capture, root access must be available on the device, through the use of the "su" command.

18.2.3 NetFlow Collector

A NetFlow collector, supporting NetFlow v9, has to be installed or made available somewhere, and bound to a IP address and port number that can be reached by the Android devices running the GE.

18.3 Installation

To install the application on the device, once all the prerequisites have been properly addressed, one can simply type, in a shell:

```
adb install flowoid.apk
```

The application is then uploaded to the device and installed, using the ADB tool.

18.4 Administration

This section will describe the minimum required steps to configure and launch the Android Flow Monitoring tool, as detailed information on how to use the application is provided in the [Android Flow Monitoring - User and Programmers Guide R2](#).

To get the NetFlow exporter up and running, one has to go through these steps:

- Open the "flowoid" application
- Select the "Preferences" tab
- Select the "Collector address" item, and enter the IP address or hostname of the NetFlow collector in the text field
- Select the "Collector port" item, and enter the port number of the NetFlow collector in the text field
- Select the "Main tab"
- Click the button to start the probe

The native probe should now be running, and the "flowoid" application can be sent to the background, using the "Back" or "Home" buttons, or switching to another application.

18.5 Sanity check procedures

The Sanity Check Procedures are the steps that a System Administrator will take to verify that an installation is ready to be tested. This is therefore a preliminary set of tests to ensure that obvious or basic malfunctioning is fixed before proceeding to unit tests, integration tests and user validation.

18.5.1 End to End testing

Once the application has been installed, and the probe has been started, one can check that it is running properly, with the following sequence of actions:

- Open the "flowoid" application
- Select the "Information" tab
- Select the "Show probe statistics" item

A screen displaying statistics about the probe activity should appear. If the screen is blank, it means that the probe is not running.

18.5.2 List of Running Processes

The list of running processes can be obtained using the ADB tool:

```
user@host:~$ adb shell
shell@android:/ $ ps | grep flow
u0_a116    6418   160    841412 37952 ffffffff 00000000 S
madyne.flowdroid
root       6905   6900    884     412  ffffffff 00000000 S
/data/data/madyne.flowdroid/files/softflowd
shell@android:/ $ exit
user@host:~$
```

- The **madyne.flowdroid** process is the Android application
- The **/data/data/madyne.flowdroid/files/softflowd** process is the native NetFlow probe

18.5.3 Network interfaces Up & Open

To communicate, the Android application and the native NetFlow probe use the **TCP port 12345**, the probe listens on it, and the application connects to it.

18.5.4 Databases

N/A

18.6 Diagnosis Procedures

The Diagnosis Procedures are the first steps that a System Administrator will take to locate the source of an error in a GE. Once the nature of the error is identified with these tests, the system admin will very often have to resort to more concrete and specific testing to pinpoint the exact point of error and a possible solution. Such specific testing is out of the scope of this section.

18.6.1 Resource availability

- The Android part of the tool is mostly used to control and monitor the probe. It does not make intensive use of RAM or storage, and should run fine in normal system conditions.
- The native probe does not depend on storage.
- RAM usage of the native probe is proportional to the number of active flows.

18.6.2 Remote Service Access

- The probe needs to be able to send NetFlow records to the collector, using the configured IP address and port number (see [NetFlow Collector](#)).

18.6.3 Resource consumption

- The Android part of the tool should have a minimal CPU & network usage at all time.
- CPU and RAM usage of the native probe is proportional to the network activity of the device, but a rapid increase in the consumption of these two resources might indicate that the Android device is currently taking part of a DoS attack, or is the target of one.

18.6.4 I/O flows

- The only expected I/O flow will be an outgoing UDP flow to the configured collector IP address and port number.

19 Content-based Security - Installation and Administration Guide

You can find the content of this chapter as well in the [wiki](#) of fi-ware.

19.1 Introduction

The Content Based Security (CBS) Optional Generic Enabler (OGE) is composed of three Java Web Applications: the CBS Provider, CBS Consumer and CBS Broker, which are packaged in Web Archive (WAR) files. It requires a Java Application Server. Note that although any application server should work, only Apache Tomcat is supported.

19.2 Installation

19.2.1 Requirements

The CBS OGE requires the following software to be installed:

- Tomcat 7 [\[1\]](#)
- Maven [\[2\]](#)

The CBS OGE requires the following Generic Enablers:

- Access Control GE [\[3\]](#)

19.2.2 Build the CBS OGE

The CBS OGE can be downloaded from the FI-WARE PPP Restricted Project website [\[4\]](#) as a zip file. The zip file contains three Maven projects, which need to be built using Apache Maven [\[2\]](#), and a local Maven repository containing some of the dependencies needed to build the projects. Each of the projects corresponds to a component of the CBS OGE: the Producer is in `fiware.cbs.producer.service.webapp`, the Consumer is in `fiware.cbs.consumer.service.webapp` and the Broker is in `fiware.cbs.broker.service.webapp`.

Once the Maven projects have been unzipped, we need to configure each component by modifying the `application.properties` file in each project as described in [Administration](#).

After modifying the files, we need to build the project in order to generate the binaries to install. To do so, we run the following commands:

```
$ cd fiware.cbs.producer.service.webapp
```



```
$ mvn clean package

$ cd fiware.cbs.consumer.service.webapp

$ mvn clean package

$ cd fiware.cbs.broker.service.webapp

$ mvn clean package
```

The WAR files are in the target folder of each project.

19.2.3 Install the CBS GE

Before installing the CBS web applications, you need to configure the application server and the web applications as described in [Administration](#).

The CBS web applications are installed by copying the WAR files (fiware.cbs.provider.service.webapp.war, fiware.cbs.consumer.service.webapp.war and fiware.cbs.broker.service.webapp.war) into the "webapps" folder of Apache Tomcat. To install them on other Java Application Servers (e.g. JBoss) please refer to the specific application server guidelines.

19.3 Administration

19.3.1 Application Server Configuration

The Apache Tomcat application server as must be configured to accept HTTPS/SSL connections. This can be achieved by using a front-end HTTPS server that will proxy all requests to the CBS GE, or by configuring the Application Server in order to accept only HTTPS/SSL connection, as described in the Tomcat SSL Configuration How To Guide^[9].

19.3.2 Key Generation

The CBS Provider, the CBS Consumer and the CBS Broker each require a keystore containing a public and private key pair and a truststore containing the public key certificates of the services it trusts.

The provider's and consumer's truststores should contain the broker's public key certificate and the broker's truststore should contain the provider's and consumer's public key certificates.

The CBS GE currently operates only on JKS format keystores. The JKS format is Java's standard "Java KeyStore" format, and is the format created by the keytool command-line utility. This tool is included in the JDK.

To create a new keystore containing a single self-signed Certificate and key pair, execute the following from a terminal command line:

```
$ keytool -genkey -alias <alias> -dname <x500 distinguished name> -  
keyalg RSA -keysize <keysize> -validity <certificate validity in  
days> -keystore \path\to\my\keystore
```

Note that the current implementation only allows a key size of up to 512 bits.

To export the self-signed certificate from the keystore, execute the following from a terminal command line:

```
keytool -export -alias <alias> -keystore \path\to\my\keystore -rfc -  
file <file>
```

To create a new truststore from scratch, you need to create a keystore with a key and then delete the key:

```
$ keytool -genkey -alias foo -keystore \path\to\my\truststore  
  
$ keytool -delete -alias foo -keystore \path\to\my\truststore
```

To import the self-signed certificate to the trust store execute the following from a terminal command line:

```
$ keytool -import -alias <alias> -file <file> -keystore  
\path\to\my\truststore -storepass <password>
```

19.3.3 CBS Provider Configuration

Before you deploy the CBS Provider web application to your Application Server, you must configure the provider key store and trust store properties in the application.properties file.

File Location: fiware.cbs.provider.service.webapp.war/WEB-INF/classes/application.properties in the build WAR file or /fiware.cbs.provider.service.webapp/src/main/resources/application.properties in the unbuilt Maven project

```
keystore.file=path/to/keystore/file

keystore.pass=[KEYSTORE_PASSWORD]

key.alias=[ALIAS_OF_PROVIDER'S_KEY]

key.pass=[KEY_PASSWORD]

truststore.file= path/to/truststore/file

truststore.pass=[TRUSTSTORE_PASSWORD]

consumer.alias=[ALIAS_OF_BROKER'S_KEY]

encryption.algorithm.mode=[ALGORITHM_MODE]

encryption.algorithm.padding=[ALGORITHM_PADDING]

keywrap.algorithm.mode=[ALGORITHM_MODE]

keywrap.algorithm.padding=[ALGORITHM_PADDING]
```

19.3.4 CBS Consumer Configuration

Before you deploy the CBS Consumer web application to your Application Server, you must configure the consumer key store and trust store properties in the application.properties file.

File Location: fiware.cbs.consumer.service.webapp.war/WEB-INF/classes/application.properties in the build WAR file or
/fiware.cbs.consumer.service.webapp/src/main/resources/application.properties in the unbuilt Maven project

```
keystore.file=path/to/keystore/file

keystore.pass=[KEYSTORE_PASSWORD]

key.alias=[ALIAS_OF_CONSUMER'S_KEY]

key.pass=[KEY_PASSWORD]

truststore.file= path/to/truststore/file

truststore.pass=[TRUSTSTORE_PASSWORD]
```

19.3.5 CBS Broker Configuration

Before you deploy the CBS Broker web application to your Application Server, you must configure the broker key store and trust store properties in the application.properties file.

File Location: fiware.cbs.broker.service.webapp.war/WEB-INF/classes/application.properties in the build WAR file or /fiware.cbs.broker.service.webapp/src/main/resources/application.properties in the unbuilt Maven project

```
keystore.file=path/to/keystore/file

keystore.pass=[KEYSTORE_PASSWORD]
```

```
key.alias=[ALIAS_OF_CONSUMER'S_KEY]

key.pass=[KEY_PASSWORD]

truststore.file= path/to/truststore/file

truststore.pass=[TRUSTSTORE_PASSWORD]

authserver.url=[ACCESS_CONTROL_GE_URL]
```

19.4 Sanity check procedures

The Sanity Check Procedures are the steps that a System Administrator will take to verify that an installation is ready to be tested. This is therefore a preliminary set of tests to ensure that obvious or basic malfunctioning is fixed before proceeding to unit tests, integration tests and user validation.

19.4.1 End to End testing

The first step is to verify that all three web applications are running:

- Navigate to the Tomcat Manager web page in a browser: <https://<host name>:<Tomcat SSL port>/manager/html/list> or use the text interface: <https://<host name>:<Tomcat SSL port>/manager/text/list>
- Verify that the following web applications are running:
 - fiware.cbs.producer.service.webapp
 - fiware.cbs.consumer.service.webapp
 - fiware.cbs.broker.service.webapp

If any of the web applications are not running, check the log files in `[TOMCAT_HOME]/logs` for errors.

Once you have verified that the web applications are running, perform the unit tests [\[6\]](#) to verify that the CBS OGE is able to protect and remove protection from data.

19.4.2 List of Running Processes

- Java.exe

19.4.3 Network interfaces Up & Open

Port	Protocol	Use
8080	TCP	Tomcat HTTP connector
8443	TCP	Tomcat HTTPS connector

Note that both of these ports can be changed in the Tomcat configuration files.

19.4.4 Databases

N/A

19.5 Diagnosis Procedures

The Diagnosis Procedures are the first steps that a System Administrator will take to locate the source of an error in a GE. Once the nature of the error is identified with these tests, the system administrator will very often have to resort to more concrete and specific testing to pinpoint the exact point of error and a possible solution. Such specific testing is out of the scope of this section.

19.5.1 Resource availability

The resource load of the CBS OGE strongly depends on the number of concurrent requests received as well as on the free main memory and disk space. The minimum requirements are:

- Minimum available memory: 256 MB
- Minimum available hard disk space: 256 MB

19.5.2 Remote Service Access

N/A

19.5.3 Resource consumption

Resource consumption strongly depends on the load, especially on the number of concurrent requests. The memory consumption of the Tomcat application server should be between 48MB and 1024MB. These numbers can vary significantly if you use a different application server.

19.5.4 I/O flows

The only expected I/O flow is of type HTTP or HTTPS, on the ports defined in the Apache Tomcat configuration files, inbound and outbound. Requests interactivity should be low.

19.6 References

1. [↑ http://tomcat.apache.org/tomcat-7.0-doc/setup.html](http://tomcat.apache.org/tomcat-7.0-doc/setup.html)
2. [↑ ^{2.0} ^{2.1} http://maven.apache.org/](http://maven.apache.org/)
3. [↑ https://forge.fi-ware.eu/plugins/mediawiki/wiki/security/index.php/FIWARE.ArchitectureDescription.Security.Access.Control.Generic.Enabler](https://forge.fi-ware.eu/plugins/mediawiki/wiki/security/index.php/FIWARE.ArchitectureDescription.Security.Access.Control.Generic.Enabler)
4. [↑ https://forge.fi-ware.eu/frs/?group_id=23](https://forge.fi-ware.eu/frs/?group_id=23)
5. [↑ http://tomcat.apache.org/tomcat-7.0-doc/ssl-howto.html](http://tomcat.apache.org/tomcat-7.0-doc/ssl-howto.html)
6. [↑ Content-based Security - Unit Testing Plan and Report](#)