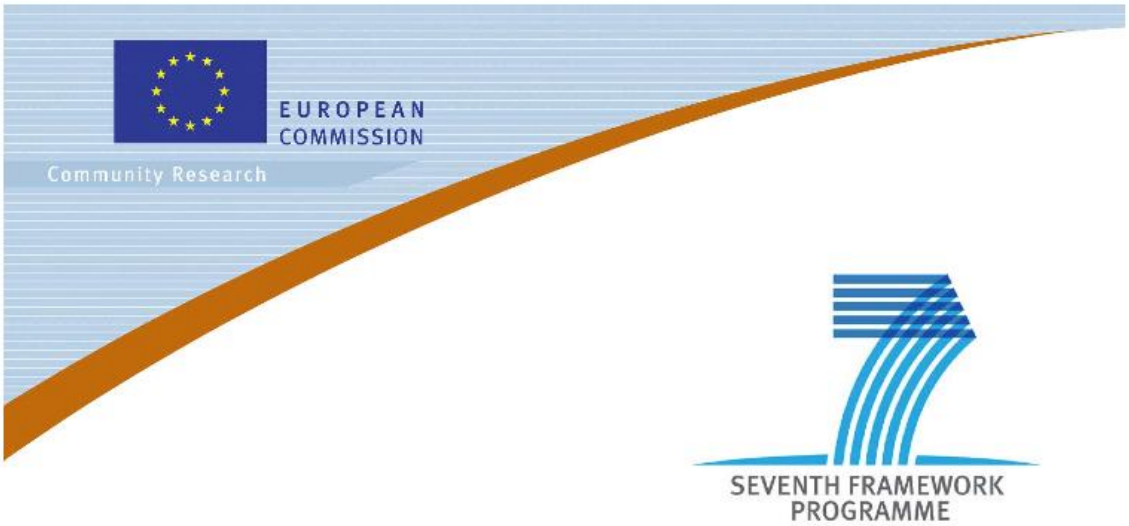


Private Public Partnership Project (PPP)

Large-scale Integrated Project (IP)

D.8.3.3: 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.8.3.3
Project Document Date: 2014-09-25
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.3), 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.

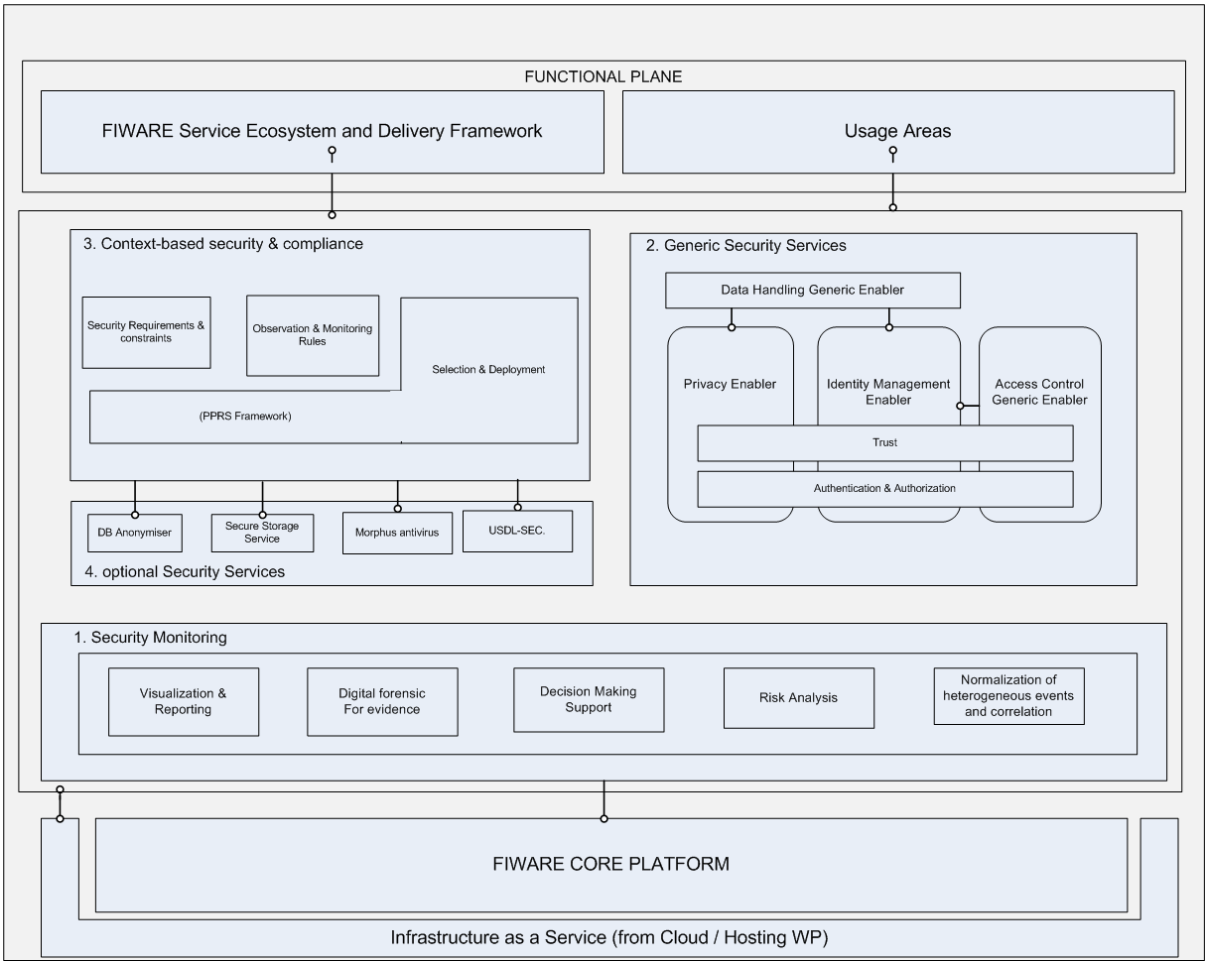
The high level architecture is formed by four main modules: Security monitoring mechanisms (M1), a set of General Core Security Mechanisms (e.g. Identity Management and Privacy solutions) (M2), Context-Based Security and Compliance (M3) where an enhanced version of USDL for security will support the 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.org

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.org/

The following resources were used to generate this document:

D.8.3.3 Installation and Administration Guide front page

Security Monitoring - Installation and Administration Guide

Security Monitoring/Service Level SIEM - 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

Security Monitoring / Remediation - Installation and Administration Guide

Security Monitoring / Visualisation Framework - Installation and Administration Guide

Identity Management - KeyRock - Installation and Administration Guide

Privacy - Installation and Administration Guide

Data Handling GE - Installation and Administration Guide

Access Control - Installation and Administration Guide

Context-based Security and Compliance - Installation and Administration Guide

DB Anonymizer GE - Installation and Administration Guide

Content-based Security - Installation and Administration Guide

Secure Storage Service - 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 our wikis. 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
v3	Version	2014-09-26	Thales

1.10 Table of Contents

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/Service Level SIEM - 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
- Security Monitoring / Remediation - Installation and Administration Guide
- Security Monitoring / Visualisation Framework - Installation and Administration Guide

3 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.

3.1 Introduction

The Service Level SIEM (SLS) provided for FI-WARE in the Security Monitoring GE integrates a preconfigured version of the Open Source OSSIM SIEM with a Storm cluster (http://storm-project.net/) to provide scalability and distributed real time computation. A set of Java processes, 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.

This guide describes the installation and administration steps that need to be followed in order to install first the two baseline tools used by the Service Level SIEM, OSSIM and Storm, and then the specific files and modifications required to run the Service Level SIEM on the Storm cluster and integrated with the OSSIM framework and databases.

We also differentiate in this guide between what need to be installed with the Security Probes (which include a SIEM Agent and a set of plugins) on the Sensors and the Service Level SIEM server installed on a centralized host to receive the events already normalized coming from the different Security Probes.

3.2 System Requirements

3.2.1.1 *Operating System and OSSIM*

OSSIM is distributed by Alienvault (http://www.alienvault.com) usually as a standalone Debian based Operating System. It must be previously installed and configured in the machine where the Service Level SIEM is going to be running. Detailed instructions about the installation can be found in the section *Service Level SIEM Installation steps* included in this guide or in the OSSIM official website (http://www.ossim.net).

OSSIM can be also installed from the source code on a standard Debian or Ubuntu linux distribution. The following kernel parameters need to be added to the file */etc/sysctl.d/ossim.conf* and applied:

```
kernel.printk = 1 4 1 7
net.ipv4.tcp_timestamps = 0
net.ipv6.conf.all.disable_ipv6 = 1
net.ipv6.conf.all.autoconf = 0
net.ipv6.conf.default.autoconf = 0
kernel.panic = 10
fs.file-max=1000000
```

```
vm.swappiness=10
```

3.2.1.2 *Java VM*

The Service Level SIEM requires Java release 1.6 or greater installed and the JAVA_HOME variable set in the user profile. In particular, the jdk1.6.0_37 has been used in the tests.

3.2.1.3 *Database*

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

3.2.1.4 *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 Cluster mode that is 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.yp.to/daemontools.html).

3.3 Installation

3.3.1.1 *OSSIM Installation*

The most important points for a clean installation are listed here:

1. Install OSSIM from a bootable DVD
OSSIM is distributed as a standalone Debian based Operating System. The version used for the Service Level SIEM is v4.1.0.

You can download and burn the ISO from here: http://www.alienvault.com/free-downloads-services

Boot from the DVD to install the system. OSSIM requires a new partition in a physical or a virtual machine.

The wizard will require some parameters to configure the OSSIM (such as the IP, password, database, sensors available, etc).

These parameters can be changed at any time after the installation is completed, and also the wizard provides an automated installation that will configure the OSSIM with the default parameters with almost no user intervention. An automated installation is enough for a beginner user, so you can safely go with this option.

2. Install OSSIM from the source code

Although it is not the easiest way, if you want to install OSSIM on an existent standard Operating System such as Ubuntu or Debian, you can download the source code from Git and checkout the release 4.1.0:

```
#apt-get install -y git

#git clone https://git@git.assembla.com/os-sim.2.git

#cd os-sim.2

#git checkout tags/release-4.1.0
```

Then follow the instructions included in the README file to compile and install the code. It is important to remark that you will need to install previously all the OSSIM dependencies:

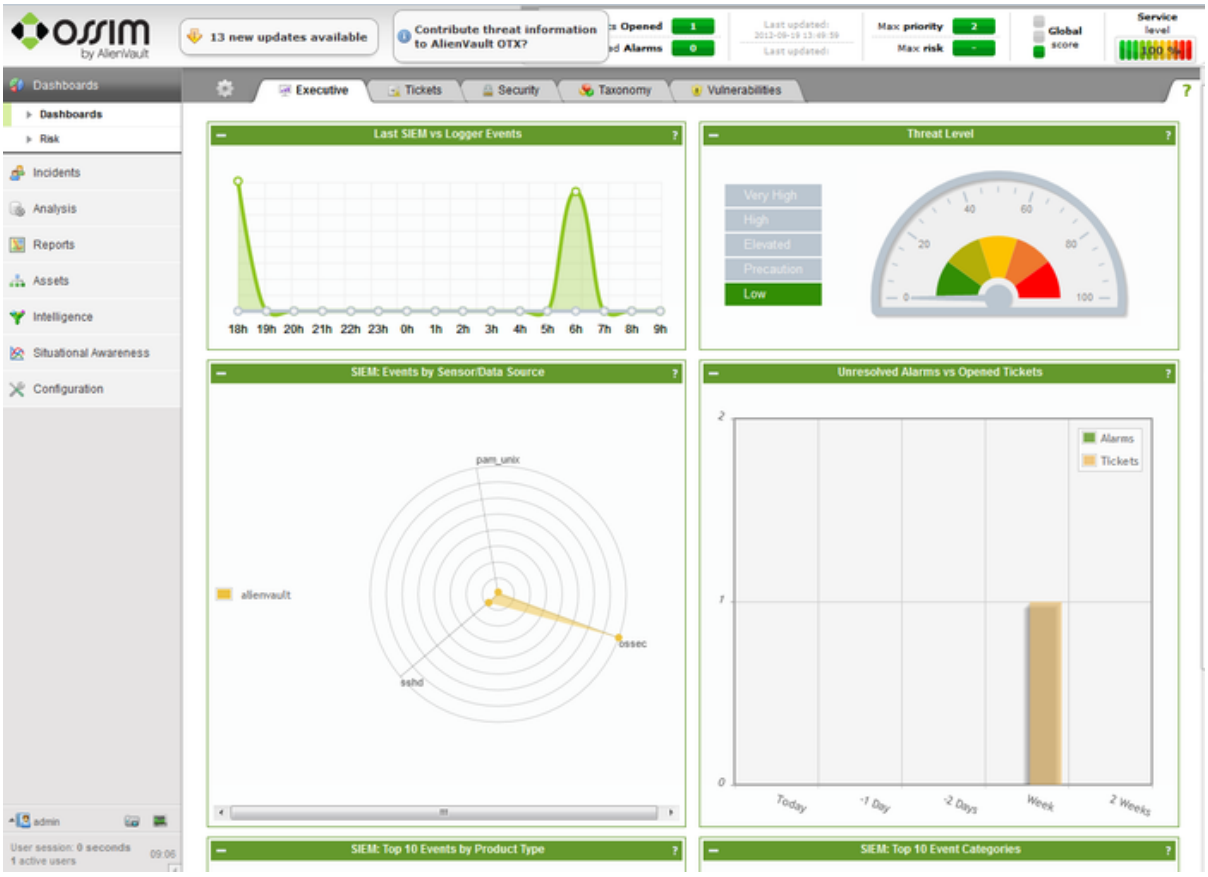
```
# cd os-sim

# apt-get update && apt-get upgrade

# apt-get install -y automake libtool autoconf intltool e2fsprogs
hdparm unzip

# apt-get install -y libglib2.0-dev gnet2.0 libgnet2.0 libgnet-dev

# apt-get install -y libxml2 libxml2-dev libxslt1-dev openssl
libssl-dev

# apt-get install -y uuid uuid-dev json-glib-1.0 libjson-glib-1.0
libjson-glib-dev

# apt-get install -y python-dev php5-geoip geoip-bin geoip-database
libgeoip1 libgeoip-dev python-geoip

# apt-get install geoclue geoclue-hostip geoclue-localnet geoclue-
manual geoclue-yahoo libgeoclue0

# apt-get install fprobe fprobe-ng

# apt-get install mysql-client-5.5 mysql-server-5.5 mysql-common
libmysqlclient18

# apt-get install libgda-5.0-mysql php5-mysql python-mysqldb snort-
mysql libxml-simple-perl

# apt-get install libdbi-perl libdbd-mysql-perl libapache-dbi-perl
libnet-ip-perl libsoap-lite-perl

# apt-get install python-dev geoip-bin geoip-database

# wget http://geolite.maxmind.com/download/geoip/api/c/GeoIP-
1.4.7.tar.gz

# tar -xvzf GeoIP-1.4.7.tar.gz

# cd GeoIP-1.4.7
```

```
# ./configure --prefix=/usr

# make

# make install

# cd ..

# wget http://geolite.maxmind.com/download/geoip/api/python/GeoIP-
Python-

1.2.7.tar.gz

# tar -zxvf GeoIP-Python-1.2.7.tar.gz

# cd GeoIP-Python-1.2.7

# python setup.py build

# python setup.py install

# cd ..

# wget
http://geolite.maxmind.com/download/geoip/database/GeoLiteCity.dat.g
z

  -O /tmp/GeoLiteCity.dat.gz

# mkdir –p /usr/share/geoip

# gunzip -c /tmp/GeoLiteCity.dat.gz >
/usr/share/geoip/GeoLiteCity.dat

# apt-get install python-pyinotify python-setuptools

# easy_install pytz

# apt-get install python-nmap nmap python-ldap python-libpcap
python-adodb

# apt-get install openjdk-6-jdk

# wget
http://ftp.gnome.org/mirror/gnome.org/sources/libgda/4.2/libgda-
4.2.13.tar.xz

# mv libgda-4.2.13.tar.xz libgda-4.2.13.tar

# tar –xvf libgda-4.2.13.tar

# cd libgda-4.2.13

# ./configure –enable-mysql=yes –enable-postgres=yes –prefix=/usr

# make

# make install

# cd ..

#./autogen.sh
```

```
#./configure --exec-prefix=/usr --datarootdir=/usr/share --
sysconfdir=/etc

   --includedir=/usr/share/ossim/include

#make

#make install
```

3. OSSIM management interface

This is a web based interface available only if the OSSIM was configured with the profile **Framework** in the installation (*the automated installation includes all the profiles by default*) or later in the configuration file */etc/ossim/ossim_setup.conf*.



OSSIM management screen

4. Fine tuning

The configuration parameters are stored in the file **/etc/ossim/ossim_setup.conf**, including the Database parameters/password.

The OSSIM configuration file (**/etc/ossim/ossim_setup.conf**) is separated in different sections, and each one is marked with a title in brackets, except for the first one, which has no title.

To mention the most relevant ones:

- **First section**:

Main configuration parameters, including the IP, hostname, mail server and profile of the OSSIM server.

- **[*database*]**

Database IP, port, db names, username, password.

- **[*framework*]**

Information regarding the framework IP/port.

- **[*sensor*]**

Active OSSIM sensors are listed here with the networks monitored, network interfaces listening, etc.

- **[*server*]**

Active OSSIM plugins are listed here, which are in charge of translating detected events to OSSIM event format.

If you want to change the server configuration, use the command:

```
# ossim-setup
```

This launches a configuration wizard.

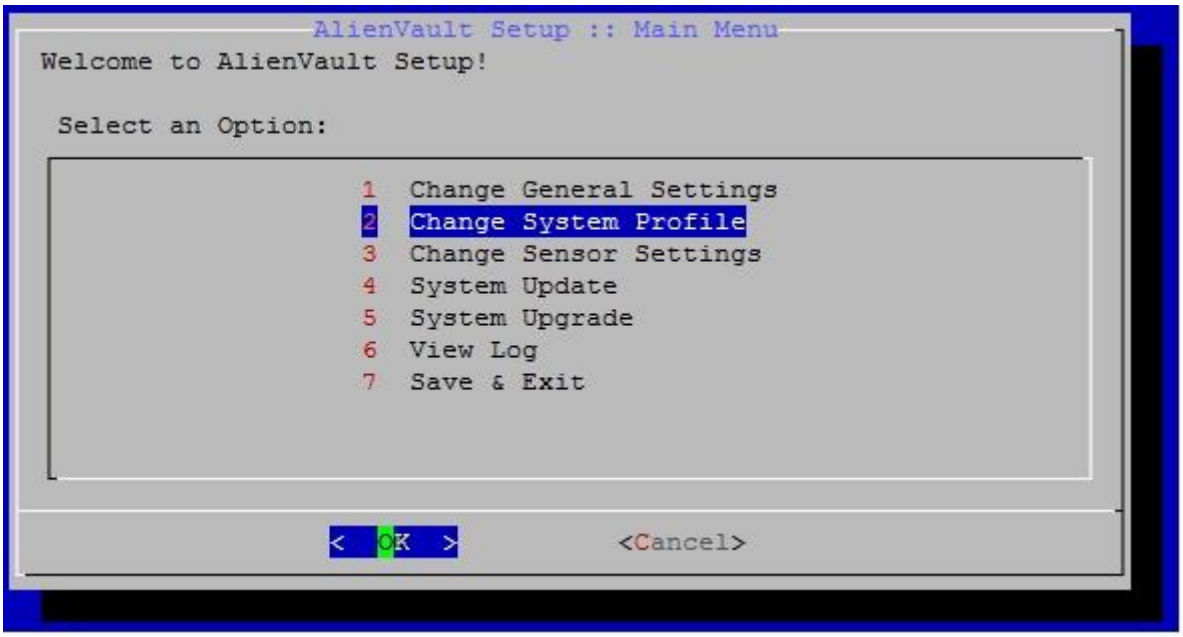
Users can update its keyboard and local layout (from default English – UK) by using the following OSSIM commands:

```
dpkg-reconfigure console-data
dpkg-reconfigure tzdata
```

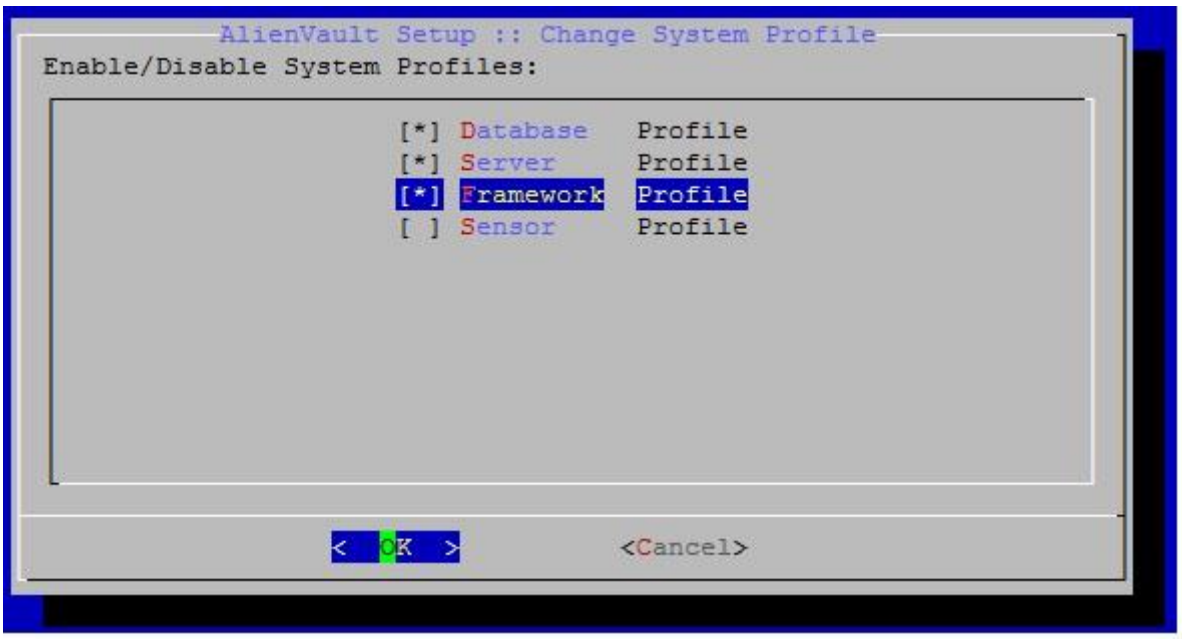
Both, server and visualization framework components and collector component can be configured by using the following OSSIM command:

```
# ossim-setup
```

- Under the Change System Profile settings option users are able to update server and local database configuration:

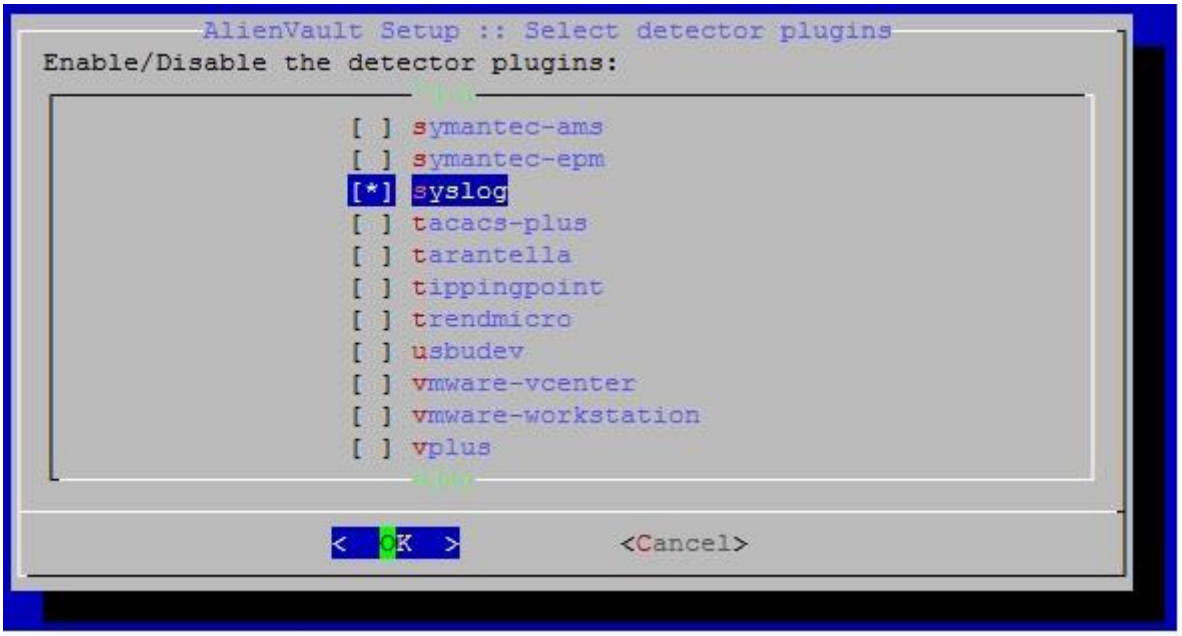


OSSIM Configuration screen



OSSIM Configuration screen

- By using Change Sensor Settings option users can enable or disable OSSIM detector & collector plug-ins. By default, only the OSSIM syslog & snare collector agent has been enabled in the component installed.



OSSIM Configuration screen

- Finally, the path where the log for OSSIM syslog & snare collector agent is being stored can be configured by editing the parameter location in the agent .cfg file under /etc/ossim/agent/plugins directory. If this is the case, the user should also restart the collector with the following command:

```
/etc/init.d/ossim-agent restart
```

3.3.1.2 *Storm Cluster Installation*

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

1. Definition of nodes 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 tasks to machines and monitoring for failures.
- **Worker nodes**: Each worker node runs a daemon called *Supervisor* that listens for tasks assigned to its machine and starts and stops as many worker processes as necessary based on what the Nimbus daemon has assigned to it.

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 for 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 nodes. 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 is 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. /usr/local/zookeeper-3.4.5) and create a link from */opt/zookeeper* to that location.
- Create the configuration file */opt/zookeeper/conf/zoo.cfg* with the name of your ZooKeeper servers. For example:

```
tickTime=2000
dataDir=/var/log/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/log/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:

```
# cd /opt/zookeeper; ./bin/zkServer.sh start
```

- Test your deployment by connecting to the hosts:

```
# cd /opt/zookeeper; ./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. For example, the following line could be added to the cron:

```
/opt/zookeeper/bin/zkCleanup.sh /var/log/zookeeper -n 7
```

And since ZooKeeper is fail-fast, it will exit the process if it encounters any error case. Consequently, in a production installation 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 pkg-config
# 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 the Storm software

Download and extract the Storm software (v0.9.0) into Master and Worker machines, create a link from */opt/storm* to the folder where it has been unzipped and add the */opt/storm/bin* directory to your PATH variable in the *~/.bashrc* file.

```
export STORM_HOME=/opr/storm
export PATH=$PATH:$STORM_HOME/bin
```

5. Configure the Storm Cluster

The file */opt/storm/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 the list of the hosts in the Zookeeper cluster for your Storm cluster. It should look something like:

```
storm.zookeeper.servers:
   - "130.206.81.165"
```

NOTE: If the port that your Zookeeper cluster uses is different than the one by 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, confs, and things like that). You should create that directory on each machine, give it proper permissions, and then fill in here the directory location. 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 parameter.
- *nimbus.host*: The worker nodes need to know which machine is the master in order to download topology jars and confs. 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 six ports here, then Storm will be able to allocate up to six workers to run on this machine. By default, this setting is configured to run up to 6 workers on the ports 6700-6705. For example:

```
supervisor.slots.ports:
    - 6700
    - 6701
    - 6702
    - 6703
    - 6704
    - 6705
```

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 processes on your local machine. In remote mode, you submit topologies for execution on a cluster of machines. To be able of starting and stopping 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"
```

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

3.3.1.3 *Security Probe Installation*

In case you have a distributed system, it is not necessary to install the complete Service Level SIEM component on each Sensor but only the Security Probe which is composed by:

- **SIEM Agent** responsible for collecting all the data generated by the different data sources and send it to the server in a standardized way. Usually, one agent per machine is enough, but several could be installed if necessary, each of them collecting from different data sources and sending to different servers. This agent is a modified version of the **ossim-agent** included in the standard distribution of OSSIM.
- **Plugins** in charge of reading from the logs generated by the data sources considered in each sensor monitored and standardize them so that the Agent can send them to the server. A set of these plugins is provided by default including the most common data sources, such as syslog, snare, cisco, apache, snort, nagios, nessus, etc. More information about the creation of new plugins can be found in the Security Monitoring - Installation and Administration Guide.

3.3.1.3.1 *Prerequisites*

We are assuming that the Security Probes are deployed on the nodes with Ubuntu Server 12.04 LTS operating system installed. The following commands must be executed to ensure all the dependencies are installed:

```
# sudo apt-get install python-dev geoip-bin geoip-database

# wget http://geolite.maxmind.com/download/geoip/api/c/GeoIP-
1.4.7.tar.gz

# tar -xvzf GeoIP-1.4.7.tar.gz

# cd GeoIP-1.4.7

# sudo apt-get install gcc zlib1g-dev make

# ./configure --prefix=/usr

# make

# sudo make install

# cd ..

# wget http://geolite.maxmind.com/download/geoip/api/python/GeoIP-
Python-

1.2.7.tar.gz

# tar -zxvf GeoIP-Python-1.2.7.tar.gz

# cd GeoIP-Python-1.2.7

# python setup.py build

# sudo python setup.py install

# cd ..
```

```
# wget
http://geolite.maxmind.com/download/geoip/database/GeoLiteCity.dat.g
z

# sudo mkdir –p /usr/share/geoip

# sudo gunzip -c GeoLiteCity.dat.gz >
/usr/share/geoip/GeoLiteCity.dat

# sudo apt-get install python-pyinotify python-setuptools

# sudo easy_install pytz

# sudo apt-get install python-nmap nmap python-ldap python-libpcap
python-adodb openjdk-6-jdk

# sudo mkdir -p /var/log/ossim

# sudo chmod 755 /etc/init.d/ossim-agent
```

3.3.1.3.2 *Installation steps*

To install the Security Probe in a host it is necessary:

1. Install the agent and plugins:

Unzip the file **SecurityProbe.tar.gz**, which includes the source code, provided with the distribution of the Service Level SIEM (file **ServiceLevelSIEM-3.3.3.tar.gz**) and execute:

```
# sudo python setup.py install --prefix=/usr
```

NOTE: The option *--help-commands* can be used to get all the available options.

2. Configure the Agent:

Once installed, the file /etc/ossim/agent/config.cfg must be edited to:

- check the **[control-framework]** section to define the ID for the agent (to be identified from the Framework the source of the events) and the ip where the ossim control framework is running.

```
enable=True

id=<id>

ip=<server_ip>

port=40003
```

- check the **[output-server]** section: it enables to send the events to the OSSIM Server core engine included in the Service Level SIEM:

```
enable=True

ip=<server_ip>
```

```
port=40001
send_events=True
```

- check the **[output-storm]** section: it enables to send the events to the Storm cluster installed with the Service Level SIEM for their processing in the high-performance correlation engine:

```
enable=True
ip=<server_ip>
port=41000
send_events=True
```

- check the **[plugins]** section: it defines the plugins to be considered by the agent. It is necessary to include a line by each plugin with the format <plugin_name>=<plugin:location>.

```
nmap-monitor=/etc/ossim/agent/plugins/nmap-monitor.cfg
ntop-monitor=/etc/ossim/agent/plugins/ntop-monitor.cfg
ping-monitor=/etc/ossim/agent/plugins/ping-monitor.cfg
snare=/etc/ossim/agent/plugins/snare.cfg
whois-monitor=/etc/ossim/agent/plugins/whois-monitor.cfg
wmi-monitor=/etc/ossim/agent/plugins/wmi-monitor.cfg
fiware=/etc/ossim/agent/plugins/fiware.cfg
cisco-asa=/etc/ossim/agent/plugins/cisco-asa.cfg
snort_syslog=/etc/ossim/agent/plugins/snort_syslog.cfg
```

3. Configure the data sources and plugins:

Each data source to be monitored requires a plugin to collect and normalize its logs. These plugins are installed under the folder */etc/ossim/agent/plugins*. So, if for example a data source generates the logs in a location different from the default value, the file <plugin>.cfg associated to it should be edited and the parameter *location* changed with the right location of the log. The activation/deactivation of the plugins is done through the section [plugins] included in the /etc/ossim/agent/config.cfg file.

Some of the data sources that could be installed with the Security Probes in order to generate information relevant for the security monitoring of a node are:

- pads - Passive Asset Detection System
- tcptrack - Monitor TCP connections on the network
- snort - Open source network intrusion detection system
- openVAS-Client - the client part of the OpenVAS Security Scanner
- nagios3 - Network/systems status monitoring daemon

- ntop – Network usage monitor
- arpwatch - Monitor ethernet/ip address pairings
- p0f - Identify remote systems passively

They can be installed with this command:

```
sudo apt-get install snort rsyslog nagios3 tcptrack ntop pads
arpwatch p0f
```

4. Start the Agent:

The agent is installed as a service in the host so you can execute:

```
# sudo service ossim-agent start
```

5. Add the ossim agent to the services in the machine so it can be automatically started:

```
sudo update-rc.d ossim-agent start 20 3 4 5 .
```

3.3.1.4 *Service Level SIEM Installation*

All the required files in order to add the Service Level SIEM functionalities to the OSSIM core engine and the topology to the Storm Cluster are included in the **ServiceLevelSIEM-3.3.3.tar.gz** file which can be downloaded from https://forge.fi-ware.org/frs/?group_id=7. It includes the following folders:

- admin: it includes the scripts to administrate the ServiceLevelSIEM topology
- conf: it includes the configuration files for the Service Level SIEM topology
- db: it includes the MySQL scripts to create and populate the required SIEM databases. The script *sls_EPLdb.sql* can be used to recreate only the specific tables for the Service Level SIEM topology whereas the script *ServiceLevelSIEM.sql* includes a dbdump of all the necessary databases
- ossim-sls: it includes the files required to adapt OSSIM to the protocol SSL
- securityprobe: it includes the files required to install the SIEM agent and plugins in a host
- test: it includes the scripts used for testing the Service Level SIEM features
- topologies: it includes the jar file with the Service Level SIEM topology to be run in the Storm cluster
- www: it includes the Service Level SIEM dashboard integrating the standard OSSIM dashboard with the modifications done to adapt it to the Service Level SIEM

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 file **ServiceLevelSIEM-3.3.3.tar.gz**. It must be unzipped and the following script executed:

```
#./installSLS_Storm.sh
```

This script makes use of the variable *$STORM_HOME* with the Storm installation folder (*/opt/storm*) and in case this variable is not configured, an error will be shown. The Jar file with the topology to be started in the Storm cluster for the Service Level SIEM will be copied into the folder *$STORM_HOME/topologies* and the folders *conf* and *admin* copied into the Storm installation folder.

The scripts for the administration of the zookeeper and storm processes can be add to the services in the machine so it can be automatically started:

```
sudo update-rc.d zookeeper start 10 2 3 4 5 . stop 10 0 1 6 .
sudo update-rc.d storm-nimbus start 15 2 3 4 5 . stop 15 0 1 6 .
sudo update-rc.d storm-supervisor start 20 2 3 4 5 . stop 20 0 1 6 .
```

2. Install the Service Level SIEM files in OSSIM

Once you have unzipped the Service Level SIEM file, go to the folder **SecurityProbe** to install the modified OSSIM agent and the plugins required for the collection and normalization of events in the Service Level SIEM. In this case, since the OSSIM standard distribution has been installed previously, it is only required to unzip the file *SecurityProbe.tar.gz* and execute the following command in order to update it:

```
# sudo python setup.py install --prefix=/usr
```

To install the Service Level SIEM Dashboard you only need to untar in the system under the folder */usr/share* the file **ServiceLevelSIEMdashboard.tar.gz** (provided in the file **ServiceLevelSIEM-3.3.3.tar.gz** into the folder *www*):

```
# cd www
# tar -C /usr/share -zxvf ServiceLevelSIEMdashboard.tar.gz
```

To add support for the use of the SSL protocol you will need to install the Unix application **socat**. In the folder *ossim-ssl* in the file **ServiceLevelSIEM-3.3.3.tar.gz** you can find the script *ossim-socat* to automatize the administration of the socat process and a dummy SSL certificate created for testing purposes. Follow the steps below to add

```
# sudo apt-get install sudo
# cd ossim-ssl
# cp ossim-socat /etc/init.d
# sudo update-rc.d ossim-socat start 20 2 3 4 5 . stop 10 0 1 6 .
# cp -pR ssl /etc/ossim
```

3. Database configuration

In order to perform the required modifications in the OSSIM databases to adapt them to the Service Level SIEM, the following MySQL script included in the file **ServiceLevelSIEM-3.3.3.tar.gz** into the folder *db* must be executed.

```
#cd db
#gunzip ServiceLevelSIEM.sql.gz
#mysql -uroot -p < ServiceLevelSIEM.sql
```

Once recreated the databases you will need to reset the user *admin* in the Service Level SIEM Dashboard with the following command that will provide you with a new password to be changed the next time you access the dashboard:

```
# /usr/bin/ossim-reset-passwd admin
```

If you already are using OSSIM and it has the databases installed and with data, you only will need to execute the script *sls_EPLdb.sql* (also located in the folder *db*) instead of the previous one to adapt it to the Service Level SIEM.

Once created the databases, a new user for FI-WARE must be created with permissions for the *alienvault* and *alienvault_siem* databases:

```
mysql> create user 'fiware'@'localhost' identified by '<password>';
mysql> grant all privileges on alienvault.* to 'fiware'@'localhost';
mysql> grant all privileges on alienvault_siem.* to
'fiware'@'localhost';
```

NOTE: If you have installed OSSIM from the source code instead of using the ISO image, it is possible you have errors in the dashboard because the functions *inet6_pton* or *inet6_ntop* are not found. In that case you can follow these instructions to add basic IPv6 and IDNA functions to MySQL v5:

```
# sudo apt-get install libc6-dev libmysqlclient-dev libidn11-dev
# wget https://bitbucket.org/watchmouse/mysql-udf-
ipv6/get/tip.tar.bz2
# tar -zxvf tip.tar.bz2
# cd watchmouse-mysql-udf-ipv6-c733da2d2703
# make
# sudo make install
# service mysql restart
```

And once connected to mysql:

```
mysql> CREATE FUNCTION inet6_pton RETURNS STRING SONAME
"mysql_udf_ipv6.so";

mysql> CREATE FUNCTION inet6_ntop RETURNS STRING SONAME
"mysql_udf_ipv6.so";

mysql> CREATE FUNCTION inet6_lookup RETURNS STRING SONAME
"mysql_udf_ipv6.so";

mysql> CREATE FUNCTION inet6_rlookup RETURNS STRING SONAME
"mysql_udf_ipv6.so";

mysql> CREATE FUNCTION inet6_mask RETURNS STRING SONAME
"mysql_udf_ipv6.so";
```

If the libraries are not found, please check the location of the plugins in your system and copy them from the default installation folder */usr/lib/mysql/plugin* to the one used in your mysql installation (e.g. /usr/lib64/mysql/plugin).

It is also possible you get the following error: *ERROR 1045 (28000): Access denied for user 'debian-sys-maint'@'localhost' (using password: YES)*. If this happens, it can be solved searching the debian-sys-maint password in the file */etc/mysql/debian.cnf* and executing the following instruction in MySQL:

```
mysql> GRANT ALL PRIVILEGES ON *.* TO 'debian-sys-maint'@'localhost'
IDENTIFIED BY "<password>" WITH GRANT OPTION;
```

4. Configuration of the Service Level SIEM:

Make sure the following configuration files are adapted to your system and requirements:

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

This is the OSSIM server configuration file where you need to include in the *[database]* section the right information to access the database. The ports and ip address will be configured by default but they can be also checked.

You need to include a new block *[ssl]* with the parameters shown below if you want to enable the use SSL (Secure Socket Layer) protocol for the communications between the SIEM agents and the server (*socat=yes*) and the location of the certificate (*cert=/etc/ossim/ssl/server.pem*).

```
[ssl]

socat=no

cert=/etc/ossim/ssl/server.pem

listen_port=50001
```

Finally you need to check the plugins included in the sections *[server]* and *[sensor]* and add the ones required for the security monitoring to be performed. For example, you need to *fiware* to the

parameters *[sensor] -> detectors* and *[server] -> server_plugins* if you want to be able of detecting events coming from FI-WARE Generic Enablers.

```
...
[sensor]
detectors=snare, syslog, fiware
...
[server]
...
server_plugins=osiris, pam_unix, ssh, snare, sudo, fiware
...
```

- **/etc/ossim/agent/config.cfg**

The detailed configuration of this file used by the OSSIM agent can be found in the *Security Probe Installation* section. Here we will just remark that the section *[output-storm]* need to be configured with the ip address where your Nimbus server 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 (see *ServiceLevelSIEM.conf* file).

```
[output-storm]
enable=True
ip=192.168.215.118
port=41000
send_events=True
```

The database user and password also need to be checked and configured in the following files:

- /etc/ossim/server/config.xml
- /etc/ossim/idm/config.xml
- /etc/ossim/framework/ossim.conf

Finally, once the configuration files have been modified, the OSSIM server and agent must be restarted in order to take the new parameters. The script **$STORM_HOME/admin/restart-ossim.sh** can be executed with this purpose or:

```
# service ossim-server restart
# service ossim-agent restart
```

- **$STORM_HOME/conf/ServiceLevelSIEM.conf**

This file includes different sections to configure the Service Level SIEM topology that will be running in the Storm Cluster: the listening port for the events coming from the ossim-agent, the schema used for the event and alarms, the information to be used in the Storm topology for the different parallel processing, the configuration for the use of the protocol SSL in the communication between agents and server, and the information to access the OSSIM databases.

```
//AGENT OSSIM PROPERTIES
spout_port = 41000

//INCOMING EVENT SCHEMA
schemaNames =
event_id,plugin_id,plugin_sid,date,device,interface,src_ip,dst_ip,
userdata1,userdata2,userdata3,userdata4,userdata5,userdata6,userdata
7,userdata8,
userdata9,fdate

//SLS ALARM PARAMETERS
alarmSchema = AlarmID,BacklogID,ListOfEventID
DirectiveSLS = 70000

// TOPOLOGY PARAMETERS
topologyName = "topologyPATTERN"
TOPOLOGY_WORKERS = 6
//Leave spout parallelism to 1 (To be analysed)
spoutParallelism = 1
// Preprocessing and filtering Bolt
schemaParallelism = 2
policyParallelism = 2
// Database writer Bolt
dbWriterParallelism = 2
actionParallelism = 2
crossCorrParallelism = 2

//DATABASE PARAMETERS
```

```
databaseIP = localhost
databasePort = 3306
databaseAlarms = alienvault
databaseEvents = alienvault_siem
userName = root
pwd = <password>

//SSL Configuration (TLS/SSLv3/NONE)
sslProtocol = TLS
keyStore = /etc/ossim/ssl/serverKeyStore.jks
keyStorePassword = <password>
keyStoreType = JKS
trustStore = /etc/ossim/ssl/serverTrustStore.jks
trustStorePassword = <password>
trustStoreType = JKS
```

3.3.1.5 *Supervisor Installation*

This step is only necessary in case you want to have a supervisory process to manage the ZooKeeper and Storm clusters. But it is very important to have it installed in a production environment because 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, and for this reason it requires the use of a supervisor process. 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 a patch adding "-include /usr/include/errno.h" to the file *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:12345:respawn:/command/svscanboot
```

The process *svscan* starts a supervisor process for each subdirectory in that */service* folder. Consequently, to supervise the Zookeeper it is necessary to create a directory called *zookeeper* and include inside the following script. The name of the script must be **run** and have execution permissions (*chmod 755 run*):

```
#!/bin/sh

exec su -l root -c "export JAVA_HOME=/usr/local/jdk1.6.0_37; cd
/opt/zookeeper && ./bin/zkEnv.sh && ./bin/zkServer.sh start &"
```

And the same for the storm processes *Nimbus* and *Supervisor* (depending on the host in the cluster where it must be running):

```
#!/bin/sh

exec su -l root -c "export STORM_HOME=/opt/storm; cd /opt/storm &&
./bin/storm nimbus &"


#!/bin/sh

exec su -l root -c "export STORM_HOME=/opt/storm; cd /opt/storm &&
./bin/storm supervisor &"


#!/bin/sh

exec su -l root -c "export STORM_HOME=/opt/storm; cd /opt/storm &&
./bin/storm ui &"
```

3.4 Administration

3.4.1.1 ***Run all the Storm daemons***

- *Nimbus*: Run the command "*storm nimbus &*" on the master machine.
- *Supervisor*: Run the command "*storm supervisor &*" 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 &*". The UI can be accessed by navigating your web browser to http://{nimbus host}:8080.

To automatize the administration of the Storm daemons as services in the system, the following scripts have been included that can be executed instead of the command storm:

- /etc/init.d/zookeeper: It starts the zookeeper process.
- /etc/init.d/storm-nimbus: It starts the nimbus daemon.
- /etc/init.d/storm-supervisor: It starts the supervisor daemon and start the Service Level SIEM topology.

The following order must be taken:

```
# service zookeeper start
# service storm-nimbus start
# service storm-supervisor start
```

The daemons will generate the logs files *Nimbus.log* and *Supervisor.log* in the **$STORM_HOME/logs** directory.
NOTE: If you are in a production environment, 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 so if the Nimbus or the Supervisors daemons restart, the running topologies are unaffected.

3.4.1.2 *Run all the OSSIM daemons*

The following daemons need to be started to have all the OSSIM functionalities available:

- alienvault-idm
- OSSIM server
- OSSIM framework
- OSSIM agent
- Socat Port Forwarder (this Unix daemon is used to redirect events arriving with the protocol SSL to the OSSIM server)

```
# service alienvault-idm start
# service ossim-server start
# service ossim-framework start
# service ossim-agent start
# service ossim-socat start
```

3.4.1.3 *Run 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 in the **$STORM_HOME/admin** folder.

```
# ./startSLTopology.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 worker processes are also located under the folder **$STORM_HOME/log**. A file is created by each worker process with the number of the port assigned to it. For example:

```
~/storm-0.9.0-wip16/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-wip16/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 for events with the following command:

```
~/storm-0.9.0-wip16/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-wip16/logs$
```

We can also check the topology has been started with the following command that lists all the active topologies in the Storm cluster:

```
# 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
```

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

```
# ./startSLTopology_localmode.sh
```

The scripts **stopSLSTopology.sh** and **restartSLSTopology.sh** are also provided in the *$STORM_HOME/admin* folder to respectively stop and restart the Service Level SIEM topology.

3.4.1.4 *OSSIM administration in detail*

The OSSIM daemons can be administrated with the following scripts:

- /etc/init.d/ossim-server start|stop|restart
- /etc/init.d/ossim-agent start|stop|restart

For any other administrative action, the Service Level SIEM Dashboard can be used.

3.4.1.5 *Storm topology administration in detail*

We can differentiate two situations:

- 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-3.3.3-SNAPSHOT-jar-with-dependencies.jar
storm.sls.main.SLSTopologyPattern
```

- 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-3.3.3-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 ServiceLevelSIEM
```

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_HOME}/admin** to do it automatically:

- startSLSTopology.sh (or startSLSTopology_localmode.sh)
- stopSLSTopology.sh
- restartSLSTopology.sh

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

3.4.1.6 *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:

- -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.

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.

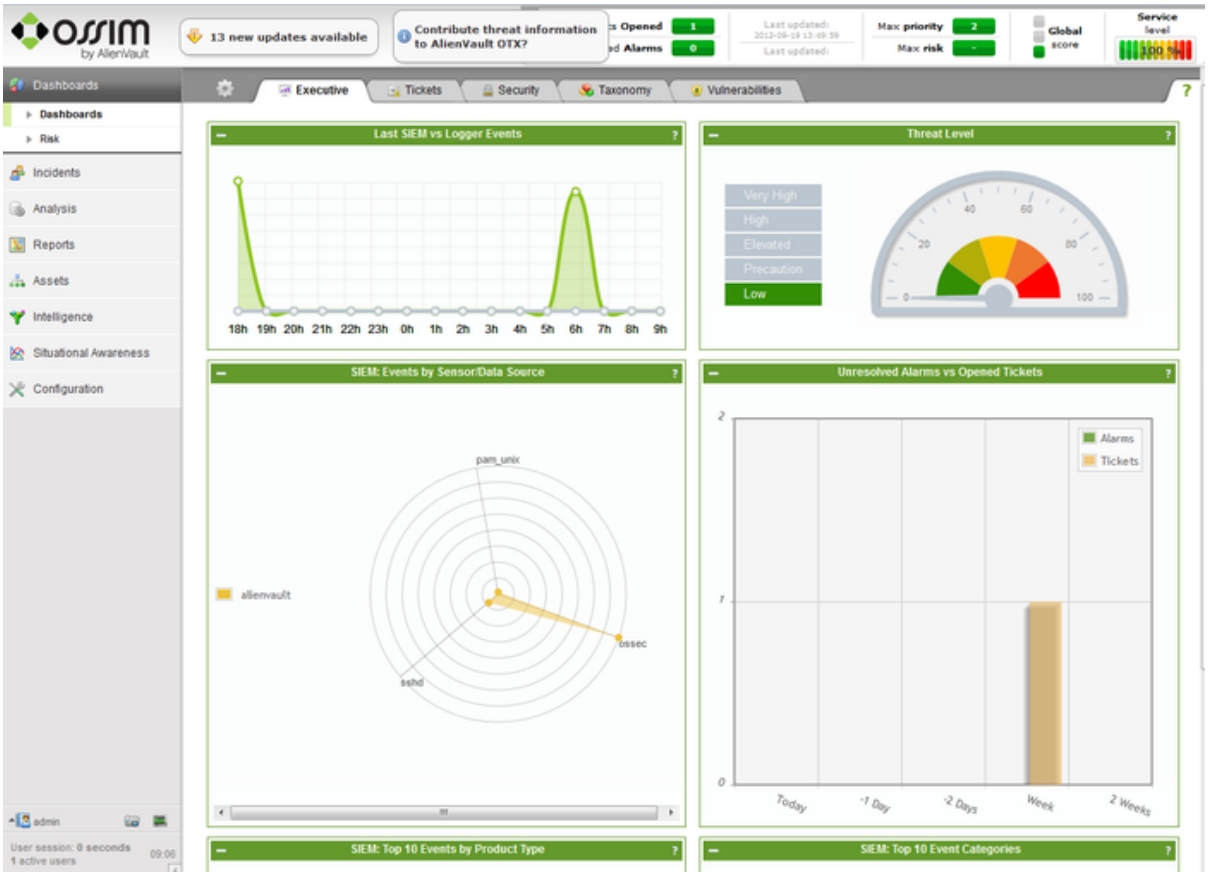
3.5.1 End to End testing

Users should validate the VM IP (130.206.81.27) is accessible for both SSH & HTTPS protocols:

- Under SSH protocol connection a standard Linux login screen is displayed
- Under HTTPS protocol connection the OSSIM Framework login page is displayed

OSSIM Framework login page

- Once login with the user and password provided for test (fi-test/siemtest) the main management OSSIM page will be displayed.



OSSIM management screen

In order to verify that basic OSSIM installation has been performed without problems, go to the Diagnosis Procedures section.

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 *ServiceLevelSIEM* 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
```

3.5.2 List of Running Processes

The following processes should be up and running:

- ossim-server (it can be started with: *service ossim-server start*)
- ossim-agent (it can be started with: *service ossim-agent start*)
- zookeeper (it can be started with: *service zookeeper start*)
- storm nimbus (in the master node) (it can be started with: *service storm-nimbus start*)
- storm supervisor (in the worker nodes) (it can be started with: *service storm-supervisor start*)
- mysql (it can be started with: *service mysql start*)

NOTE: The processes zookeeper, storm nimbus and storm supervisor can be running also within a *supervise* process. All of them will be running in the same server in a standalone installation.

3.5.3 Network interfaces Up & Open

The following ports must be listening for connections in the server host:

- port 40001/tcp - ossim-server
- port 40002/tcp - ossim-framework
- port 40003/tcp - alienvault-idm
- port 41000/tcp - spout to pass events to the Storm cluster
- port 50001/tcp - socat for the port forwarding of events coming from agents using SSL protocol
- port 514/udp - to receive events from the FI-WARE Generic Enablers through the Syslog Server

OSSIM listens by default on the Ethernet card **eth0** (that needs to be configured in promiscuous mode) for network event data.

The main administration interface is accessed from the standard SSL port **443** which can be accessed from here: *https://localhost*
Additionally, the Service Level SIEM uses a MySQL database available in the port **3306/tcp**.
The **SSH** and **SFTP** are listening for communication purposes in the port **22/tcp**.

3.5.4 Databases

The Service Level SIEM uses a MySQL database which is created and configured from the OSSIM ISO installation and adapted with the use of the script *sls_EPLdb.sql* included in the file *ServiceLevelServiceLevelSIEM-3.3.3.tar.gz*. It can be also be completely recreated with the use of the script *ServiceLevelSIEM.sql* also included in the file *ServiceLevelServiceLevelSIEM-3.3.3.tar.gz*.

A password is created randomly at installation time and it needs to be stored in the file:

```
/etc/ossim/ossim_setup.conf
```

Inside this file there is section for the DB configuration, including the IP, port, username and database names.

```
[database]
acl_db=ossim_acl
db_ip=127.0.0.1
db_port=3306
event_db=snort
ocs_db=ocsweb
ossim_db=ossim
osvdb_db=osvdb
pass=#PASSWORD#
rebuild_database=no
type=mysql
user=#USER#
```

The following OSSIM databases must be up and accept queries:

- alienvault
- alienvault_siem

The connection details are included in the *$STORM_HOME/conf/ServiceLevelSIEM.conf* file.

Test query (to check the databases are created trying to access to one of their tables):

```
```

```
# myql -u fiware -p
mysql> select count(*) from alienvault.server;
+----------+
| count(*) |
+----------+
|        1 |
+----------+
1 row in set (0.00 sec)

mysql> select count(*) from alienvault_siem.device;
+----------+
| count(*) |
+----------+
|        2 |
+----------+
1 row in set (0.13 sec)
```

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

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:

3 GB of RAM

And comfortable use is:

8 GB of RAM

3.6.2 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

3.6.3 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 up to 8 GB of RAM. Two CPUs are required for the use of this GE.

3.6.4 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 is installed in different machines.

3.7 References

- OSSIM SIEM (Alienvault): http://www.alienvault.com
- 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

4 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.

4.1 Introduction

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

4.2 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/

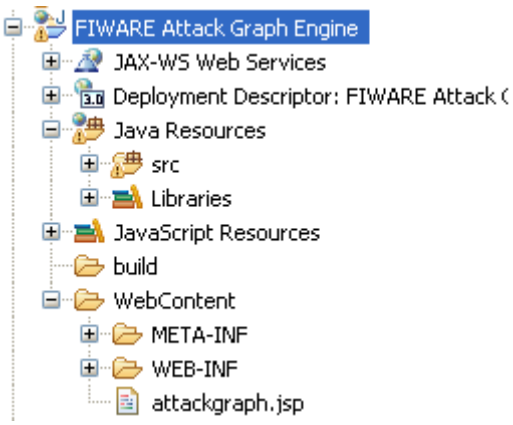
4.3 Installation

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 developement project.

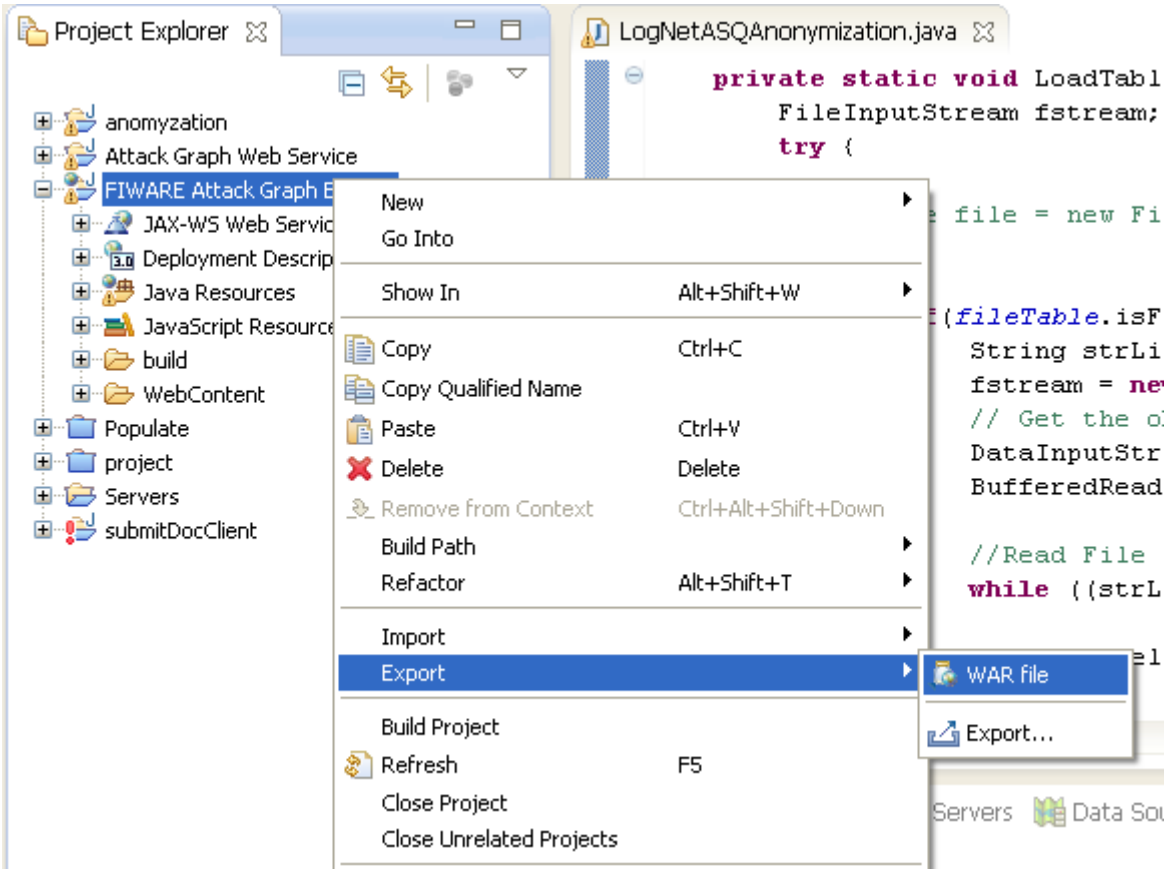
Once the web application project is imported, the files structure looks like:



files structure of web application project

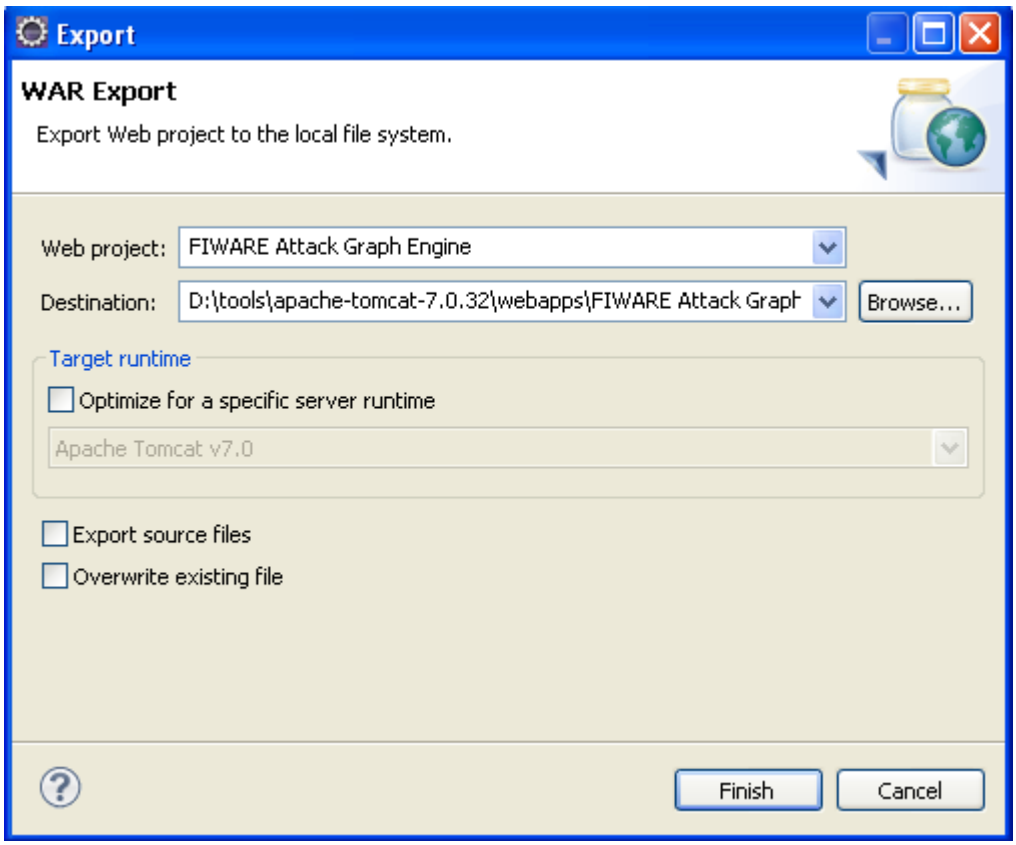
After importing the project, you can export a WAR file. The process of exporting this WAR file is given in following pictures.

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 Appache Root folder"



Seetting where to deploy the WAR File

Click on Finish to deploy the WAR.

4.4 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.

4.5 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

4.5.1 End to End testing

Open a web browser (e.g. Firefox or Internet Explorer) and put the following line in the address bar: http://<Base URL>/AttackGraphEngine/attackgraph.jsp to access the Mulval Attack Paths Web Engine Web app.

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.

4.5.2 List of Running Processes

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

4.5.3 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.

4.5.4 Databases

N/A

4.6 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.

4.6.1 Resource availability

The needed resoure 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

4.6.2 Remote Service Access

N/A

4.6.3 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.

4.6.4 I/O flows

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

4.7 References

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

http://tomcat.apache.org/

5 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.

5.1 Introduction

The Scored Attack Path application provides tools to security operators for scoring the risk and business process impact of attack paths. This application is written in Java and has a web-interface. This document will explain how to install this application.

5.2 Requirements

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

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

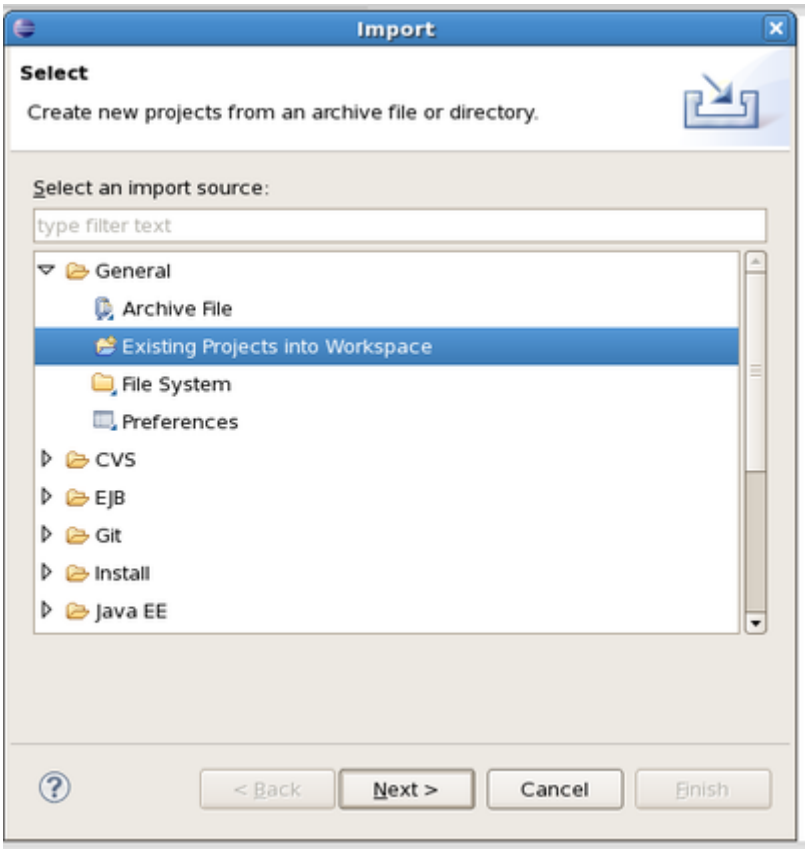
In order to successfully deploy the vulnerability assessment and remediation prototype, the software listed below is required:

- 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.

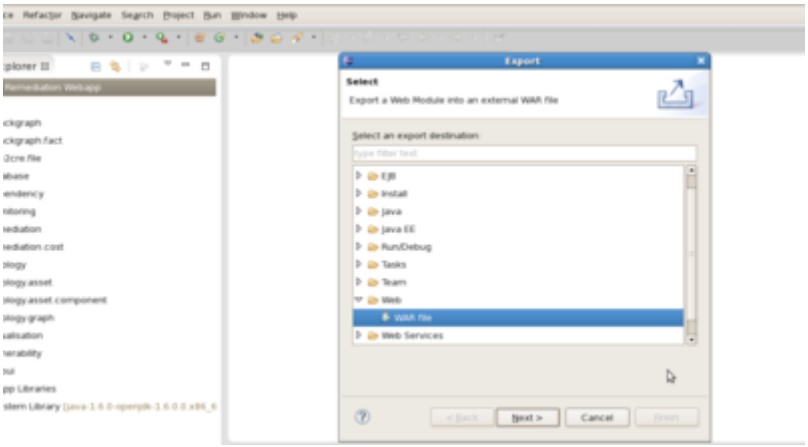
5.3 Installation

5.3.1 Building application

The build procedure of the Scored Attack Path 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 Scored Attack Path application prototype contains a ready to use Eclipse project and class-path descriptor file (.project .classpath). Therefore, to build the Scored Attack Path 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 Scored Attack Path 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.



5.3.2 Installation instructions

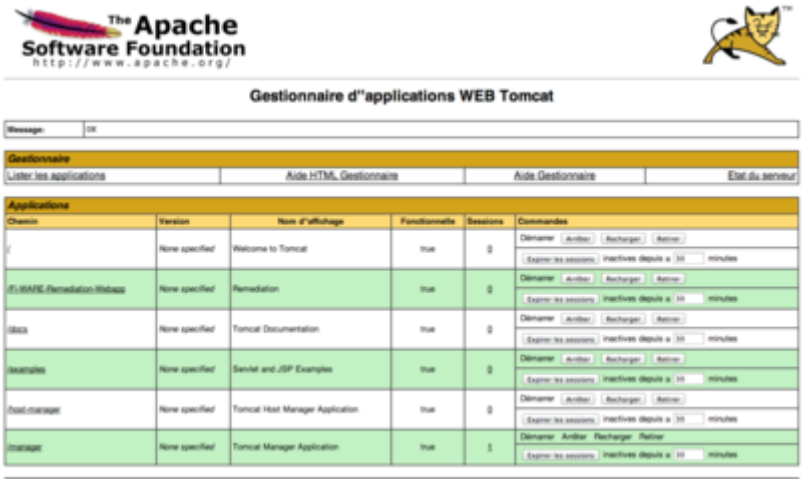
In this section, the installation of the Scored Attack Path application is described. Although, the screenshots show the installation process in an CentOS 5 environment, the installation on Windows and other platforms which are capable of executing the Java platform and Tomcat follows the same logic.

5.3.2.1 *Configuration files needed by the application*

No particular configuration files are required by the Scored Attack Path application.

5.3.2.2 *Installation of the Scored Attack Path application on Tomcat*

The Scored Attack Path 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.



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

The successful installation of the Scored Attack Pathapplication 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://<Base URL address>/Fi-Ware-Scored-Attack-Paths-Webapp

5.4.2 List of Running Processes

"tomcat" process should be running.

5.4.3 Network interfaces Up & Open

Ports used by tomcat (generally 8080 or 80).

5.4.4 Databases

The Scored Attack Path application requires no databases.

5.5 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.

5.5.1 Resource availability

The needed resoure 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

5.5.2 Remote Service Access

N/A

5.5.3 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.

5.5.4 I/O flows

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

6 Security Monitoring / Remediation - Installation and Administration Guide

You can find the content of this chapter as well in the wiki of fi-ware.

6.1 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.

6.2 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 Ubuntu 13.10 and successfully tested under Ubuntu 12.04 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.

Both environment variables JAVA_HOME and JRE_HOME must be set to the path where JRE/JDK is installed. If it is not possible to set it globally, you can put it directly in MulVAL by appending the following lines in the top of the $MULVALROOT/utils/graph_gen.sh file :

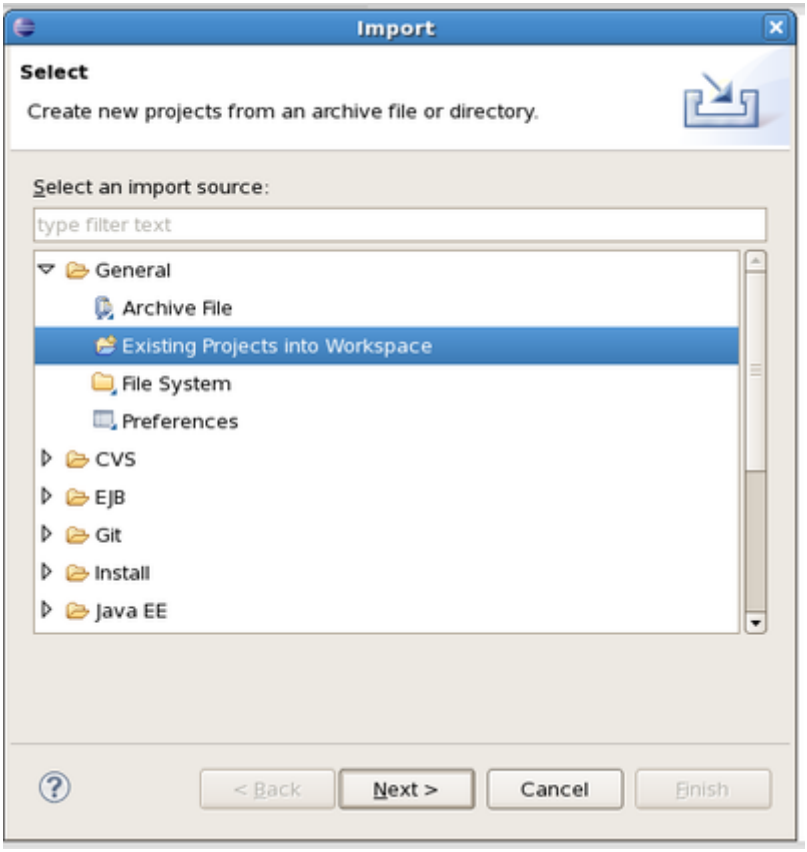
```
export JAVA_HOME=/path/to/JDK/or/JRE
export JRE_HOME=/path/to/JDK/or/JRE
```

6.3 Installation

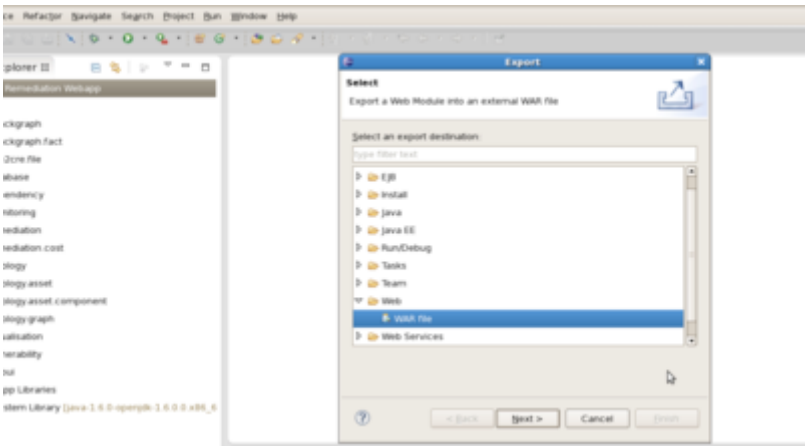
6.3.1 Building application

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.



6.3.2 Installation instructions

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

6.3.2.1 *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 launch as root, it is generally "/root" if tomcat is launch as "www-data", 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"):

- xsb-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 topology XML file.

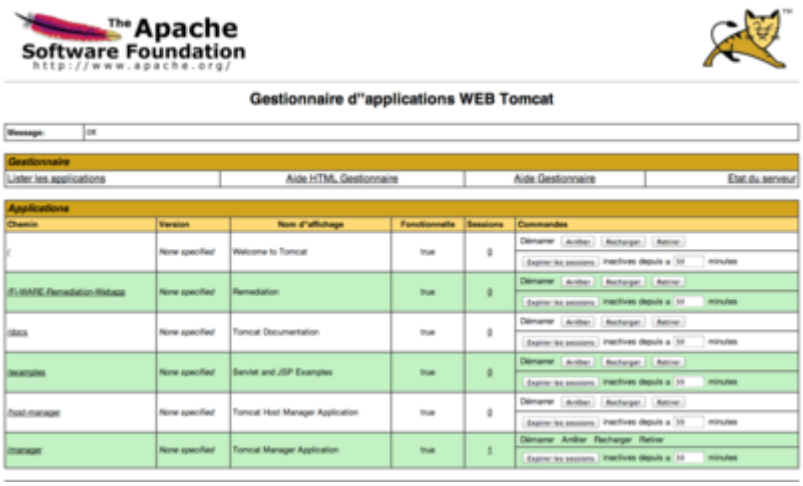
The easier to do is to put all the files and folder referenced in "config.properties" in the "~/.remediation/" folder ("./cost-parameters/" "./database.db" "./topology.xml") This can be done with the following command :

```
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.

6.3.2.2 *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 below a screenshot of the deployment of the war file on Tomcat 7.



6.3.3 Filling or updating the remediation database (optional)

In addition to the remediation application are given 3 python scripts that allow to either fill or update the remediation database from 3 different sources of data: the NVD (National Vulnerability Database), Microsoft bulletins, or Oracle patches information.

6.3.3.1 *Filling or updating the remediation database from the National Vulnerability Database*

To create a new remediation database, just launch the two commands below :

```
mkdir -p resources/nvd
python main.py
```

It will create the necessary resources and will then create an complete the remediation database located in resources/database.db . Notice that the execution of this script may be very long (several hours, depending of your internet connection, because it will first download the NVD files ~500Mo) and then add it to the remediation database.

6.3.3.2 *Add Microsoft Bulletins to the remediation database*

In order to add the patches information coming from the Microsoft Bulletins to the remediation database, first download the Microsoft Security Bulletin Data from Microsoft's website (MSRC-CVRF.zip): http://www.microsoft.com/en-eg/download/details.aspx?id=36982. Then, execute the following commands:

```
mkdir resources
unzip MSRC-CVRF.zip
mv MSRC-CVRF-* resources/MSRC-CVRF
mv <path-to-remediation-database> resources/database.db
python main.py
```

This will update the resources/database.db remediation database with all the patches information coming from Microsoft Bulletins

6.3.3.3 *Add Oracle patches information to the remediation database*

In order to add the patches information coming from Oracle to the remediation database, execute the following commands:

```
mkdir resources
mv <path-to-remediation-database> resources/database.db
python main.py
```

This will download patches information from Oracle website, then update the resources/database.db remediation database with all the patches information coming from Oracle

6.4 Sanity check procedures

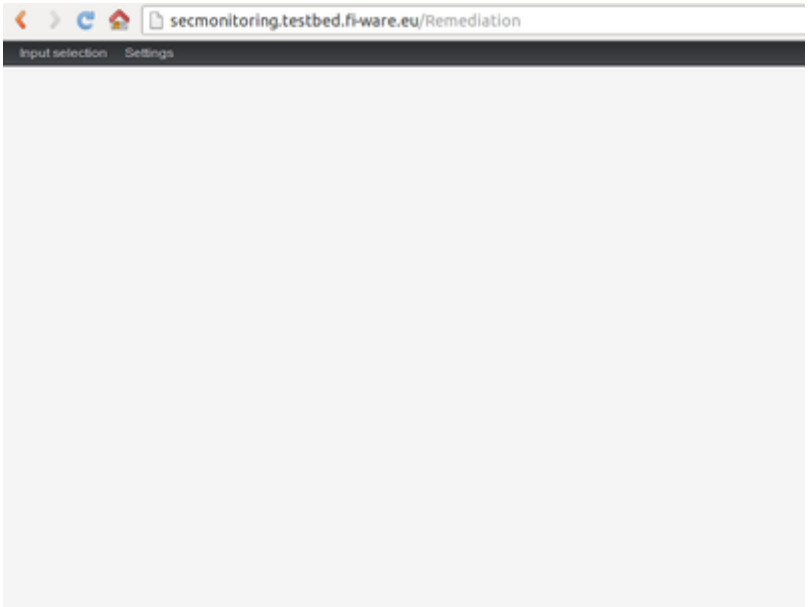
The Sanity Check Procedures are the steps that a System Administrator will take to verify that an installation is ready to be tested. This is therefore a preliminary set of tests to ensure that obvious or basic malfunctioning is fixed before proceeding to unit tests, integration tests and user validation.

6.4.1 End to End testing

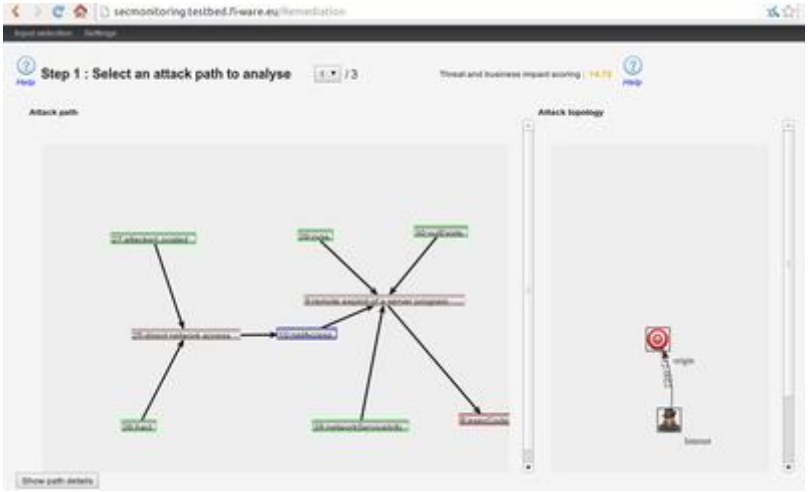
The successful installation of the remediation application can be verified by accessing the URL of the remediation application with the internet browser:

http://<Base URL address>/Remediation.

If the installation was successful, the page as depicted in screenshot below is visible and no error messages are shown (note the grey menu-bar in the top of the application).



It is normal that first, the application seems empty, as you have to select the input to draw the graph. For a first test, please click on "Input selection" -> "Load default datas from disk". If a graph is shown as on screenshot bellow, the application is working.



If it is not the case, a problem occured when generating the attack graph. If details about the problem are not shown in the main window, more detailed information about the problem could be either in Tomcat log file $TOMCAT_ROOT/logs/catalina.out, or in XSB log file : /root/.remediation/xsb_log.txt.

One common problem that occur when the Java JRE has not been installed with a packet manager is that the environment variable JAVA_HOME is not filled properly. This can be fixed as explained in Requirements

6.4.2 List of Running Processes

"tomcat" process should be running.

6.4.3 Network interfaces Up & Open

Ports used by tomcat (generally 8080 or 80).

6.4.4 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 patchs"

6.5 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.

6.5.1 Resource availability

The needed resoure 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

6.5.2 Remote Service Access

N/A

6.5.3 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.

6.5.4 I/O flows

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

7 Security Monitoring / Visualisation Framework - Installation and Administration Guide

You can find the content of this chapter as well in the wiki of fi-ware.

7.1 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. This web application has been tested in the latest version of Google Chrome and Firefox.

7.2 Installation

7.2.1 Requirements

The Visualisation Framework requires the following software to be installed:

- Tomcat 7 [1]
- Maven [2]
- HSQLDB [3]
- Google Chrome [4]

The Visualisation Framework requires the following Generic Enablers:

- Security Monitoring GE MulVal Attack Paths Engine [5]
- Security Monitoring GE Service Level SIEM [6]

7.2.2 Configuring External Databases

The default behaviour of the visualisation framework is to use a single database for all data. External databases can be defined in certain circumstances. The default database used by the Visualisation Framework for its own data is HSQLDB, however this can be configured by editing the `jdbc.properties` file located at `com.thalesgroup.uk.trt.fiware.web.webapp/src/main/resources/`.

7.2.2.1 *Configuring external SIEM database*

Create the file '`jdbc-siem.properties`' and place in '`com.thalesgroup.uk.trt.fiware.web.webapp\src\main\resources\`'. An example of the file contents is shown below; replace values with required connection values:

```
jdbc-siem.driver=com.mysql.jdbc.Driver
jdbc-siem.url=jdbc:mysql://localhost:3306/alienvault
```

```
jdbc-siem.username=fiware
jdbc-siem.password=f1war3
jdbc-siem.dialect=org.hibernate.dialect.MySQL5Dialect
```

The driver should be added to the maven .POM file for the project in order to pull in the correct dependencies. The project should be rebuilt and redeployed after configuring an external database.

7.2.2.2 *Configuring external Attack Graph database*

Create the file 'jdbc-attackgraph.properties' and place in 'com.thalesgroup.uk.trt.fiware.web.webapp\src\main\resources\'. An example of the file contents is shown below; replace values with required connection values:

```
jdbc-attackgraph.driver=com.mysql.jdbc.Driver
jdbc-attackgraph.url=jdbc:mysql://localhost:3306/attackgraph
jdbc-attackgraph.username=fiware
jdbc-attackgraph.password=f1war3
jdbc-attackgraph.dialect=org.hibernate.dialect.MySQL5Dialect
```

The driver should be added to the maven .POM file for the project in order to pull in the correct dependencies. The project should be rebuilt and redeployed after configuring an external database.

7.2.2.3 *Configuring external IDMEF database*

Create the file 'jdbc-idmef.properties' and place in 'com.thalesgroup.uk.trt.fiware.web.webapp\src\main\resources\'. An example of the file contents is shown below; replace values with required connection values:

```
jdbc-idmef.driver=com.mysql.jdbc.Driver
jdbc-idmef.url=jdbc:mysql://localhost:3306/idmef
jdbc-idmef.username=fiware
jdbc-idmef.password=f1war3
jdbc-idmef.dialect=org.hibernate.dialect.MySQL5Dialect
```

The driver should be added to the maven .POM file for the project in order to pull in the correct dependencies. The project should be rebuilt and redeployed after configuring an external database.

7.2.3 Build the Visualisation Framework

The Visualisation Framework can be downloaded from the FI-WARE PPP Restricted Project website [7] as a zip file. The zip file contains the Maven project

'com.thalesgroup.uk.trt.fiware.web.webapp' (which needs 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 project has been unzipped, we need to build them in order to generate the binaries to install. To do so, we run the following commands:

```
$ cd com.thalesgroup.uk.trt.fiware.web.webapp
$ mvn clean package
```

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

7.2.4 Install the Visualisation Framework

The Visualisation Framework web application is installed by copying the WAR file 'fiware-dashboard.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.

7.3 Launching the Visualisation Web Application

In order to run the Visualisation Framework web application the following applications need to be launched (this assumes the default configuration of the internal database has been used):

- 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.fiware --dbname.0 fiware.app.fiware
```

- Java application server, such as Tomcat

7.4 Administration

The Visualisation Framework web application requires a user login to access any of the web pages. Default users have been configured for this release; it is currently not possible to add new users without rebuilding the project. Default users are explained in the table below.

Users		
Username	**Password**	**Roles**

siem	siem	ROLE_USER, ROLE_SIEM
ag	ag	ROLE_USER, ROLE_AG
admin	admin	ROLE_USER, ROLE_SIEM, ROLE_AG, ROLE_ADMIN

Assigning roles to users allow authorisation to certain pages. For example the user '`siem`' can access the SIEM web pages, while this user cannot access ATTACK GRAPH pages. Roles are described in the table below.

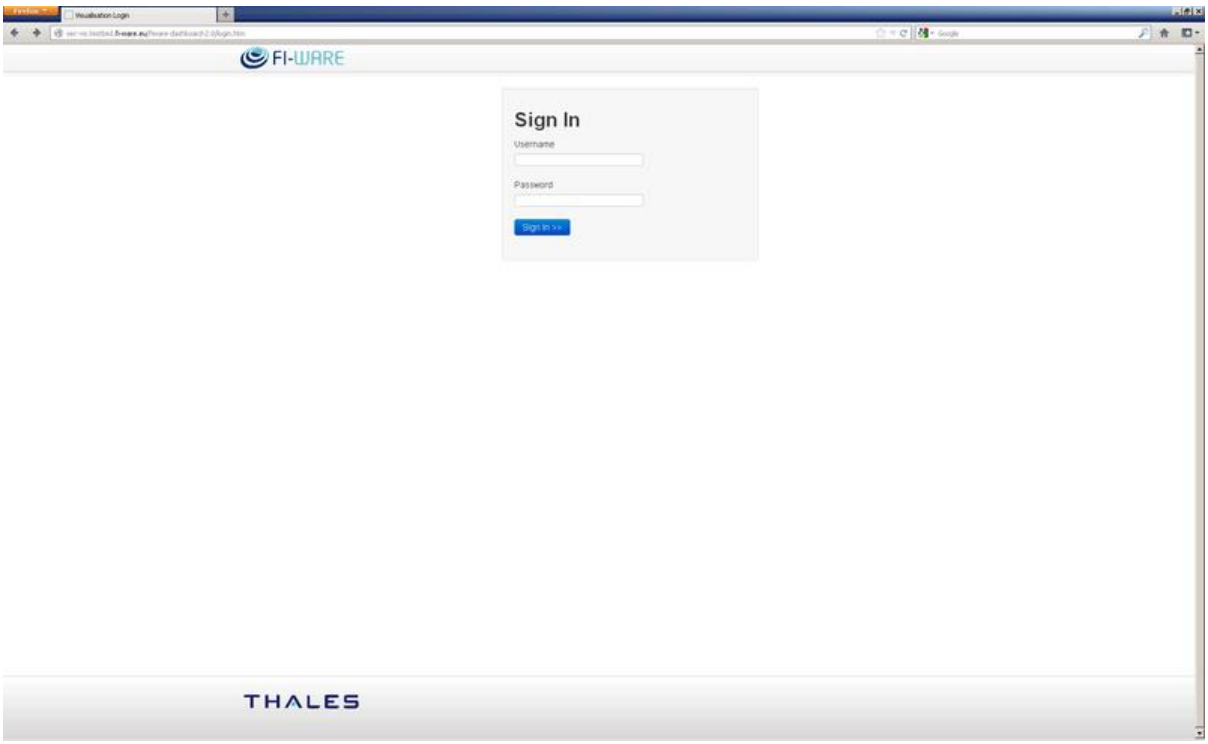
Roles		
Role	**Description**	**Mapping**
ROLE_USER	General access to the dashboard, required by all web pages	/
ROLE_ADMIN	Access to the administration pages	/admin/
ROLE_SIEM	Access to SIEM data and web pages	/siem/
ROLE_AG	Access to ATTACK GRAPH data and web pages	/attackgraph/

7.5 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.

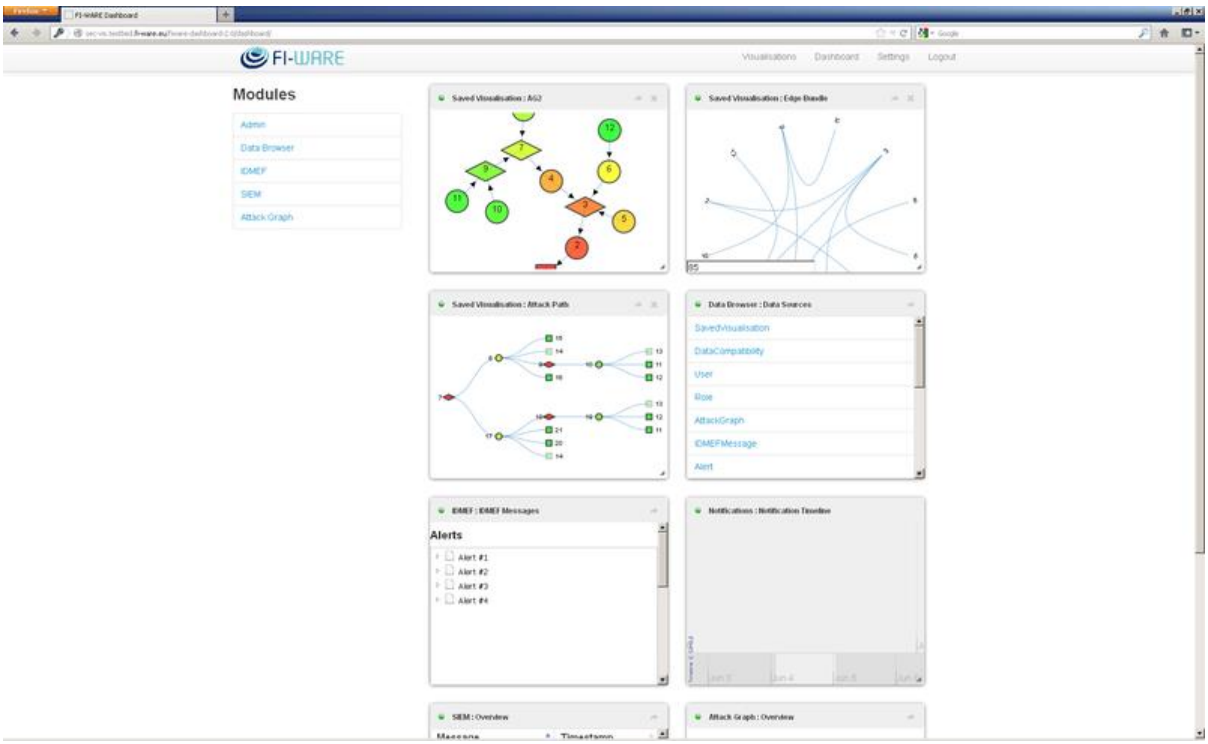
7.5.1 End to End testing

To verify access to the Visualisation Framework, navigate to the login page (shown below) and attempt to login to the web application.



Web application login page

If the login is successful, a page similar to the one in the figure below will be shown. Successful login proves the web application server is running and responding to requests. Navigating to the login page via a logout redirect proves the user is logged out prior to log in, and that browser cache is not being used. http://localhost:<Tomcat HTTP port>/fiware-dashboard/logout.htm



Web application dashboard, shown after successful login

7.5.2 List of Running Processes

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

7.5.3 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.

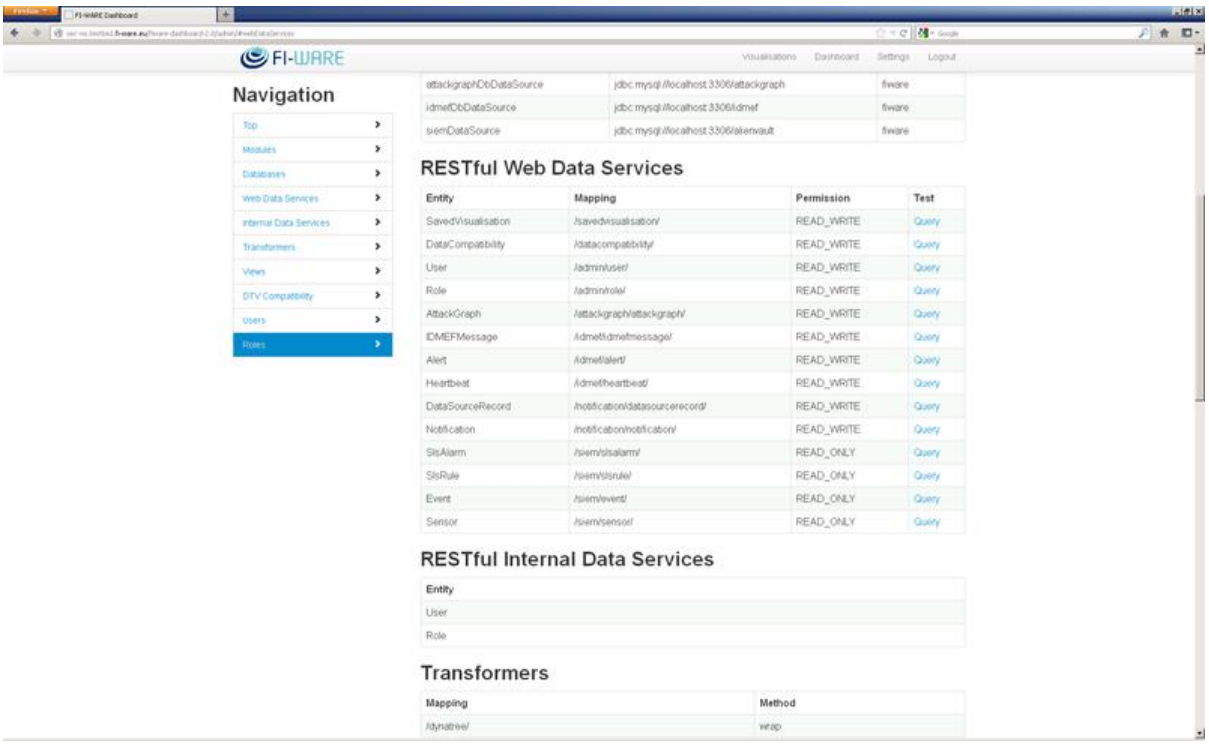
7.5.4 Databases

As a minimum, the HDBSQL database should be running and accepting connections for the following configuration:

- URL: jdbc:hsqldb:hsql://localhost/fiware.app.attackgraph
- Port: 9001
- Username: sa
- Password: <blank>

Extra databases may be configured depending on installation. For example, an external database may be used for SIEM. Databases can be queried from the "`Admin`" page to sanity check connection errors via the following steps:

1. Navigate to the web application and sign in as "`admin`"
    1. Username "`admin`", password "`admin`"
2. Click the link in the left side of the web application labelled "`Admin`"
3. Find the section labelled "`RESTful Data Services`"



Querying a data service from the admin page

4. Clicking the '`query`' link in the '`Test`' column will query the data service for all entity identifiers. If this is successful a simple text string listing an array of identifiers will be shown (for example `{"ids":[1,2,3,4,5,6]}`). If any errors occur they will be displayed in the following page, an error message will be provided.

File:Sec-Visf-error.png

Web application error page

If an error occurs ensure the database is running and accepting connections using a database data viewer tool such as SQuirreL [8].

7.6 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.

7.6.1 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

7.6.2 Remote Service Access

N/A

7.6.3 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.

7.6.4 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.

7.7 References

1. ↑ http://tomcat.apache.org/tomcat-7.0-doc/setup.html
2. ↑ [2.0] [2.1] http://maven.apache.org/
3. ↑ http://hsqldb.org
4. ↑ http://www.google.com/intl/en_uk/chrome/browser/
5. ↑ Security Monitoring / MulVAL Attack Paths Engine - Installation and Administration Guide
6. ↑ Security-Monitoring:_Service_Level_SIEM_Open_API_Specification
7. ↑ https://forge.fi-ware.org/frs/?group_id=23
8. ↑ http://squirrel-sql.sourceforge.net/

8 Identity Management - KeyRock - Installation and Administration Guide

You can find the content of this chapter as well in the wiki of fi-ware.

8.1 Introduction

Welcome to the Installation and Administration Guide for the Identity Management - KeyRock Generic Enabler. This generic enabler is built on an Open Source project, and so where possible this guide points to the appropriate online content that has been created for this project. The online documents are being continuously updated and improved, and so will be the most appropriate place to get the most up to date information on installation and administration.

8.1.1 Requirements

This installation guide is made for install Identity Management - KeyRock in a Ubuntu 12.04 (LTS) server.

8.2 System Installation

- **Running KeyRock locally**

1.Install Ubuntu dependencies

```
sudo apt-get install python-software-properties python g++ make

sudo add-apt-repository ppa:chris-lea/node.js

sudo apt-get update

sudo apt-get install git ruby1.9.1 ruby1.9.1-dev build-essential
openssl libreadline6 libreadline6-dev curl

git-core zlib1g zlib1g-dev libssl-dev libyaml-dev libsqlite3-dev
sqlite3 libxml2-dev libxslt-dev autoconf

libc6-dev ncurses-dev automake libtool bison  subversion pkg-config
libmysql-ruby libmysqlclient-dev imagemagick

graphicsmagick libmagickwand-dev sendmail-bin nodejs mysql-server
```

The last one is mysql server so you will need to create a root password to manage it.

Note: Here you will find instrucctions to install Ruby using rvm.

2.Install ruby gems for Rails and Capistrano

```
sudo gem install capistrano rails
```

3.Clone de repository and install KeyRock dependencies

```
git clone https://github.com/ging/fi-ware-idm

cd fi-ware-idm/
```

```
bundle install
```

4.Configure the database and populate it

```
cd config/

cp database.yml.example database.yml
```

Edit database.yml file setting your mysql password (the one that you have setted when installing mysql) and the database name for development, test, production (for development it is enough with the first one).

Create the databases in mysql

```
mysql -u root -p

> create database db_name
```

Populate the databases

```
bundle exec rake db:schema:load
```

5.Run de server

```
rails s
```

- **Deploying with Capistrano**

You must have an Ubuntu server with ruby and passenger installed.

1.Download the code into your work computer

```
git clone https://github.com/ging/fi-ware-idm

cd fi-ware-idm
```

2.Install capistrano

You must have capistrano installed in your work computer

```
gem install capistrano
```

There is an deploy configuration example distributed with the code. Copy and configure it with the details of your Ubuntu server:

```
cp config/deploy.example.rb config/deploy.rb

nano config/deploy.rb
```

3.Deploy to your server

First, you must initialize the capistrano deploy

```
cap deploy:setup
```

Then, you can deploy new code using:

```
cap deploy
```

8.3 System Administration

- **White and black lists**

As administrator of IdM KeyRock you can manage white and black lists in order to allow and deny access to users by their email domains.

There is a configuration example distributed with the code. Copy and configure it uncommenting the lines of the list that you are going to use.

```
cp config/initializers/0fiware.rb.example
config/initializers/0fiware.rb

nano config/initializers/0fiware.rb
```

White list:

```
#List of email domains that are allowed to register


whiteListPath = "config/emailLists/whitelist.txt"

if File.exists?(whiteListPath)

   config.allowed_email_domains =
File.read(whiteListPath).split("\n")[0].split(", ")

else

   config.allowed_email_domains = %w( example.com other-example.com
)

end
```

Black list:

```
#List of email domains that are not allowed to register


blackListPath = "config/emailLists/blacklist.txt"

if File.exists?(blackListPath)

   config.forbidden_email_domains =
File.read(blackListPath).split("\n")[0].split(", ")

else

   config.forbidden_email_domains = %w( example.com other-
example.com )

end
```

Then, in config/emailList, copy and edit blacklist.txt.example or whitelist.txt.example with the email domains separated by commas.

```
cp config/emailList/blacklist.txt.example
config/emailList/blacklist.txt

nano config/emailList/blacklist.txt

cp config/emailList/whitelist.txt.example
config/emailList/whitelist.txt

nano config/emailList/whitelist.tx
```

8.4 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.

8.4.1 End to End testing

1. Verify that the host address of IdM can be reached. By default, web access will show a Login Page.

2. Acquire a valid username and password and access with those credentials.

The resulting web page is the landing page of the IdM KeyRock Portal.

3. Verify that you can lview the list of applications, organizations, etc.

8.4.2 List of Running Processes

In you have run "rails s" command without errors, the portal is up and running.

8.4.3 Network interfaces Up & Open

- TCP port 80 (3000 in development mode) should be accessible to the web browsers in order to load the IdM Portal.

8.4.4 Databases

If you have correctly populated the database when installing the GE, the connection with it is up and running.

The databases and tables needed are:

```
 Tables_in_fi-ware-idm
 ------------------------------------
 | activities                       |
 | activity_actions                 |
 | activity_object_activities       |
```

```
| activity_object_audiences         |
| activity_object_properties        |
| activity_objects                  |
| activity_verbs                    |
| actors                            |
| audiences                         |
| authentications                   |
| comments                          |
| contacts                          |
| conversations                     |
| groups                            |
| notifications                     |
| oauth2_tokens                     |
| permissions                       |
| posts                             |
| profiles                          |
| receipts                          |
| relation_permissions              |
| relations                         |
| schema_migrations                 |
| simple_captcha_data               |
| sites                             |
| taggings                          |
| tags                              |
| ties                              |
| users                             |
```

You can test each table making a query like this:

```
SELECT * FROM "table_name";
```

8.5 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.

8.5.1 Resource availability

- Verify that 2.5MB of disk space is left using the UNIX command 'df'

8.5.2 Remote Service Access

Please make sure port 80 is accessible (port 3000 in development mode).

8.5.3 Resource consumption

Typical memory consumption is 100MB and it consumes almost the 1% of a CPU core of 2GHz, but it depends on user demand.

8.5.4 I/O flows

Clients access the KeyRock Interface through the client's Web Browser. This is simple HTTP traffic. It makes requests to the local database.

9 Privacy - Installation and Administration Guide

You can find the content of this chapter as well in the wiki of fi-ware.

9.1 Privacy-Preserving Attribute-Based Credential Engine

This Privacy-Preserving Attribute-Based Credential Engine enables application developers to use Privacy-ABCs with all their features without having to consider the specifics of the underlying cryptographic algorithms, similar to as they do today for digital signatures, where they do not need to worry about the particulars of the RSA and DSA algorithms either.

9.2 Overview of Codebase

We here provide a number of core components for authentication with privacy-preserving attribute-based credentials. These core components deal with the policy language that specifies the authentication requirements and components to generate and verify authentication tokens.

Building a full-fledged authentication and authorization solution requires a number of additional components such as credential storage or key management. We provide basic implementations of such components and of an example application as well. While these components could be used as well, you might want/have to replace them with your own ones to integrate the core components into your application.

Finally, we do not provide any cryptographic library which generates the cryptographic values in the authentication token. However, our engine is designed to be used with Identity Mixer and U-Prove. So all you need to do is to download either Microsoft U-Prove and/or IBM Identity Mixer separately (some of the the features will only work if Identity Mixer is installed and used).

9.3 How to Build

Note that we tested the correctness of the following instructions on a Ubuntu Server 12.04 Linux distribution.

9.3.1.1 *Requirements*

The following components are required for building the project:

- Java Development Kit (JDK) 1.6 or higher. Note that the Java Runtime Environment (JRE) is not sufficient.

  Unix: `sudo apt-get install openjdk-6-jdk`

- Maven 3.0.x

  Unix: `sudo apt-get install maven`

  Although we use Maven as build tool, there are two required libraries that are not available in public Maven repositories. Therefore, in the following, we provide instructions on how these libraries can be integrated with your local Maven repository.

- IBM Identity Mixer Version 2.3.43.

Download the binary (com.ibm.zurich.idmx.2-3-43.jar) through a web browser, and install it into your local maven repository on your VM:

```
mvn install:install-file \
   -DgroupId=com.ibm.zurich \
   -DartifactId=idmx \
   -Dpackaging=jar \
   -Dversion=2-3-43 \
   -Dfile=com.ibm.zurich.idmx.2-3-43.jar \
   -DgeneratePom=true
```

(i.e. if you downloaded the binary to your local machine, you have to e.g. SCP it to the VM)

- PLT Utilities, which is a component of the DrJava project.

  Download the binary (plt.jar) and install it into your local maven repository:

```
mvn install:install-file \
   -DgroupId=plt \
   -DartifactId=plt \
   -Dpackaging=jar \
   -Dversion=1.0 \
   -Dfile=plt.jar \
   -DgeneratePom=true
```

  (e.g. you can download it directly on the VM with the following command `wget https://drjava.svn.sourceforge.net/svnroot/drjava/trunk/drjava/lib/plt.jar`)

- Microsoft .NET runtime version 4 *FULL Profile*
  or
  Mono project version > 2.8
  Ubuntu: `sudo apt-get install mono-complete` (Note: Ubuntu does not come with Mono installed by default any more. Also, note that the *mono-runtime* package is not sufficient).

9.3.1.2 *Building*

Note: Make sure all the shell scripts (ending with .sh) used in the following are executable (e.g., by using `chmod +x fileName.sh`). Also, if you get an error message 'bad interpreter' when executing one of the following shell scripts, make sure that all line endings (invisible characters) are converted to unix style. Also make sure, that before maven is used, the `JAVA_HOME` environment variable is exported and set to the correct JDK classpath (e.g. on linux it could be done with the following command `export JAVA_HOME=/usr/lib/jvm/java-6-openjdk-amd64/` with adjusting the path to the correct one).

9.3.1.2.1 *Building the core components*

1. Only on Windows 7: start the ABC4Trust-UProve Service:

```
ABC4Trust-UProve.exe 32123
```

   This executable is located in the folder `Code/uprove-java-binding/ABC4Trust-UProve_dotNET_WebServiceServer/ABC4Trust-UProve/bin/Release/`.

   On Windows XP/Unix/Mono platforms, the service is started automatically.

2. Acquire the ABCE codebase from one the following sources:

   - From FI-Ware Forge (e.g. with the command `wget --no-check-certificate https://forge.fi-ware.org/frs/download.php/1274/privacyPreservingAuthentication-3.3.3.zip`)
   - From the Github repository (e.g. with the command `wget https://github.com/p2abcengine/p2abcengine/archive/master.zip`)

   Unzip the acquired zip file. Note: if you copied the source from Github, change the directory to `p2abcengine-master` after unzipping.

3. Change directory:

```
cd Code/core-abce
```

4. Build the code with the command

```
mvn clean install -DskipTests
```

   If the build fails with java.lang.OutOfMemoryError exception, make sure the Maven build process has enough memory:

   - Windows: `set MAVEN_OPTS=-Xmx1024m -Xms256m -XX:MaxPermSize=512m`
     Be aware that the 'set' command only sets the MAVEN_OPTS variable for the current console session. To have the variable set permanently (for all future console sessions), set this variable as Windows environment variable manually or via 'setx'.
   - Unix variants: `export MAVEN_OPTS='-Xmx1024m -Xms256m -XX:MaxPermSize=512m'`
     In Unix, to prevent this common error, these options are set automatically if you run `mvn` from the core-abce folder.

5. Once the code can successfully be built, you can take the next step and execute all unit tests:

   Windows XP/Unix/Mono platforms: First, you need to determine the absolute path to the folder containing the ABC4Trust-UProve service. For convenience, you could copy this folder's content to `/usr/lib/ABC4Trust-UProve` with the following command (on Unix):

```
sudo cp -r \

   Code/uprove-java-binding/ABC4Trust-
UProve_dotNET_WebServiceServer/ABC4Trust-
UProve/bin/Release \

   /usr/lib/ABC4Trust-UProve
```

Knowing the path to the service, you can perform (for example):

```
mvn test -DPathToUProveExe=/usr/lib/ABC4Trust-UProve | tee
~/output-sanitycheck.txt
```

Windows 7: `mvn clean test`

All components should finish with the status "SUCCESSFUL".
Note: The tests can take between 45 minutes and several hours to finish (because the generation of the cryptographic key pairs is non-deterministic).

9.3.1.2.2 *Building and starting the Identity Broker*

The Identity Broker is a GUI front end that allows to host all the user's privacy components in the cloud. Thus, you only need to perform the following installation steps if you want to make use of such cloud-based deployment.

6. Change directory to the Identity Broker's one (i.e. `Code/java-ui-cloud`)
7. Build and start the web-based graphical user interface.
   - on linux: run the the script named `stop-build-deploy.sh`, i.e.:

     ```
     ./stop-build-deploy.sh   2>&1   |   tee   ~/output-
     identityagent.txt
     ```

   - on windows: run the commands in sequence, equivalent to the script above, i.e.:

     ```
     set MAVEN_OPTS=-Xmx1024m -Xms256m -
     XX:MaxPermSize=512m

     mvn -pl eu.abc4trust.ri.ui.user.product jetty:stop

     mvn clean install
     ```

     ```
     mvn -pl eu.abc4trust.ri.ui.user.product jetty:deploy-
     war
     ```

After this, the Identity Broker will be reachable on http://localhost:9093/user-ui (or instead *localhost*, a remote host's domain name/IP-address).

Make sure that the Identity Broker is running in a separate, own terminal session (e.g., use `screen -S identityAgent`). This terminal session must stay alive as long as the identity broker is running.

For performing all the cryptographic operations, the Identity Broker GUI is dependent on a locally running user service. In the following, we describe how this user service can be built, started, and configured.

9.3.1.2.3 *Building and starting the local user service*

8. Make sure, that the core is built, according to the description above.
9. Build the self-contained ABCE-services. To do this, first switch to the directory `Code/core-abce/abce-services` and then run the build script with the following command

```
./compileServices.sh
```

10. Start the local user service (while still being in the directory `Code/core-abce/abce-services`) with

```
java -jar ./target/selfcontained-user-service.war 9200
2>&1 | tee ~/output-userservice.txt
```

Be aware that the terminal session for running the user service must stay alive as long as the identity broker is running (that is, as long as the identity broker may request the handling of cryptographic operations from the user service). Thus, instead of opening multiple terminal sessions, you may want to consider running above commands in a separate terminal session. A separate terminal session can, e.g., be started with `screen -S userService`.

11. Start a new terminal session, go to the directory `Code/java-ui-cloud/`, and configure the local user service by executing the following command

```
./userservice-democonfig.sh
```

9.3.1.3 ***Eclipse Import***

You can optionally use Maven to generate Eclipse project files (.project):

```
mvn eclise:eclipse
```

The projects are generated in the individual module folders and can be imported in Eclipse as existing projects.

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

Core Components

The core components of the Privacy GE represent a software library that application developers can use to implement the various components in a privacy-preserving authentication system. Hence, the core components of the Privacy GE itself do not represent a readily usable (privacy-preserving authentication) system and a testing is therefore only possible at the level of the software library itself.

For example, if a party wants to act as an issuer of privacy-preserving credentials by means of the Privacy GE, first, this party has to perform certain integration and configuration steps of the Privacy GE and can only afterwards start up a RESTful web service that handles issuing requests. The Privacy GE comes with automated tests that, just for the purpose of testing, perform these mentioned steps and then temporarily start up a RESTful web service to check whether the library is working properly.

To perform this test and see whether it is successful, programmers can run the following command in the Code/core-abce directory:

```
mvn test -DPathToUProveExe=/usr/lib/ABC4Trust-UProve | tee
~/output-sanitycheck.txt
```

This command will perform the mentioned tests. All components should finish with the status "SUCCESSFUL".
Note: The tests can take between 45 minutes and several hours to finish (because the generation of the cryptographic key pairs is non-deterministic).

Identity Broker
In addition to the core components, we also provide an Identity Broker component that depends on and builds upon the core components. The Identity Broker allows to run the user-side components of the Privacy GE in the cloud.

For an initial test whether the above installation and configuration steps have been performed successfully, check that

- http://localhost:9093/user-ui (if applicable, substitute *localhost* with the correct IP address or domain name of the host where the Identity Broker is running)

is accessible with your Web browser. For detailed tests on whether everything is running properly, we refer to the Privacy GE's Unit Testing Plan and Report.

9.4.2 List of Running Processes

Identity Broker
If the identity broker and the user service have been stared according to the previously described steps, then the process snapshot for the current user (which can be obtained with the command `ps -u`) should look similar to the following one:

```
USER        PID %CPU %MEM    VSZ   RSS TTY      STAT START
TIME COMMAND
root      28599  0.0  0.0   4400   612 pts/3    S+   15:58
0:00 /bin/sh ./stop-build-deploy.sh
root      28600  0.0  0.0   7188   640 pts/3    S+   15:58
0:00 tee /root/output-identityagent.txt
root      28654 13.2  3.7 2355004 151928 pts/5  Sl+  15:59
0:09 java -jar ./target/selfcontained-user-service.war 9200
```

```
root      28655  0.0  0.0   7188   648 pts/5    S+   15:59
0:00 tee /root/output-userservice.txt
```

Core Components
N/A (Not Applicable): it is up to the application developers which processes should be running after they have integrated the Privacy GE into their application. Although the mentioned automated tests will start up a number of processes for testing purposes, these processes are automatically terminated after testing and thus do not remain running.

9.4.3 Network interfaces Up & Open

Identity Broker
To operate the identity broker, the following ports have to be opened for TCP communication on the hosting machine:

- 9093

Core Components
N/A (Not Applicable): although the mentioned automated tests do start up processes that listen at specific ports, these are just test ports used during automated testing and thus do not remain open.

9.4.4 Databases

N/A (Not Applicable): the Privacy GE does not use relational databases as data store. Thus there are no databases that can be checked for accepting queries.

9.5 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.

9.5.1 Resource availability

N/A (Not Applicable): given that the Privacy GE is just a software library and not a full-fledged readily usable system, it depends on the application that uses this library which amount of system resources (RAM, hard disk space, etc.) is required.

9.5.2 Remote Service Access

N/A (Not Applicable): the Privacy GE is supposed to be a component of other enablers and it is therefore decided by the integrating enabler (and not the Privacy GE) what the connection parameters are.

9.5.3 Resource consumption

N/A (Not Applicable, see Resource Availability)

9.5.4 I/O flows

N/A (Not Applicable, see Resource availability)

9.6 Acknowledgements

This documentation was created by Alexandra Institute, Miracle, and IBM Research - Zurich with support from the ABC4Trust, FI-WARE, FutureID, and PrimeLife EU FP7 projects.

Note that the Privacy-Preserving Attribute-Based Credential Engine is open-source software whose primary home is on Github. As we seek to jointly maintain both the documentation in FI-WARE and the one on Github, you will find an similar version of this document in our Github repository.

However, please be aware that the reference documentation is the one on Github.

10 Data Handling GE - Installation and Administration Guide

You can find the content of this chapter as well in the wiki of fi-ware.

10.1 Introduction

This guide describes the installation and administration information for a web application bundle that implements the Data Handling GE Open Specification. It is composed by two distinguished web applications:

- the actual Data Handling GE implementation (which is indeed the **PPL** software), that exposes the GE ReSTful API (later referenced with the bundle name "ppl-rest.war");
- a use case application that demonstrates the usage of the ReSTful API: this is a remote storage-like application with delegation and obligation support (later referenced with the bundle name "ppl-webapp.war").
- An Identity-based encryption system (IBE), including third party named PKG (Private Key Generator) exposes the certificate generation, encryption/decryption Restful API (later referenced with the bundle name "ppl-pkg.war");
- A use case application that demonstrates the usage of the ppl-pkg.war Restful API: using the certificate generation and decryption Restful services (later referenced with the bundle name "ppl-pkg-webapp.war").

While the actual implementation of Data Handling GE Open Specification is represented by the first bundle, the second is described here, as it facilitates a comprehension of the API and of its applicability scenarios.

The remainder part of the document contains the instruction for deploying the PPL bundle (i.e., PPL and its use case application).

10.2 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\software

10.2.1 Database

We used the MySQL 5.1.43 database (available at http://www.mysql.com). In order to manage this database we used the EasyPHP 5.3.5 package (available at http://www.easyphp.org/) that includes the Apache HTTP web server, PHP, MySQL and phpMyAdmin.

10.2.2 Apache Tomcat Sever

In order to deploy the PPL engine and the use case we application locally you need a host server. We recommend an Apache Tomcat 7 server. [1]

Moreover, it can be necessary to specify the following parameters, to ensure enough memory available for Apache Tomcat operations. The parameters are:

```
-Xms512m -Xmx1024m -XX:MaxPermSize=256m
```

Different Operating Systems require different configurations for setting such parameters in the Tomcat start-up scripts. For instance, on Win32/64 systems, it can be sufficient to set the following options in:

```
%CATALINA_HOME%\bin\setenv.bat
```

10.2.3 Java VM

The executable for the PPL engine requires Java version 1.7 and the Java Cryptography Extension (JCE). It is possible to find software on this webpage, were also installation instructions are available.

10.2.4 Web Browser

The demo is web-based and the user interfaces should be displayed in a browser. We recommend to use Google Chrome.

10.3 PPL Configuration

A number of configuration files are necessary for the correct deployment of the PPL bundle. **They must be verified at each installation process and in case changed to reflect the actual system settings**. For each of the software composing the Data Handling GE, detailed installation information can be found in the following subsections.

- For PPL, configuration files can be found in ppl-rest.war
  - /WEB-INF/classes/log4j.properties : configuration of the logging system (Apache Log4J)
  - /WEB-INF/classes/system.properties : internal configuration of PPL
  - /WEB-INF/classes/META-INF/persistence.xml : connection information to the DB (**IMPORTANT: MUST BE CHANGED**)
- For the use case application, configuration files can be found in in ppl-webapp.war:
  - /WEB-INF/classes/log4j.properties : configuration of the logging system (Apache Log4J)
  - /WEB-INF/classes/webapp.properties : internal configuration of the use case webapp
- For PKG, configuration files can be found in ppl-pkg.war
  - /WEB-INF/classes/log4j.properties : configuration of the logging system (Apache Log4J)
  - /WEB-INF/classes/system.properties : internal configuration of PPL
- For the PKG use case application, configuration files can be found in in ppl-pkg-webapp.war:
  - /WEB-INF/classes/log4j.properties : configuration of the logging system (Apache Log4J)

- /WEB-INF/classes/system.properties : internal configuration of the use case webapp

10.3.1 Apache Log4J configuration

Both bundle components use the same configuration for Log4J. An exhaustive guide about its configuration can be found at the following link. Administrators are requested to configure Log4J according to their will and requirements, according to the provided reference. A suggested configuration for debugging purposes is the following:

```
# Set root logger level to DEBUG and its only appender to A1.
#log4j.rootLogger=DEBUG, A1
#log4j.appender.A1=org.apache.log4j.ConsoleAppender
#log4j.appender.A1.layout=org.apache.log4j.PatternLayout
#log4j.appender.A1.layout.ConversionPattern=%-4r [%t] %-5p %c %x
- %m%n

log4j.rootCategory=INFO, CONSOLE
log4j.appender.CONSOLE=org.apache.log4j.ConsoleAppender
log4j.appender.CONSOLE.Threshold=INFO
log4j.appender.CONSOLE.layout=org.apache.log4j.PatternLayout
log4j.appender.CONSOLE.layout.ConversionPattern=%-4r %-5p %c %3x
- %m%n
```

10.3.2 PPL (rest) configuration

Besides the mentioned Log4J configuration file, PPL is configured by two configuration files, *persistence.xml* and *system.properties* .

Persistence.xml aims at providing the Java application server with connection information for the DB to be used. It defines configuration directives for Hibernate, an Object-Relational framework. It specifies, for the databases in use by PPL (primelifePU, primelifePU-dc, primelifePU-3p), their respective connection parameters. **It is extremely important that a system administrator updates the connection parameters FOR EACH DATABASES in *persistence.xml*, otherwise PPL would not start-up.** An administrator should update the values of *hibernate.connection.** properties, according to the existing DBMS and network configuration. Other settings can be essentially ignored. A relevant excerpt of file contents is the following (*please modify all persistence-unit sections for primelifePU, primelifePU-dc, primelifePU-3p*):

```
    <persistence-unit name="primelifePU">
        <properties>
```

```
            <property name="hibernate.dialect"
value="org.hibernate.dialect.MySQLDialect" />

            <property name="hibernate.connection.driver_class"
value="com.mysql.jdbc.Driver" />

            <property name="hibernate.connection.username"
value="'''<change here>'''" />

            <property name="hibernate.connection.password"
value="'''<change here>'''" />

            <property name="hibernate.connection.url"
value="jdbc:mysql://localhost:3306/ppl" /> '''<change here if
needed>'''


            <!-- <property name="hibernate.hbm2ddl.auto"
value="create-drop"/> -->

            <!-- <property name="hibernate.hbm2ddl.auto"
value="create"/> -->

            <property name="hibernate.max_fetch_depth" value="3" />

            <property name="hibernate.show.sql" value="false" />

            <property name="hibernate.archive.autodetection"
value="hbm"/>


            <property name="cache.provider_class"
value="org.hibernate.cache.NoCacheProvider"/>

        </properties>

    </persistence-unit>
```

system.properties contains a number of configuration parameters for PPL operations. In particular, it defines connection details for sending notification emails (as part of the obligation framework), as well as for specifying the file name for enabling the accountability logging functionalities. Please see Data Handling GE Open Specification for more information on the latter aspect.
The file contents are:

```
# Email configuration

mail.smtp.host=<SMTP server>

mail.smtp.port=25

from=<mail account that will result as mail sender address for
notifications>
```

```
# Log EventTrigger actions
DS_log=EventTriggerDS_log.txt
DC_log=EventTriggerDC_log.txt
3P_log=EventTrigger3P_log.txt
```

10.3.3 PKG Configuration

System.properties contains a number of configuration parameter for the PKG operations. In particular, it defines security parameters details for initialization. Please be careful changing the value or the encryption and decryption process could be corrupt and may raise exception.

The file contents are:

```
# WARNING
# This file contains sensible data specially
# The Security configuration
# Only change if you know what you're doing
#Security configuration

C=475486472035438767215055519281714630206494610347947559169770049273
815204951654968982587270188186193977516111376162931348463892393920873414354066714909613883183

Q=124390545951226430551283785674456862191313220729067630208670134263
0837662207

L=382252902259852731869589271622333305314135625920121236176091685664980085954596375211 2

M=516343242707202790968923653883591927307989569128746037331706646250807756043

X=143504601412543216482412960842037842878173880218035750300580205127838052127450465561787894049604100646800206144278708127568354646255370111881762687639996072436 0

Y=285594270277992134203453674260356474159550990982420139203968960165715856402996210728070282560033315669052030597982616061934680299655770773896149817373229442740 8
```

10.3.4 Use case web application (ppl-webapp) configuration

Besides the mentioned Log4J configuration file, PPL is configured by another resource, *system.properties* .

system.properties essentially contains one setting, which is the URL of the PPL ReSTful interface. It is up to system administrators to update this setting, according to network and service configuration. The file content is:

```
com.sap.research.primelife.webapp.property.ppl.url=http://<URL where the PPL-rest bundle can be invoked>/ppl-rest/

pkg.url=http://localhost:8080/ppl-pkg/
```

10.3.5 Use case PKG web application configuration

Besides the mentioned Log4J configuration file, PPL is configured by another resource, *system.properties*.

system.properties essentially contains one setting, which is the URL of the PPL ReSTful interface. It is up to system administrators to update this setting, according to network and service configuration. The file content is:

```
ppl-pkg=http://<URL where the PPL-pkg bundle can be invoked>/ppl-pkg/

ppl-rest=http://<URL where the PPL-rest bundle can be invoked>/ppl-rest/
```

10.4 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
- ppl-3p

1- Make sure MySQL server is running

2- Initialize the Database, by executing the sql scritpt db-tools/create.sql or db-tools/ Do not forget to configure the correct connection information in db-tools/META-INF/persistence.xml Then run db-tools/reset_db.bat (Win) or db-tools/reset_db.sh (UNIX)

3- Deploy the PPl-Engine (ppl-rest.war) on a Tomcat Server

4- Deploy the ppl-webapp.war into your local Tomcat Server 5- Deploy the PKG-Engine (ppl-pkg.war) on a Tomcat Server 6- Deploy the ppl-pkg-webapp.war into your local Tomcat Server

Once the DB populated, and the servers deployed, the following endpoints should be responding:

- Use case web interface http://localhost:8080/ppl-webapp/
- REST endpoint: http://localhost:8080/ppl-rest/
- Use case web interface for the pkg http://localhost:8080/ppl-pkg-webapp/
- PKG REST endpoint: http://localhost:8080/ppl-pkg/

10.5 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.

10.5.1 End to End testing

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

- Data subject: http://localhost:8080/ppl-webapp/ an HTML frontend with the FIWARE logo is properly displayed
    - Another test can be to navigate with a web browser to URL: http://<server URL>:8080/ppl-rest/application.wadl
- PKG webapp: http://localhost:8080/ppl-pkg-webapp/ an HTML frontend with the FIWARE name and a menu in the top left corner is displayed.
    - Another test can be to navigate with a web browser to URL: http://<server URL>:8080/ppl-pkg/application.wadl

The URLs http://<server URL>:8080/ppl-rest/application.wadl and http://<server URL>:8080/ppl-pkg/application.wadl return a WADL file dynamically generated by the respective web applications, if and only if their REST services are properly initialized. Therefore, browsing such URLs is a convenient end-to-end testing means.

Successful execution log using cURL for URL http://<server URL>:8080/ppl-pkg/application.wadl (with browsers like Google Chrome, the XML structure of the WADL file is returned):

```
> GET http://<server url>:8080/ppl-pkg/application.wadl HTTP/1.1
> User-Agent: curl/7.30.0
> Host: <server url>:8080
```

```
> Accept: */*
> Proxy-Connection: Keep-Alive
>
< HTTP/1.1 200 OK
* Server Apache-Coyote/1.1 is not blacklisted
< Server: Apache-Coyote/1.1
< Last-modified: Mon, 28 Apr 2014 10:45:29 CEST
< Content-Type: application/vnd.sun.wadl+xml
< Content-Length: 2215
< Date: Mon, 28 Apr 2014 08:45:40 GMT
< Proxy-Connection: Keep-Alive
< Connection: Keep-Alive
< Age: 0
<
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<application xmlns="http://wadl.dev.java.net/2009/02">
    <doc xmlns:jersey="http://jersey.java.net/"
jersey:generatedBy="Jersey: 1.12
 02/15/2012 04:51 PM"/>
    <grammars/>
    <resources base="http://<server url>:8080/ppl-pkg/">
        <resource path="cipher/">
            <resource path="/{mode}">
                <param xmlns:xs="http://www.w3.org/2001/XMLSchema"
name="mode" s
tyle="template" type="xs:string"/>
                <method id="decrypt" name="POST">
                    <request>
                        <representation mediaType="multipart/form-
data"/>
                    </request>
                    <response>
                        <representation mediaType="multipart/form-
data"/>
```

```
                    </response>
                </method>
            </resource>
        </resource>
        <resource path="controller/">
            <method id="getSysParam" name="POST">
                <response>
                    <representation mediaType="multipart/form-
data"/>
                </response>
            </method>
        </resource>
        <resource path="/certificate">
            <resource path="/request">
                <method id="certificateRequest" name="POST">
                    <request>
                        <representation mediaType="multipart/form-
data"/>
                    </request>
                    <response>
                        <representation mediaType="text/plain"/>
                    </response>
                </method>
            </resource>
        </resource>
        <resource path="/keypair">
            <resource path="/{keyType}">
                <param xmlns:xs="http://www.w3.org/2001/XMLSchema"
name="keyType
" style="template" type="xs:string"/>
                <method id="getPublicKey" name="POST">
                    <request>
                        <representation mediaType="multipart/form-
data"/>
```

```
                    </request>
                    <response>
                        <representation
mediaType="application/json"/>
                    </response>
                </method>
            </resource>
        </resource>
    </resources>
</application>
* Closing Connection #0
```

10.5.2 List of Running Processes

The list of running process needed for the execution of PPL on a Win32/64 box are:

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

On *nix systems, this list might vary slightly, but essentially it should comprise:

- java (for Apache Tomcat, or similar entries for the Java application server in use)
- mysqld
- apache (according to system configuration)

10.5.3 Network interfaces Up & Open

PPL is a web application bundle that is run by an application server, and uses a DBMS. Therefore, it requires:

1. A running Java application server (Apache Tomcat), whose HTTP and HTTPS ports must be up and reachable; default ports are 8080 and 8443, for production deployments, they should be made available, preferably via a proxy, respectively on ports 80 and 443;
2. A running MySQL database, active on its default port 3306, which should be protected by remote accesses for security reasons;
3. Additionally, it is recommended to allow PPL to access an SMTP mail server, without authentication, in order to permit the sending of email notification on resource events: connection parameters can be specified using the *system.properties* configuration file of ppl-rest.war, as explained in previous section "PPL Configuration".

10.5.4 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 It the three DB are initiated as described in the previous section

10.6 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.

10.6.1 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, 2000 MB RAM and 500 MB of application-dedicated storage.

10.6.2 Remote Service Access

PPL does not require access to any external remote service. However, it can make use of an internal SMTP server, in order to send email notifications on resource usage to external users. It is up to the system administrator to support this functionality, and to configure it as explained in the previous section "PPL Configuration".

10.6.3 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

10.6.4 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.

11 Access Control - Installation and Administration Guide

You can find the content of this chapter as well in the wiki of fi-ware.

11.1 Introduction

In the rest of document, the Thales Access Control GE implementation will be simply referred to as 'Authorization Server'.

11.2 System Requirements

- Operating System: Linux (Ubuntu Server 12.04/14.04 LTS recommended) i686/x86_64
- Java Platform: JDK 7 (JRE is not enough), Glassfish Server 3.x, OpenDJ 2.5-Xpress1 or later
- RAM: 4GB min
- CPU: 2.6 GHz min
- Disk space: 10 GB min
- An Identity Management instance must be reachable in the environment if IdM integration is required, especially for OAuth.

11.3 Installation

11.3.1 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 stratum if using NTP. This is critical for checking the validity of OAuth access tokens generated by the IdM GE.

11.3.2 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. **This procedure requires using a JDK 1.7 or later.** (For the sake of simplicity, we do not use a subordinate CA, although you should for production, see keytool 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-cert.pem
```

11.3.3 Application Server Setup (Glassfish)

Glassfish requires Java. If you don't have JDK 7 installed, read the next section; if you have, skip it.

11.3.3.1 *JDK (Java) Intallation*

You can refer to the JDK installation instructions from Ubuntu Java documentation. For Ubuntu < 12.04 LTS, we strongly recommend upgrading to 12.04 LTS before you go on. The installation steps are summarized below:

```
$ sudo add-apt-repository ppa:webupd8team/java
```

If you have 'add-apt-repository: command not found' error, try again after doing the following:

```
$ sudo apt-get install python-software-properties
```

Then go on:

```
$ sudo apt-get update

$ sudo apt-get install oracle-java7-installer
```

11.3.3.2 *Server Installation*

Download *glassfish-unix.sh* file from Glassfish website, and run it with sudo:

```
$ sudo wget
http://download.java.net/glassfish/3.1.2.2/release/glassfish-
3.1.2.2-unix.sh

$ sudo sh glassfish-3.1.2.2-unix.sh
```

If you have errors running the script above, in particular because you don't have a graphical environment (no X server), try the -s option to run in silent mode instead. In this case, you need to supply a glassfish install template as input. Use the template given in example 1-2 of Glassfish Installation Guide - Performing a Silent Mode Installation, or the one below, and fill in the properties with your desired settings, save it to file say *glassfish-install-template.txt* and run the script:

```
$ cat glassfish-install-template.txt

Domain.Configuration.ADMIN_PASSWORD=changeit

Domain.Configuration.ADMIN_PASSWORD_REENTER=changeit

Domain.Configuration.ADMIN_PORT=4848

Domain.Configuration.ADMIN_USER=admin

Domain.Configuration.DOMAIN_NAME=domain1

Domain.Configuration.HTTP_PORT=8080
```

```
InstallHome.directory.INSTALL_HOME=/opt/glassfish3

UpdateTool.Configuration.ALLOW_UPDATE_CHECK=false

UpdateTool.Configuration.BOOTSTRAP_UPDATETOOL=false

UpdateTool.Configuration.PROXY_HOST=

UpdateTool.Configuration.PROXY_PORT=


$ sudo sh glassfish-3.1.2.2-unix.sh -s -a glassfish-install-
template.txt
```

When running the above command on x86_64 platforms, you may encounter one of the following errors:

- SEVERE INTERNAL ERROR: /usr/lib/jvm/java-7-oracle/jre/lib/amd64/xawt/libmawt.so: libXrender.so.1: cannot open shared object file: No such file or directory
- SEVERE INTERNAL ERROR: /usr/lib/jvm/java-7-oracle/jre/lib/amd64/xawt/libmawt.so: libXtst.so.6: cannot open shared object file: No such file or directory
- SEVERE INTERNAL ERROR: /usr/lib/jvm/java-7-oracle/jre/lib/amd64/xawt/libmawt.so: libXi.so.6: cannot open shared object file: No such file or directory

In this case, do the following before you try again:

```
$ sudo apt-get install libxrender1 libxtst6 libxi6
```

Start Glassfish:

```
$ sudo /opt/glassfish3/bin/asadmin start-domain
```

11.3.3.3 *Server Preliminary Setup*

If you have multiple JDKs installed besides JDK 1.7, set the Glassfish domain's `java_home` property to the specific Oracle JDK 1.7 for the domain (more info):

```
$ asadmin set "server.java-config.java-
home=/usr/lib/jvm/jdk1.7.0_25"
```

Run the following command on the Glassfish domain (more info):

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

Restart Glassfish to apply changes above (if you have multiple domains, you have to specify the domain where the GE will be deployed, otherwise the default 'domain1' is used like in this example):

```
$ asadmin restart-domain
```

11.3.3.4 *Server Security Setup*

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

- Remove Glassfish homepage or replace it with unsensitive information: (*GLASSFISH_DOMAIN_DIR/docroot/index.html*). Any information identifying specifically your running software (e.g. the server product such as Glassfish, version, etc.) or hardware is usually considered sensitive in this context.
- Enable secure admin, ONLY if you want remote access to the Glassfish admin GUI:

```
$ asadmin enable-secure-admin
```

- Disable XPoweredBy header on all HTTP listeners:

  To find the available HTTP listeners to modify:

```
$ asadmin get server.network-config.protocols.protocol.*
```

  Disable the header for each one:

```
$ asadmin set server.network-config.protocols.protocol.http-
listener-1.http.xpowered-by=false

$ asadmin set server.network-config.protocols.protocol.http-
listener-2.http.xpowered-by=false
```

- Change the Server header:

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

- ... or remove the Server header:

```
$ asadmin create-jvm-options -Dproduct.name=""
```

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

```
$ asadmin stop-domain

$ asadmin change-master-password domain1
```

11.3.3.5 *Server Certificate Setup*

By default, Glassfish uses a self-signed server certificate for SSL connections, generated during the installation. You can stay with it and skip the rest of this section, but only for testing. For production, it is strongly recommended to generate your own CA-signed certificate, as detailed below.

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, first make the CSR:

```
$ keytool -certreq -keystore taz-server-keystore.jks -alias taz-server -file taz-server.csr
```

4. Make the CA sign the server certificate (change CA keystore password accordingly, and the validity period according to your security requirements):

```
$ keytool -gencert -keystore taz-ca-keystore.jks -alias taz-ca -infile taz-server.csr -validity 365 -ext ku:c=dig,keyEncipherment -rfc -outfile taz-server-cert.pem
```

5. Import the CA certificate into the server truststore

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

6. Import the server key and certificate signed by the CA into the server keystore:

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

7. 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
```

8. Copy *taz-server-truststore.jks* file to the Glassfish domain's config directory (e.g. ${GLASSFISH_DOMAIN_DIR}/config)

9. Set the server's truststore password to the same master password set with *change-master-password* command in the previous section (DO NOT SET THE TRUSTSTORE PASSWORD WITH JVM OPTION 'javax.net.ssl.trustStorePassword' AS THE MASTER PASSWORD WILL BE USED INSTEAD ANYWAY):

```
$ keytool -storepasswd -keystore taz-server-truststore.jks
```

10. Modify the Glassfish domain's keystore in the Glassfish domain's *config* directory (e.g. *${GLASSFISH_DOMAIN_DIR}/config*), after making a backup copy of it:

```
$ cp keystore.jks{,.orig}
```

11. Replace the key for alias 's1as' in the keystore with the one from your keystore 'taz-server-keystore.jks' created in the previous section (use the same password set with *change-master-password* command in the previous section for the -destkeypass argument):

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

11.3.3.6 *Server TLS Setup*

1. Set the server's truststore path (DO NOT SET THE TRUSTSTORE PASSWORD WITH JVM OPTION 'javax.net.ssl.trustStore.password' AS THE MASTER PASSWORD WILL BE USED INSTEAD ANYWAY):

```
$ asadmin start-domain

$ asadmin delete-jvm-options -
Djavax.net.ssl.trustStore=\${com.sun.aas.instanceRoot}/config/c
acerts.jks

$ asadmin create-jvm-options -
Djavax.net.ssl.trustStore=\${com.sun.aas.instanceRoot}/config/t
az-server-truststore.jks
```

2. Configure the group assignment of clients authenticated by certificates:

```
$ asadmin set server.security-service.auth-
realm.certificate.property.assign-groups=authenticated
```

3. You may want to change the SSL port (default: 8181), for instance to the standard port (443):

```
$ asadmin set server.network-config.network-listeners.network-
listener.http-listener-2.port=443
```

4. Make sure the server hostname matches the CN of the certificate subject DN.

5. Restart Glassfish to apply above changes (*asadmin restart-domain* cannot be used anymore after changing the master password):

```
$ asadmin stop-domain

$ asadmin start-domain
```

11.3.3.7 *Server Performance Tuning*

You can find most recommended performance tuning tips and other useful tips for putting Glassfish in production, on the blog [Putting GlassFish v3 in Production: Essential Surviving Guide](). In brief:

```
$ asadmin delete-jvm-options -client:-Xmx512m

$ asadmin create-jvm-options \\-server:-XX\\:+AggressiveHeap:-
Xmx1400m:-Xms1400m
```

If the host has multiple CPUs (multi-core), run the following command, replacing number '4' in this example with the actual number of CPUs:

```
$ asadmin create-jvm-options -XX\\:ParallelGCThreads=4:-
XX\\:+UseParallelOldGC
```

If the JVM is 64-bit:

```
$ asadmin create-jvm-options -XX\\:+UseCompressedOops
```

Restart the Glassfish domain to apply changes. For complete tuning, refer to the *Performance Tuning Guide* of your respective Glassfish version. Some useful tips to keep in mind for performance tuning:

- The number of CPUs on Linux is obtained with 'nproc' command;
- Check whether the JVM is 64-bit: *java -version*;
- asadmin create-jvm-options command may take multiple options separated by ':' (colon) character;
- If an option to asadmin create-jvm-options already includes a ':' character (e.g. -XX:...), escape it with double '\' (backslash);
- If you get the error "The stack size specified is too small, specify at least..." at JVM startup, increase the value of -Xss option as told (or more).
- If you get the error "Too small initial heap for new size specified" at JVM startup, increase the value of -Xmx/-Xms

11.3.4 User and Role Management Setup

This section is about configuring an LDAP directory for storing authorized users and user-role assignments required to access the Authorization Server interface.

11.3.4.1 *User Directory Setup*

Although OpenDJ is given as example in some of the following steps, you can use any other LDAP v3 directory to store user roles. The only specific requirement is that the directory server must provide a mechanism so that, after adding a member (object class 'person') - to a group entry (object class 'groupOfNames'), a specific attribute in the member's entry is automatically filled in with the new group membership. This is the equivalent to OpenDJ 'isMemberOf' virtual attribute (or OpenLDAP 'memberOf'). See section *18.1 Virtual Attributes - isMemberOf* in OpenDJ's *Administration Guide*.

The following steps assumed you have already a LDAP directory server deployed in your environment. For OpenDJ from ForgeRock, refer to the Installation Guide. For Ubuntu, you can use the deb package available on the OpenDJ Downloads page and follow the installation steps. In particular, do not forget to run this after installing the package:

```
$ sudo /opt/opendj/setup --cli
$ sudo /opt/opendj/bin/status
```

1. If you do not have a base DN setup in the directory, create one, such as 'dc=example,dc=com'. For OpenDJ, this can be done when running the setup command above, or with the Control Panel.
2. Create a user branch and a role branch (organizationalUnit objectClass), such as 'ou=users' and 'ou=roles', if you do not have one or the other already, in the above directory base

(*ldapmodify* is provided in */opt/opendj/bin* if using OpenDJ, or in ldap-utils package for most Linux distributions):

```
$ cat add-branches.ldif

dn: ou=users,dc=example,dc=com

objectClass: organizationalUnit

ou: users

dn: ou=roles,dc=example,dc=com

objectClass: organizationalUnit

ou: roles


$ ldapmodify --defaultAdd --port 389 --bindDN "cn=Directory
Manager" --filename add-branches.ldif
```

3. Configure the directory server to enforce user mail uniqueness on the user branch. Example for OpenDJ:

```
$ 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
```

4. If your directory server is not OpenDJ, configure your directory server to enforce an equivalent to OpenDJ 'isMemberOf' mechanism.

11.3.4.2 ***User Registration***

For using the API provided by the application, users must connect via HTTPS and authenticate with client certificates signed by the CA mentioned in #Certificate Authority Setup. In addition, users must be registered in the user database set up in #User and Role Management Setup. Therefore, to allow a given user to access the API, as an administrator, you must first make his/her signed certificate and register his/her in the user database (LDAP directory in this case).

11.3.4.2.1 *Client Certificate Setup*

This section is about creating a client certificate signed by the trusted CA for clients - application users - authenticating to the Authorization Server. Example with keytool:

1. On the client side, generate the client keypair (the dname argument does not matter as it will be overriden 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=Self Signed"
```

2. On the client side, generate the client CSR and send it to the CA:

```
$ keytool -certreq -keystore johndoe-taz-client-keystore.jks -
alias johndoe-taz-client -outfile john.doe-taz-client.csr
```

3. On the CA/Authorization Server host used in section 'Certificate Authority Setup', make the client certificate signed by the CA using the previously 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 -gencert -keystore taz-ca-keystore.jks -alias taz-ca
-infile john.doe-taz-client.csr -dname "CN=$(/usr/bin/uuid -v1),
OU=Thales AuthzForce Users, O=FIWARE" -validity 365 -rfc -ext
ku:c=digitalSignature,nonRepudiation,keyEncipherment,dataEnciph
erment,keyAgreement -ext eku:c=clientAuth -outfile johndoe-taz-
client-cert.pem
```

4. On the client side, 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
```

5. On the client side, 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
```

6. On the client side, 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
```

11.3.4.2.2 *Registration in the User Database*

This section is about adding a new application user in the user directory set up in #User and Role Management Setup. Create the 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 (UUID), as defined in previous section #Client Certificate Setup. For example with user *John Doe*:

```
$ cat add-user.ldif

dn: uid=28789263-ed04-4d08-bb42-
35ea4950fc79,ou=users,dc=example,dc=com

objectClass: inetOrgPerson

uid: 28789263-ed04-4d08-bb42-35ea4950fc79

cn: John Doe

sn: Doe

givenName: John mail: john.doe@example.com
```

```
$ ldapmodify --defaultAdd --port 389 --bindDN "cn=Directory Manager"
--filename add-user.ldif
```

11.3.4.3 *Superadmin Role Assignment*

```
The Superadmin is responsible for managing the application. In
particular, a user must have this role to create or remove
policy administration domains - referred simply as domains in
the rest of the document - for your end-users (policy
administrators and Policy Enforcement Points). Therefore,
before you can do anything with the application's API, you must
first assign the Superadmin role to one or more users
registered by following the previous section #User
Registration, and that you want to make Superadmins. For
example, to assign some user ''John Doe'' the role
''Superadmin'', first create entry ''cn=Superadmin'' with
objectClass ''groupOfNames'' under the ''ou=roles'' branch
created in section 'User Role Management Setup', if not already
created (first ''Superadmin'' user). Then add a 'member'
attribute with the above user DN as value to this group entry
to finalize the user-role assignment. You can do both at the
same time:$ cat add-role.ldif

dn: cn=Superadmin,ou=roles,dc=example,dc=com

objectClass: groupOfNames

cn: Superadmin

member: uid=28789263-ed04-4d08-bb42-
35ea4950fc79,ou=users,dc=example,dc=com


$ ldapmodify --defaultAdd --port 389 --bindDN "cn=Directory
Manager" --filename add-role.ldif
```

The 'member' attribute is multi-valued so that you can have more than one member in such group entries obviously, and therefore one or more Superadmins. Besides this role, you can assign other roles which are restricted to a specific domain. Therefore, we will explain this after deploying the application in the next section #Authorization Server Application Setup, and addressing the creation of new domains with the API in #Domain Creation.

11.3.5 Authorization Server Application Setup

1. Download the latest release package from the FI-WARE PPP Restricted Project website, package "Security-AccessControlGE", and unzip it (XXX=version):

2. unzip authzforce-fiware-dist-XXX-bin.zip

You will get a directory called *authzforce-fiware-dist-XXX*.

If unzip command is not found, install it and try again:

```
$ apt-get install unzip
```

11.3.5.1 *Configuration*

1. Deploy the configuration files to Glassfish domain config folder as follows:

```
$ cp -a authzforce-fiware-dist-XXX/conf
${GLASSFISH_DOMAIN_DIR}/config/authzforce
```

2. In file *core-pdp.xml*, inside the *attributeFinderModule* element with attribute *class="com.thalesgroup.authzforce.finder.LdapAttributeFinderModule"*, replace *url* value with the URL of the LDAP directory server used in section 'User Role Management Setup'; *bindDN* and *bindPassword* with the credentials of a directory server admin account that has read permissions on the user and role branch set up in section *User Role Management Setup*; and finally, replace all occurrences of *dc=example,dc=com* with the base DN you have configured in 'User Role Management Setup'.

11.3.5.2 *Web Application Deployment*

1. Deploy the war file from the release package to Glassfish server as a web application (XXX = release version).

```
$ asadmin deploy --name authzforce-XXX --contextroot authzforce
--enabled=true authzforce-fiware-dist-XXX/authzforce-fiware-
webapp-XXX.war
```

   If you used *enabled=false* because you wanted to check/modify the webapp content before starting the webapp, you can then enable with command: *$ asadmin enable authzforce*

2. The application requires SSL client certificate authentication. So make sure the web deployment descriptor (web.xml) is properly configured to use realm-name 'certificate' to authentication. Then edit the Certificate Realm in glassfish administration console and set the 'Assign-Groups' property to 'authenticated'. If you want to specify another property value, make sure it matches the 'group-name' of the 'security-role-mapping' defined for role 'authenticated' in the webapp file 'glassfish-web.xml' (same folder as web.xml).

11.3.6 Policy Domain Management

11.3.6.1 *The Concept of Policy Domain*

The application is multi-tenant, i.e. it allows users or organizations to work on authorization policies in complete isolation from each other. In this document, we use the term "domain" instead of "tenant". In this context, a policy domain consists of:

- Various metadata about the domain: ID, name, description;
- The root XACML <PolicySet>;

- Optional <PolicySet>s that may be referenced in <Policy(Set)Reference>s by the aforementioned root <PolicySet>;
- Attribute Finders configuration; attribute finders resolve attributes from other sources thant the XACML <Request>, e.g. external databases, external services, server environment, etc.

The reasons for creating different domains:

- Users or organizations do not want others to access their data, or even be impacted by others working on the same application.
- The same user or organization may want to work on different domains for different use cases; e.g. work with one policyset for production environment, another for testing, another for a specific use case project, etc.

11.3.6.2 *Domain Creation*

You create a domain by doing a HTTPS POST request with XML payload to URL https://${SERVER_NAME}:${PORT}/authzforce/domains. Replace ${SERVER_NAME} and ${PORT} with your server hostname and port for HTTPS. You can do it with *curl* tool, with the certificate of a registered user - say *John Doe* - having the Superadmin role. But first, you need to get your private key in PEM format. One way consists to get the keystore in PKCS12 format first, as explained at the end of #Client Certificate Setup. Then convert to PEM format with *openssl* tool as follows:

```
$ openssl pkcs12 -in john.doe-taz-client-keystore.p12 -clcerts -out
john.doe-taz-client-keystore.pem
```

Now, you are able to make the domain creation request with curl, using the client keystore *john.doe-taz-client-keystore.pem* in PEM format resulting from previous command, and the CA certificate *taz-ca-cert.pem*:

```
$ curl --verbose --cacert taz-ca-cert.pem --cert john.doe-taz-
client-keystore.pem --request POST --header "Content-Type:
application/xml;charset=UTF-8" \

 -d "<?xml version="1.0" encoding="UTF-8"?><taz:properties
xmlns:taz="http://thalesgroup.com/authz/model"><name>MyDomain</name>
<description>This is my domain.</description></taz:properties>" \

 --header "Accept: application/xml"
https://${SERVER_NAME}:${PORT}/authzforce/domains



 ...

 Enter PEM pass phrase:

 ...

> POST /authzforce/domains HTTP/1.1

> User-Agent: curl/7.22.0 (x86_64-pc-linux-gnu) libcurl/7.22.0
OpenSSL/1.0.1 zlib/1.2.3.4 libidn/1.23 librtmp/2.3

> Host: az.testbed.fi-ware.eu
```

```
> Content-Type: application/xml;charset=UTF-8
> Accept: application/xml
> Content-Length: 227
>
...
< HTTP/1.1 200 OK
< Server: Authorization System
< Date: Mon, 04 Aug 2014 13:00:12 GMT
< Content-Type: application/xml
< Transfer-Encoding: chunked
<
<?xml version="1.0" encoding="UTF-8" standalone="yes"?><link
xmlns="http://www.w3.org/2005/Atom" rel="item" href="0ae7f48f-1f13-
11e3-a300-eb6797612f3f"/>
```

The *href* value in the response above gives you the domain ID (in the form of a UUID), that you will now use for assigning user roles on the domain.

11.3.6.3 *Domain Role Assignment*

Domain users must have one of the two roles may be assigned only at the domain level:

- Domain_Admin;
- PEP.

If this is the first time you assign a role on this domain, you have to create an OU for the domain in the directory (objectClass *organizationalUnit*), under the role branch *ou=roles* created in section 'User Role Management Setup'. The *ou* value is the domain ID seen as *href* value in the previous section, e.g. *ou=0ae7f48f-1f13-11e3-a300-eb6797612f3f* for domain 0ae7f48f-1f13-11e3-a300-eb6797612f3f. Then create a group entry for each role (objectClass *groupOfNames*) under this OU entry (similarly to Superadmin role in #Superadmin Role Assignment). For example:

```
$ cat add-domain-roles.ldif
dn: ou=0ae7f48f-1f13-11e3-a300-
eb6797612f3f,ou=roles,dc=example,dc=com
objectClass: organizationalUnit
ou: 0ae7f48f-1f13-11e3-a300-eb6797612f3f
description: My Domain 1
dn: cn=Domain_Admin,ou=0ae7f48f-1f13-11e3-a300-
eb6797612f3f,ou=roles,dc=example,dc=com
objectClass: groupOfNames
```

```
cn: Domain_Admin

dn: cn=PEP,ou=0ae7f48f-1f13-11e3-a300-
eb6797612f3f,ou=roles,dc=example,dc=com

objectClass: groupOfNames

cn: PEP


$ ldapmodify --defaultAdd --port 389 --bindDN "cn=Directory
Manager" --filename add-domain-roles.ldif
```

Finally, assign the users to one of the roles by adding a 'member' attribute with the user DN (in the user branch) as value (similarly to Superadmin role in #Superadmin Role Assignment). For example, assigning *Domain_Admin* role on the domain '0ae7f48f-1f13-11e3-a300-eb6797612f3f' to a user with ID '67c277db-911a-4ed2-bfda-3931bfdfb7e0' and another with ID 'b667f018-b52b-42a3-8c3e-863e31c54c58,ou=users,dc=fiware,dc=eu', then another with ID 'e1d25e2f-3421-447c-9819-3563db6cd179' to role *PEP*:

```
$ cat assign-user-roles.ldif

dn: cn=Domain_Admin,ou=0ae7f48f-1f13-11e3-a300-
eb6797612f3f,ou=roles,dc=example,dc=com

changetype: modify

add: member

member: uid=67c277db-911a-4ed2-bfda-
3931bfdfb7e0,ou=users,dc=example,dc=com

member: uid=b667f018-b52b-42a3-8c3e-
863e31c54c58,ou=users,dc=example,dc=com

dn: cn=PEP,ou=0ae7f48f-1f13-11e3-a300-
eb6797612f3f,ou=roles,dc=example,dc=com

changetype: modify

add: member

member: uid=e1d25e2f-3421-447c-9819-
3563db6cd179,ou=users,dc=example,dc=com


$ ldapmodify --defaultAdd --port 389 --bindDN "cn=Directory
Manager" --filename assign-user-roles.ldif
```

Once users have the proper roles, they can refer to the User and Programmers Guide to learn how to manage and use their domain with the API.

11.3.6.4 *Domain Removal*

You remove a domain by doing a HTTPS DELETE request with XML payload to URL https://${SERVER_NAME}:${PORT}/authzforce/domains/{domain_ID}. For example with *curl* tool,

using the client keystore *john.doe-taz-client-keystore.pem (PEM format) of some registered user* John Doe *with Superadmin role:*

```
$ curl --verbose --cacert taz-ca-cert.pem --cert john.doe-taz-
client-keystore.pem --request DELETE --header "Content-Type:
application/xml;charset=UTF-8" \

 --header "Accept: application/xml"
https://${SERVER_NAME}:${PORT}/authzforce/domains/0ae7f48f-1f13-
11e3-a300-eb6797612f3f
```

11.4 Administration

11.4.1 Glassfish Webapp Administration

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

11.4.2 Application-Specific Administration

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

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

11.5 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.

11.5.1 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}/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>
```

If you are testing this right after a fresh installation, the 'resources' tag will be empty since there are no domains yet created, and therefore none to be listed. 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
```

Example on Linux:

```
$ wget --no-check-certificate
https://az.example.com/authzforce/services
```

11.5.2 List of Running Processes

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

11.5.3 Network interfaces Up & Open

- TCP 22
- TCP 80
- TCP 389

- TCP 443
- TCP 4848
- TCP 8443

11.5.4 Databases

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

11.6 Diagnosis Procedures

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

```
HTTP GET https://${host}:${port}/${contextRoot}/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 `${GLASSFISH_DOMAIN_DIR}/logs/server.log`.
3. If you get an error 401 or 403, see the Authorization Server logs as described in section *Application-Specific Administration*. In particular, if you see JNDI LDAP connection error, check that the OpenDJ server status by running OpenDJ control panel locally. Check the Superadmin role is still properly assigned for your certificate as done during installation. For more information, see OpenDJ directoy server troubleshooting.
4. If you get a Connection Refused, check that the Glassfish server is running by checking the server status in Glassfish admin WUI (Web User Interface)
    1. If you are unable to connect to the admin WUI, check that the Glassfish domain is started locally with command `asadmin list-domains`. If not, start the domain: `asadmin start-domain`
    2. If the command fails, check the domain logs in `${GLASSFISH_DOMAIN_DIR}/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 adress resolution, routing, etc.

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

11.6.1 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

11.6.2 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.

11.6.3 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.

11.6.4 I/O flows

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

12 Context-based Security and Compliance - Installation and Administration Guide

You can find the content of this chapter as well in the wiki of fi-ware.

12.1 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 select the appropriate solution for his/her concrete context and requirements and to manage those security solutions with a user interface that makes easy their deployment and monitoring.

12.2 System Requirements

12.2.1 Operating Systems

The current version supports MS Window O.S. and GNU/Linux, providing one binary for each platform.

12.2.2 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.

12.2.3 Database

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

12.2.4 Application Server

The PRRS has been tested with Apache Tomcat 7.0.39, not only for the PRRS Client Dashboard but also for the deployment of the available FI-WARE optional security enablers.

12.3 Installation

The PRRS has been developed using OSGI technology, so the bundles that implements the G.E. needs to be included in an OGSi container. We are using Apache Karaf as a lightweight OSGi container to provide a standalone distribution of this Context-based security & compliance GE. The Apache Karaf version 2.2.9 has been used for distributing the binaries.

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.

1. Install the software. Download the binaries (in this case for Linux) from https://forge.fi-ware.org/frs/download.php/1220/PRRS-Karaf-3.3.3.tar.gz (Access restricted to PPP members. Please

check http://catalogue.fi-ware.org/enablers/content-based-security-cbs/terms-and-conditions for further details)
Once we have unzip the file **PRRS-Karaf-3.3.3.zip**, we will have all the required files for Karaf and PRRS in a directory (e.g. D:\PRRS_v3.3.3). From now on we will refer to this root directory like **{PRRS_Karaf}**.

2. Database installation.

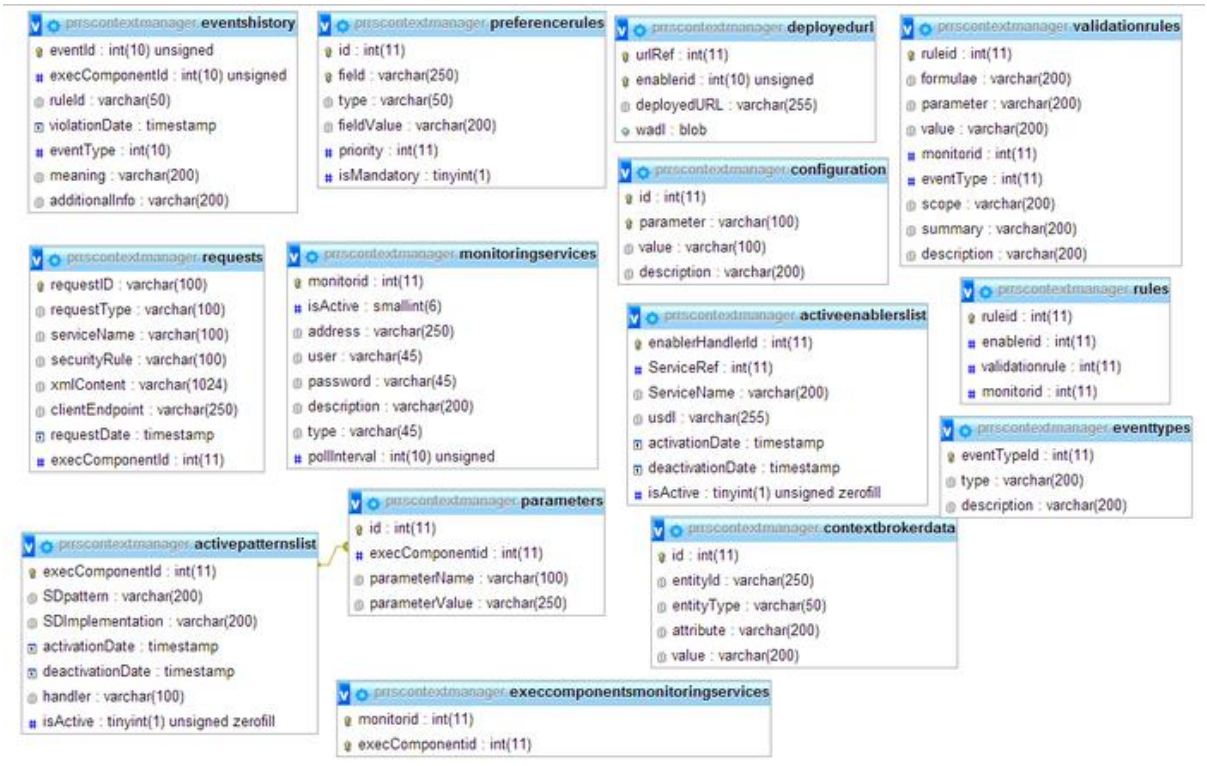
Two databases are used in the PRRS Context-based Security & Compliance GE:

- PRRS Context Manager: It records the data related to the context such as the PRRS configuration, active services, event history, user preferences, user configuration, etc. It will be used by the PRRS to select the most suitable security solution for a received request and to apply the reaction mechanisms with context changes.
- Security Repository: It is the local repository used as cache for storing information about 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.

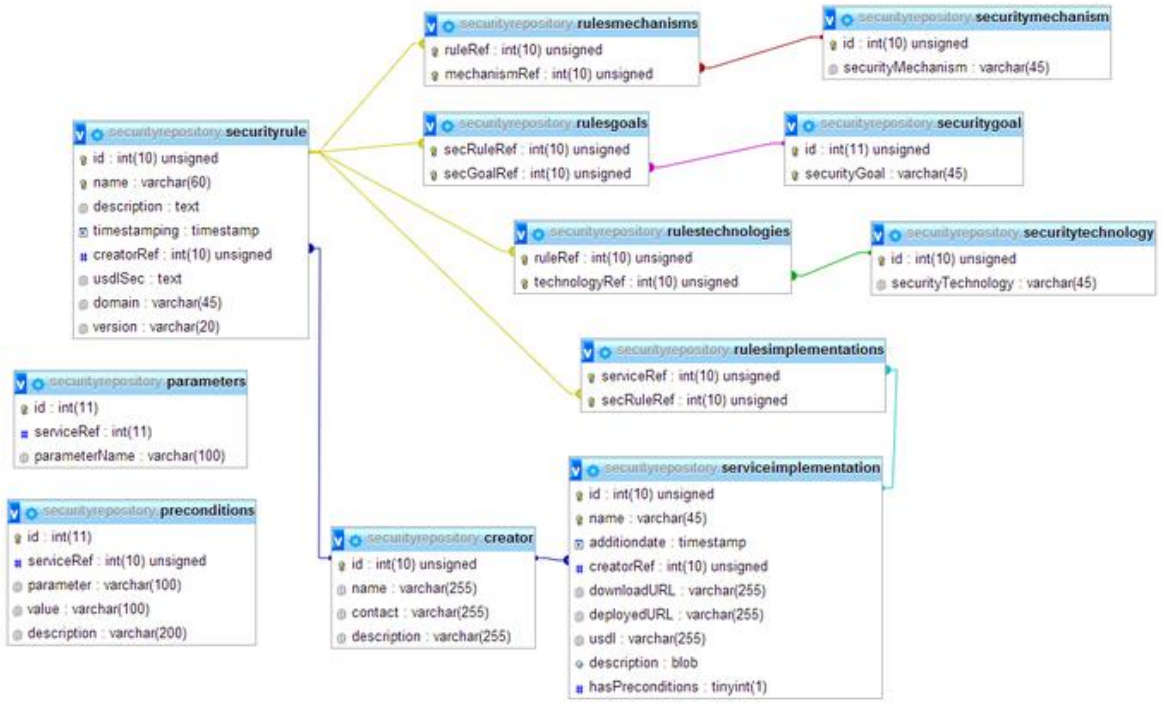
In order to create those databases and the required tables for the PRRS, the following scripts (located under the folder**{PRRS_Karaf}/PRRS_installation/databases**) must be executed in the MySQL Server:

```
mysql> source prrs_context_manager.sql
mysql> source security_repository.sql
mysql> source prrs_context_manager_init_data.sql
mysql> source security_repository_init_data.sql
```

In the following screenshots it is shown the schemas created:



PRRS ContextManager Database Schema



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 prrsContextManager and SecurityRepository tables. For example:

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

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

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

3. PRRS configuration.

The configuration files for the PRRS are located in **{PRRS_Karaf}/PRRS_installation/conf**. It is necessary to review the following parameters included in the file **sdmanager.conf**:

- Access to the databases:

The information about how to access to the databases must be configured in the following parameters:

```
database.user = fiware

database.password = fiware

database.url = jdbc:mysql://localhost:3306
```

```
database.contextdb.url =
jdbc:mysql://localhost:3306/prrscontextmanager

database.secrepo.url =
jdbc:mysql://localhost:3306/securityrepository
```

- Access to the FI-WARE Marketplace:

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/her. Consequently, the following parameters must be configured:

```
marketplace.url=http://[SERVER_URL]/FiwareMarketplace/v1

collection.name=offering/store/security/offerings

marketplace.user=testUser1

marketplace.passwd=testUser1
```

NOTE: In this example, we are assuming all the available security services are offered in a FI-WARE Marketplace instance available at the endpoint http://[SERVER_URL]/FiwareMarketplace/v1 and in a store called *security* (*collection.name=offering/store/security/offerings*). These parameters must be configured by the user with his/her credentials to access to the Marketplace.

- Access to the FI-WARE Security Monitoring:

The PRRS can generate events to be sent through Syslog to the FI-WARE Security Monitoring GE in order to monitor this enabler and the services deployed. By default, this feature is disabled (*activateLogger_FI-WARE=0*) but it can be activated and configured with the following parameters:

```
activateLogger_FI-WARE = 0

syslogServer = 192.168.215.118

syslogPort = 514
```

- Information for the deployment:

The folder where the Apache Tomcat is installed in the end-user environment must be included in the configuration file as well as the context.xml file used:

```
deploymentFolder = D:/env/projectFIWARE/apache-tomcat-7.0.39/webapps

contextXMLFile = D:/env/projectFIWARE/apache-tomcat-
7.0.39/conf/context.xml
```

Besides, for deployment testing purposes the downloading of software can be deactivated through a variable included in the configuration file. In that case, a file with the name *tmpToDeploy.zip* located in the *{PRRS_Karaf}/PRRS_installation* folder can be used to contain the software to be deployed:

```
```

```
activateDownload = 0
```

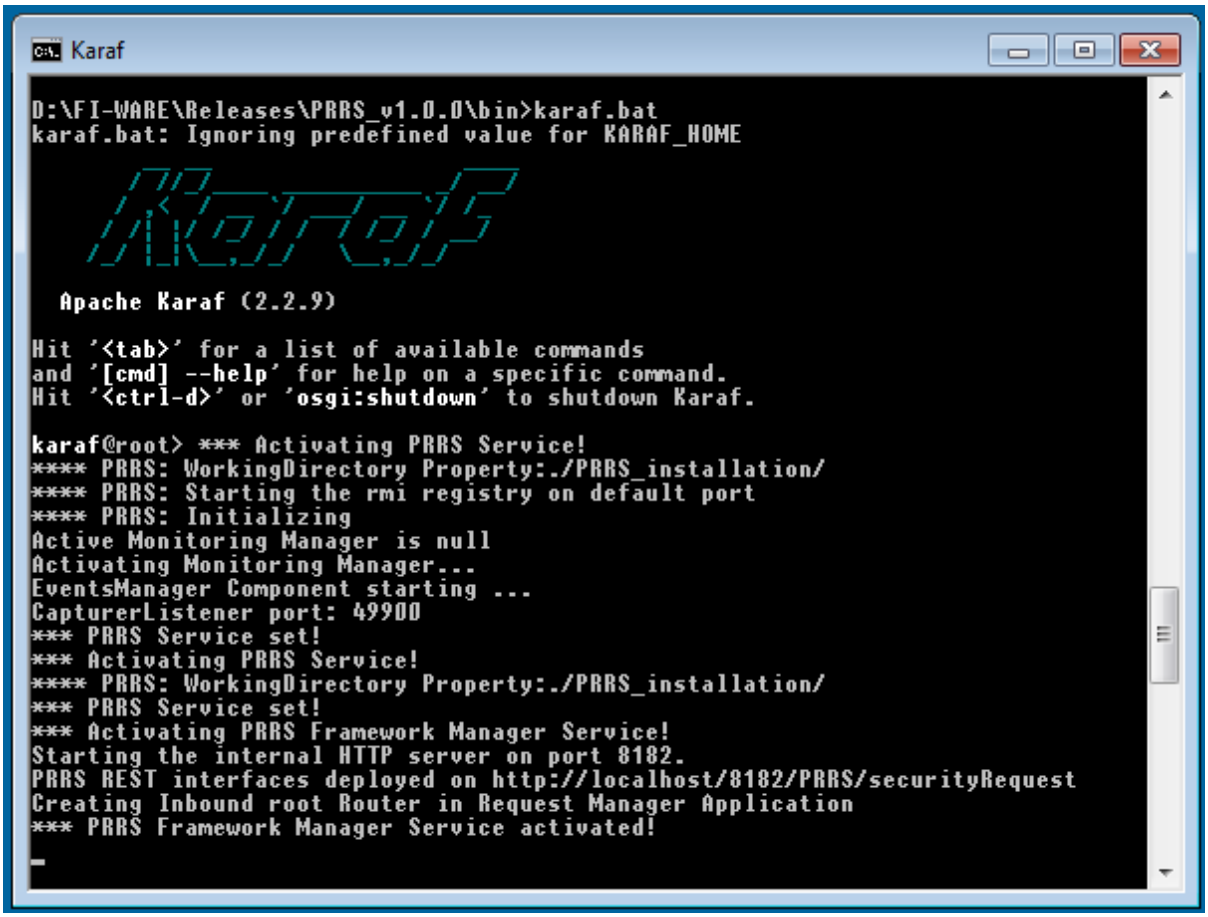
- Information for the context-monitor:

The PRRS includes the implementation of a context-monitor whose start and poll time can be configured:

```
contextmonitor.startTime = 1000

contextmonitor.pollTime = 1000
```

4. Start Karaf.

To run the PRRS it is only required to execute the 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:



Running PRRS with Karaf

To run Karaf as a daemon in Linux so it can be administrated easily without connecting to the console, the following commands must be executed (it's not recommended for testing since is easier to see the logs in interactive mode):

```
karaf@root> features:install wrapper
```

```
karaf@root> wrapper:install -s AUTO_START -n KARAF -d Karaf -D
"Karaf Service”


karaf@root> osgi:shutdown
```

The way the service is installed depends on the Linux distribution, but in the case of Ubuntu/Debian system, in order to install the service the following link must be created:

```
# ln -s /home/fiware/apache-karaf-3.2.3/bin/KARAF-service
/etc/init.d/
```

And to start the service automatically when the machine is rebooted:

```
# update-rc.d KARAF-service defaults
```

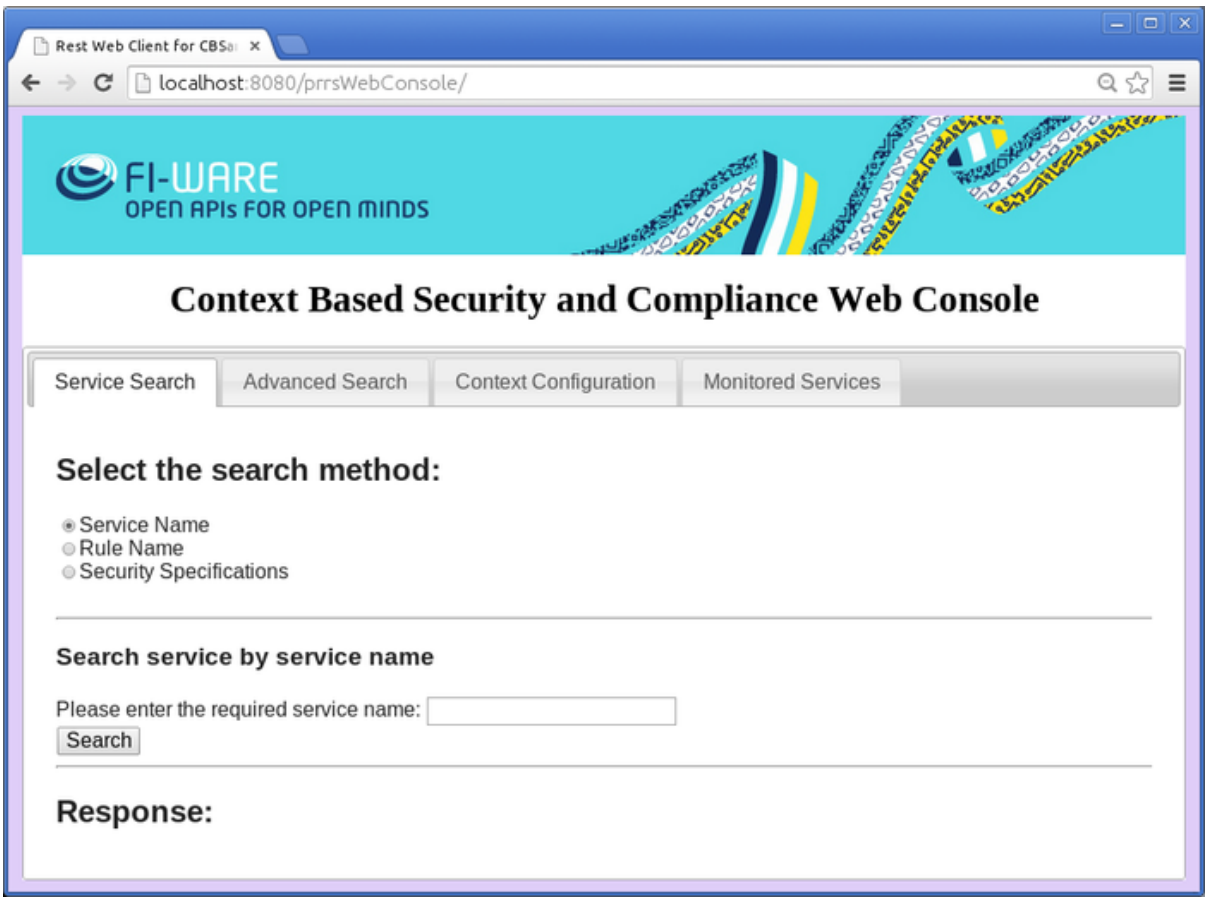
Finally, to start the service Karaf with the PRRS:

```
#/etc/init.d/KARAF-service start
```

5. Install the PRRS Client Dashboard.

The PRRS Client Dashboard to send security requests to the PRRS Framework, configure the end-user context and monitor the security solutions deployed is provided as the **prrsWebConsole.war** file (downloadable at https://forge.fi-ware.org/frs/download.php/1182/prrsWebConsole.war). Copy this war file in your Apache Tomcat installation under the webapps folder. Then, it is only necessary to restart the Apache Tomcat to deploy it. To install it on other Java Application Servers, please refer to their specific application server guidelines.

The dashboard will be accessible in: http://localhost:8080/prrsWebConsole



PRRS Client Dashboard

12.4 Administration

The PRRS is provided within an Apache Karaf container which has been installed as a service called *KARAF-service*. Consequently, it is automatically started/stopped with the Karaf.

- To start the service:

```
$ /etc/init.d/KARAF-service start
```

- To stop the service:

```
$ /etc/init.d/KARAF-service stop
```

Since the Karaf has been started as a service, it is possible to connect to the Karaf console in remote through SSH with the following command (password by default: *karaf*):

```
# ssh -p 8101 karaf@127.0.0.1
```

NOTE: Substitute 127.0.0.1 by the ip where the service is running to access remotely

If you want to administrate locally the Karaf container instead of installing it like a service, the following commands can be used:

- To start the Karaf console, run the script *karaf* located in the folder *{PRRS_Karaf}/bin*:

```
# karaf
```

- Once logged in the console, to list the bundles active in Karaf the following command must be run in the Karaf console:

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

- To stop the Karaf:

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

And confirm with yes to shutdown the Karaf instance.

All the bundles required for the execution of the PRRS inside the Karaf are located in the folder *{PRRS_Karaf}/Bundles*. This installation includes two features defined by the following files:

- common_equinox.xml: it includes all the dependencies used by the PRRS bundles
- prrs.xml: it includes the specific PRRS bundles

The features included in the Karaf container can be checked once logged to the Karaf console with the following command:

```
karaf@root> features:listurl
 Loaded   URI
  true
mvn:org.apache.karaf.assemblies.features/enterprise/2.2.9/xml/featur
es
  true    file:Bundles/prrs.xml
  true
mvn:org.apache.karaf.assemblies.features/standard/2.2.9/xml/features
  true    file:Bundles/common_equinox.xml
```

To update the version of the PRRS bundles, the following commands must be executed inside the Karaf console once the new bundles have been updated in the folder *{PRRS_Karaf}/Bundles* and the files prrs.xml and common_equinox.xml have been updated with the bundles required for the new PRRS version:

```
karaf@root> features:uninstall prrs
karaf@root> features:install prrs
karaf@root> osgi:shutdown
```

And finally, restart the Karaf to deploy the new bundles:

```
$ /etc/init.d/KARAF-service restart
```

12.5 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.

12.5.1 End to End testing

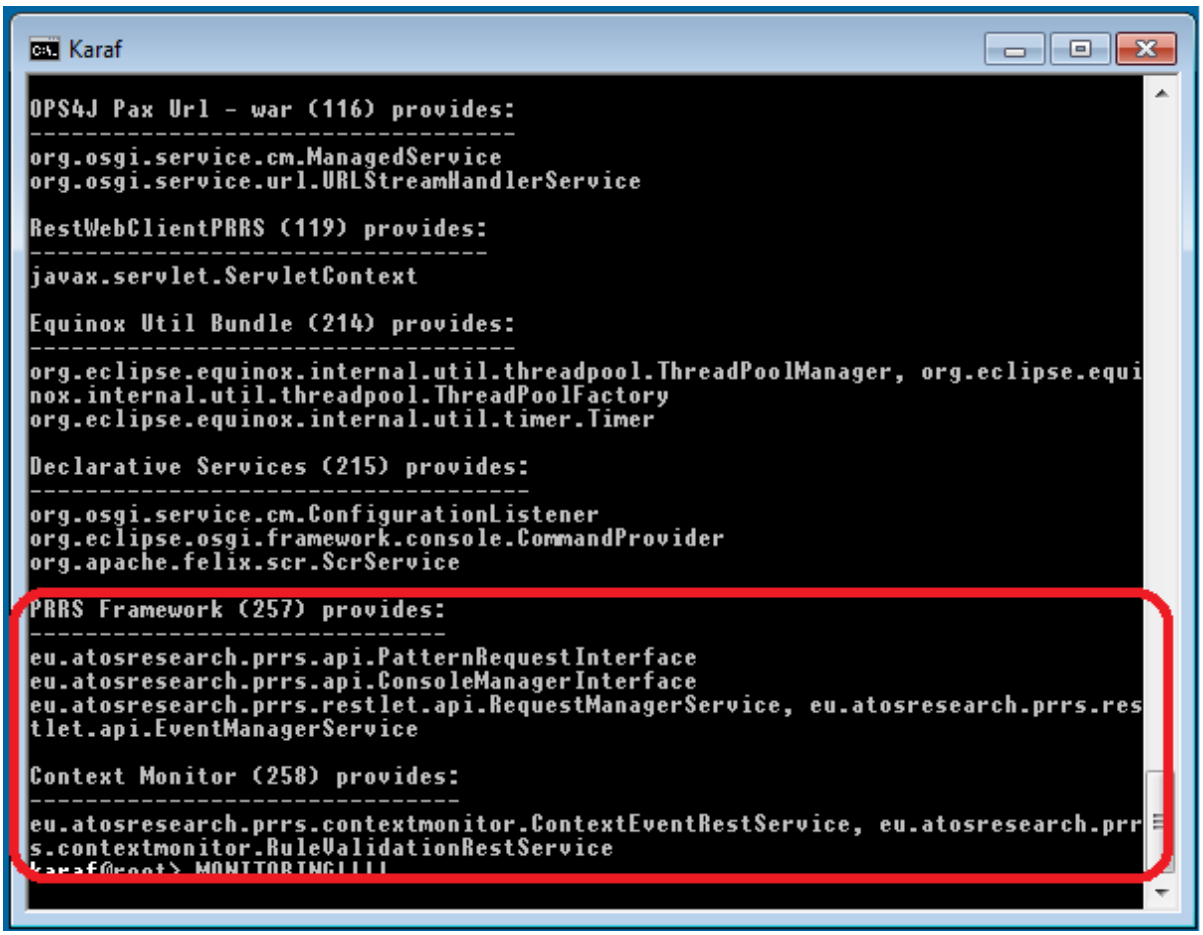
The quick testing to check that everything is up and running can be done logging to the Karaf console (password by default *karaf*):

```
# ssh -p 8101 karaf@127.0.0.1
```

and executing the following Karaf command:

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

A console like the following one should be shown:



PRRS Bundles Running

and the following bundles should be included in the response:

```
PRRS Framework (257) provides:

------------------------------

eu.atosresearch.prrs.api.PatternRequestInterface

eu.atosresearch.prrs.api.ConsoleManagerInterface

eu.atosresearch.prrs.restlet.api.RequestManagerService,
eu.atosresearch.prrs.restlet.api.EventManagerService


Context Monitor (258) provides:

------------------------------

eu.atosresearch.prrs.contextmonitor.ContextEventRestService,
eu.atosresearch.prrs.contextmonitor.RuleValidationRestService

karaf@root>
```

To verify that the PRRS Client Dashboard was correctly deployed on the application server, the URL http://localhost:8080/prrsWebConsole should be accessible and the main page with the *Context Based Security and Compliance Web Console* shown.

The user also should have access to the FI-WARE Marketplace and Repository. It can be verified with the following commands:

```
curl -v -H "Content-Type: application/xml" -u "testUser1:testUser1"
http://[SERVER_URL]/FiwareMarketplace/v1/offering/store/security/off
erings


curl -v -H "Content-Type: text/plain"
http://[SERVER_URL]/FiwareMarketplace/v1/collectionA/collectionB
```

Note that you need to use the address of your marketplace instance and the user that have access to this repository. For more information visit Marketplace - User and Programmer Guide

12.5.2 List of Running Processes

The following processes should be up and running:

- Apache Karaf (java.exe)
- Apache Tomcat (java.exe)
- MySQL server process(es)

12.5.3 Network interfaces Up & Open

The following ports shoud 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)
- 8183/tcp - PRRS Context-Manager (API REST to receive validation rules)
- 8101/tcp - Remote Karaf Console

12.5.4 Databases

The following databases must be up and running:

- prrsContextManager
- SecurityRepository

The connection details are included in the file *{PRRS_Karaf}/PRRS_installation/conf/sdmanager.conf*.

Test query:

```
D:\FI-WARE\Releases\PRRS_v1.0.0> mysql -u fiware -p
mysql> use prrscontextmanager;
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>
```

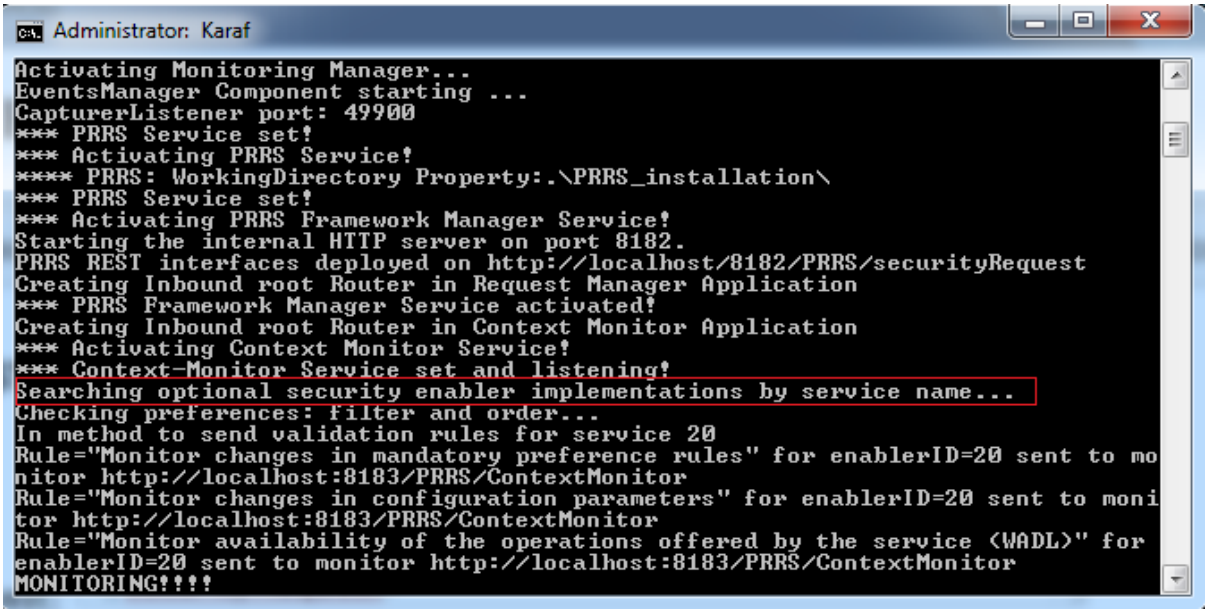
12.6 Diagnosis Procedures

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.

12.6.1 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 search.
- 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.



Searching Service

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.

12.6.2 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

12.6.3 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.

12.6.4 I/O flows

I/O flow on HTTPS (or HTTP) standard ports. Requests interactivity should be low because they represent 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.

12.7 References

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

13 DB Anonymizer GE - Installation and Administration Guide

You can find the content of this chapter as well in the wiki of fi-ware.

13.1 Introduction

DB Anonymizer is a Java Web Application, packaged in a WAR file. It uses a relational database (MySQL) 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 7 is supported)

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

13.2 Installation

13.2.1 Database

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

13.2.1.1 *DB Configuration*

DB Anonymizer has to connect to a MySQL database schema, 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 populated as follows:

```
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`)
);

CREATE TABLE `deepsearchsave` (
 `GID` bigint(20) NOT NULL,
 `columncounter` int NOT NULL,
 `columnrisk` float NOT NULL,
 PRIMARY KEY (`GID`,`columncounter`)
);

CREATE TABLE `bootstrapsave` (
 `GID` bigint(20) NOT NULL,
 `columncounter` int NOT NULL,
 `columnrisk` float NOT NULL,
 PRIMARY KEY (`GID`,`columncounter`)
);

CREATE TABLE `riskcolumnsave` (
 `GID` bigint(20) NOT NULL,
 `columncounter` int NOT NULL,
 `columnrisk` float NOT NULL,
  PRIMARY KEY (`GID`,`columncounter`)
);

CREATE TABLE `policies` (
 `GID` bigint(20) NOT NULL,
 `columncounter` int NOT NULL,
 `columnname` text NOT NULL,
 `columntype` text NOT NULL,
 `hidden` boolean NOT NULL,
 PRIMARY KEY (`GID`,`columncounter`)
```

```
);
```

13.2.2 Application Server

The application has been tested with Apache Tomcat 7.X. Compatibility with other application servers could be achieved, provided that administrators configure adequately JDBC connection pooling functionalities.

13.2.2.1 *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").

Alternatively, the application WAR package contains a context.xml file in path "/META-INF", containing the same information.

It should contain the following information:

```
<?xml version="1.0" encoding="UTF-8"?>
<Context>
    <!-- Specify a JDBC datasource -->

    <Resource name="jdbc/dbanonymizer" 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"
       removeAbandoned="true"
       removeAbandonedTimeout="60"
       logAbandoned="true"
       validationQuery="select 1"
/>

</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.

13.2.2.2 *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.

13.2.2.3 *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.

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 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>/services/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>/services/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 FIWARE.OpenSpecification.Security.DBAnonymizer in the API specification.

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

13.3.2 List of Running Processes

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

13.3.3 Network interfaces Up & Open

DB Anonymizer is a web application that is run by an application server, and uses a DBMS. Therefore, it requires:

- A running Java application server (Apache Tomcat), whose HTTP and HTTPS ports must be up and reachable; default ports are 8080 and 8443, for production deployments, they should be made available, preferably via a proxy, respectively on ports 80 and 443.
- A running MySQL database, active on its default port 3306, which should be protected by remote accesses for security reasons.

13.3.4 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 the configured database with the query:

```
SELECT * FROM <<YOUR DATABASE NAME>>.results;
```

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

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

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.

A rough resource estimation for a basic deployment could be the following: a (virtual/physical) machine has at least 2 CPUs, in order to manage effectively concurrent requests to different components (DB Anonymizer + RDBMS backend). Moreover, a system should also have at least 4 GB RAM, and 5 GB storage to be dedicated to DB Anonymizer + RDBMS needs.

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

13.4.2 Remote Service Access

DB Anonymizer does not require access to any external remote service, nor it offers a management interface. Any configuration has to be performed by altering configuration files on the file system that hosts the application server.

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, 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.

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 Content-based Security - Installation and Administration Guide

You can find the content of this chapter as well in the wiki of fi-ware.

14.1 Introduction

The Content Based Security (CBS) Optional Generic Enabler (OGE) is composed of three Java Web Applications: the CBS Producer, 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.

14.2 Installation

14.2.1 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] - this is used to make policy-based decisions on whether the user can access the protected data. Note that the CBS Broker contains a hardcoded policy decision point for use in the end to end test, so that the installation can be verified independently. However, the Access Control GE is for a full deployment.
- Identity Management GE [4] - this is used to manage the identities of users and their roles. Note that it is not needed to run the end to end test, so that the CBS installation can be verified independently. However, it is needed for a full deployment.

The Access Control GE needs to be configured with policies. An example policy is included in the CBS distribution for use in the End to End test and the CBS unit tests [5].

14.2.2 Build the CBS OGE

The CBS OGE can be downloaded from the FI-WARE PPP Restricted Project website [6] 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.

14.2.3 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.producer.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.

14.3 Administration

14.3.1 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[7].

14.3.2 Key Generation

Within the CBS GE, public keys are required for two uses: to establish SSL sessions and to apply/remove protection from content. When establishing an SSL session, the client (which in the CBS GE is itself a server) needs to establish that the server it is connecting to is trusted by checking the server's SSL certificate. In order to do this, the client must either hold a copy of the server's public key certificate or, if the certificate is signed by the CA, it must hold the CA's certificate. Trusted

public key certificates are stored in a truststore. Public keys are also used to protect content. For example, to apply a signature to an item, the Producer digitally signs the data using its private key to show that the message hasn't been tampered with and to prove who the message came from. The signature is verified by the Consumer using the Producer's public key. The private key is stored in a keystore and must not be shared.

A public and private key pair should be created for each of the three CBS services (the CBS Producer, the CBS Consumer and the CBS Broker). Each private key is stored in its own keystore and should not be shared. The public key certificate associated with each private key should be shared with the other CBS services. This is done by exporting the certificate from the keystore to a file and importing the certficate file into the truststores of the other services. The truststore should also contain any SSL certificates for the servers it uses. The table below summarises which keys are needed where.

CBS Service	**Keys in Keystore**	**Keys in Truststore**
Producer	Producer private key	Broker public key
Consumer	Consumer private key	Broker public key, Tomcat SSL certificate
Broker	Broker private key	Producer public key, Consumer public key, Access Control GE SSL certificate

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 for each service:

```
$ 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.

Each service also needs a truststore containing the certificates of the services it trusts. The certificates must first be exported from the keystore containing the keypair. The certficates are then imported into the truststore.

To create a new truststore, you need to create a keystore with a key and then delete the key by executing the following for each service:

```
$ keytool -genkey -alias <alias> -keystore \path\to\my\truststore
```

```
$ keytool -delete -alias <alias> -keystore \path\to\my\truststore
```

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

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

To import a certificate into the truststore execute the following from a terminal command line:

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

14.3.3 User Configuration

Before you deploy the CBS web applications to your Application Server, you must configure users in the `users.properties` file.

File Location: `<WAR_FILE>/WEB-INF/classes/users.properties` in the built WAR file or `<MVN_PROJECT>/src/main/resources/users.properties` in the unbuilt Maven project.

To add a user, append an entry in the following format:

```
username=password,role[,role][,enabled|disabled]
```

Available roles are:

- ROLE_USER - required to access the web interface and used with default policy to test that access to containers is denied
- ROLE_ADMIN - required to access admin interfaces
- ROLE_MANAGER - used in default policy to access containers with the label ‘sensitive’
- ROLE_BROKER - required to access the broker’s generateKey interface
- ROLE_CONSUMER - required to access the broker’s generateKey interface
- ROLE_TEST - required to access the end to end test web page

14.3.4 CBS Producer Configuration

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

File Location: `fiware.cbs.producer.service.webapp.war/WEB-INF/classes/application.properties` in the built WAR file or

`/fiware.cbs.producer.service.webapp/src/main/resources/application.properties` in the unbuilt Maven project.

The following properties are required for the CBS Producer to run.

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

keystore.pass=[KEYSTORE_PASSWORD]

key.alias=[ALIAS_OF_PRODUCER'S_KEY_IN_KEYSTORE]

key.pass=[KEY_PASSWORD]

truststore.file= path/to/truststore/file

truststore.pass=[TRUSTSTORE_PASSWORD]

broker.alias=[ALIAS_OF_BROKER'S_KEY_IN_TRUSTSTORE]

broker.url=[URL OF BROKER SERVICE]

encryption.algorithm.mode=[ALGORITHM_MODE]

encryption.algorithm.padding=[ALGORITHM_PADDING]

keywrap.algorithm.mode=[ALGORITHM_MODE]

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

The following properties are required for OAuth integration with the Identity Management GE.

```
oauth.client.id=[ID OF CLIENT REGISTERED AT IDM GE]
oauth.client.secret=[CLIENT SECRET REGISTERED AT IDM GE]
```

```
oauth.client.accesstokenuri=[IDM GE URL FOR REQUESTING OAUTH ACCESS TOKEN]

oauth.client.userauthorizationuri=[IDM GE URL FOR REQUESTING USER AUTHORISATION]

oauth.client.scope=[ID OF CBS SERVICE REGISTERED AT IDM GE]


oauth.serviceId=[URL OF CBS SERVICE REGISTERED AT IDM GE]


oauth2.scope=[ID OF CBS SERVICE REGISTERED AT IDM GE]

oauth2.tokenvalidationurl=[IDM GE URL FOR VALIDATING OAUTH TOKEN]
```

14.3.5 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 built WAR file or `/fiware.cbs.consumer.service.webapp/src/main/resources/application.properties` in the unbuilt Maven project.

The following properties must be configured to enable the CBS service to run.

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


keystore.pass=[KEYSTORE_PASSWORD]


key.alias=[ALIAS_OF_CONSUMER'S_KEY_IN_THE_KEYSTORE]


key.pass=[KEY_PASSWORD]


truststore.file= path/to/truststore/file


truststore.pass=[TRUSTSTORE_PASSWORD]


broker.url=[URL OF CBS BROKER SERVICE]
```

The following properties are required for OAuth integration with the Identity Management GE.

```
oauth.client.id=[ID OF CLIENT REGISTERED AT IDM GE]

oauth.client.secret=[CLIENT SECRET REGISTERED AT IDM GE]

oauth.client.accesstokenuri=[IDM GE URL FOR REQUESTING OAUTH ACCESS
TOKEN]

oauth.client.userauthorizationuri=[IDM GE URL FOR REQUESTING USER
AUTHORISATION]

oauth.client.scope=[ID OF CBS SERVICE REGISTERED AT IDM GE]


oauth2.scope=[ID OF CBS SERVICE REGISTERED AT IDM GE]

oauth2.tokenvalidationurl=[IDM GE URL FOR VALIDATING OAUTH TOKEN]
```

14.3.6 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 built 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]
```

14.4 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.

14.4.1 End to End testing

Note that this test only verifies that the CBS GE has been installed correctly and that the Producer, Consumer and Broker are able to operate. It does not check that it is integrated with the Identity Management GE or the Access Control GE. Use the Integration Tests[8] defined on the Testbed wiki to check that it is integrated correctly.

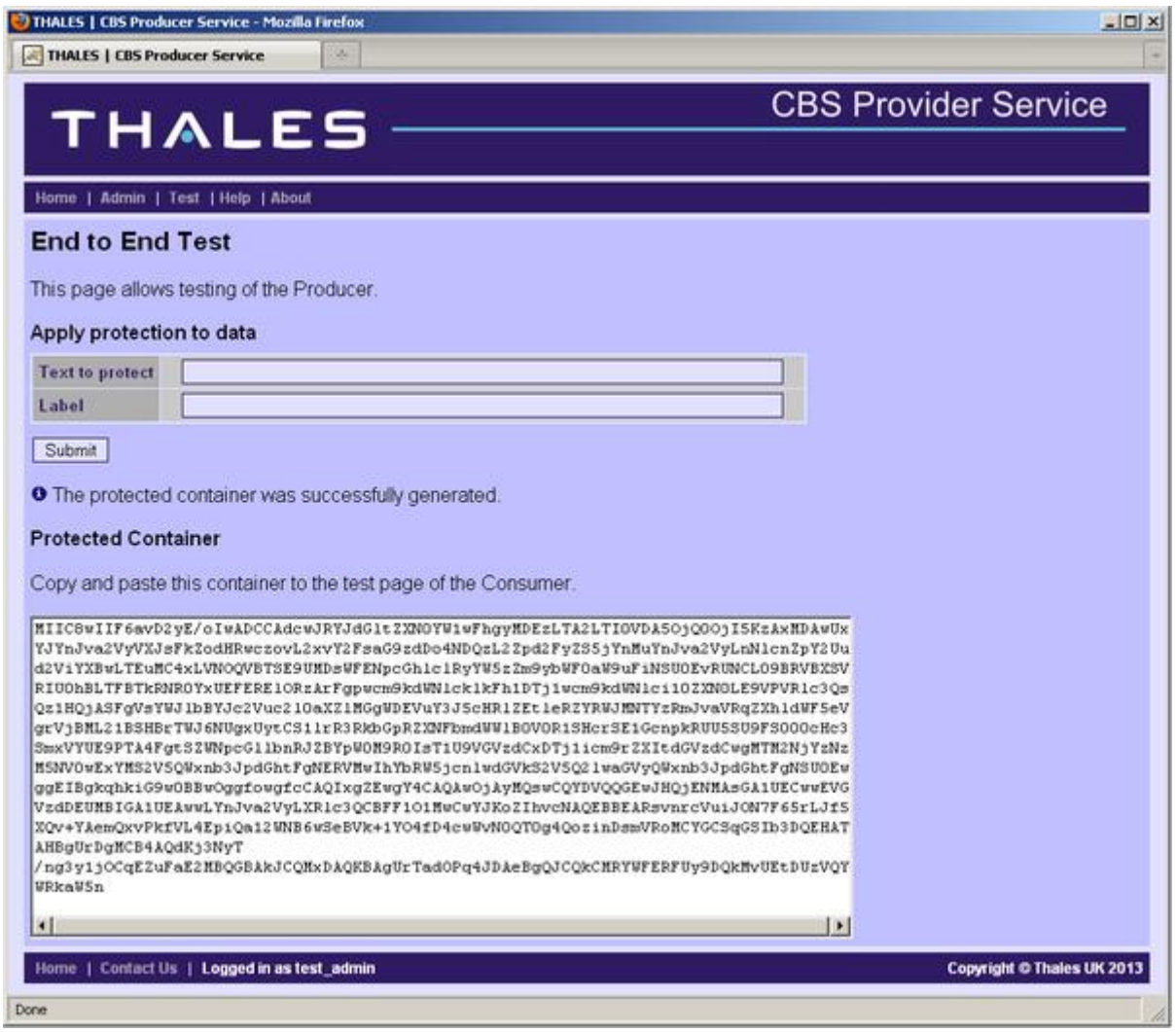
The first step is to verify that all three web applications are running:

- 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, use the web interface to check that you can apply protection to some text and then remove the protection.

1. In a browser, navigate to the test page on the CBS Producer: https://<host name>:<Tomcat SSL port>/fiware.cbs.producer.service.webapp/webui/test. Log in with username ‘test_manager’ and password ‘password’.
2. In the field ‘Text to protect’ type in some text to be protected
3. In the field ‘Label’, type ‘sensitive’.
4. In the field ‘Protection Mode’, select the ‘Encrypt Only’ radio button.
5. Click on ‘Submit’
6. If the Producer is correctly configured, you should see the message, “The protected container was successfully generated.” The base64 encoded protected container should be displayed, as shown in the figure below.



Producer test web page

7. Copy the protected container
8. In a browser, navigate to the test page on the CBS Consumer: https://<host name>:<Tomcat SSL port>/fiware.cbs.consumer.service.webapp/webui/test. Log in with username ‘test_manager’ and password ‘password’, which has the role ‘ROLE_MANAGER’.
9. In the field ‘Container to unprotect’ paste the protected container
10. Click on ‘Submit’
11. If the Consumer and Broker are correctly configured, you should see the message, “Protection was successfully removed from the container.” The unprotected text should be displayed, as shown in the figure below.

Consumer test web page: user authorised

12. Logout of the consumer using the 'Logout' link on the right of the navigation bar or on the home page.
13. In a browser, navigate to the test page on the CBS Consumer: https://<host name>:<Tomcat SSL port>/fiware.cbs.consumer.service.webapp/webui/test. Log in with username ‘test_user’ and password ‘password’, which has the role ‘ROLE_USER.
14. In the field ‘Container to unprotect’ paste the protected container
15. Click on ‘Submit’
16. If the Consumer and Broker are correctly configured, you should see the message, “Decrypt failed (not authorised).”

Consumer test web page: user unauthorised

14.4.2 List of Running Processes

- Java.exe

14.4.3 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.

14.4.4 Databases

N/A

14.5 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.

14.5.1 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

14.5.2 Remote Service Access

The CBS OGE needs access to an instance of the Access Control GE over HTTPS on port 443.

14.5.3 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.

14.5.4 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.

14.6 References

1. ↑ http://tomcat.apache.org/tomcat-7.0-doc/setup.html
2. ↑ 2.0 2.1 http://maven.apache.org/
3. ↑ FIWARE.ArchitectureDescription.Security.Access_Control_Generic_Enabler
4. ↑ FIWARE.ArchitectureDescription.Identity_Management_Generic_Enabler
5. ↑ Content-based_Security_-_Unit_Testing_Plan
6. ↑ https://forge.fi-ware.org/frs/?group_id=23&release_id=402#security-contentbasedsecurity-3-2-1-title-content
7. ↑ http://tomcat.apache.org/tomcat-7.0-doc/ssl-howto.html
8. ↑ https://forge.fi-ware.org/plugins/mediawiki/wiki/testbed/index.php/Security,_Privacy_and_Trust_-_Test_Cases_-_V3

15 Secure Storage Service - Installation and Administration Guide

You can find the content of this chapter as well in the wiki of fi-ware.

15.1 Introduction

This document is intended as an installation and administration guide for usage of the Secure Storage Service provided by Thales.

15.2 System requirements

15.2.1 Database

We used the MySQL 5.0.77 database. To install it, do the following:

```
apt-get install mysql-server
```

15.2.2 Apache Sever

15.2.2.1 *Installation*

In order to deploy the SSS engine and the use case application locally you need a host server. We recommand an Apache Tomcat 6 server Apache Tomcat 6.0:

```
To install Apache 2

#apt-get install apache2

To install Tomcat 6

#pat-get install tomcat6 tomcat6-admin  tomcat6-common tomcat6-docs
tomcat6-examples tomcat6-user

To install the Apache 2 connector to the Tomcat engine

#apt-get install libapache2-mod-jk
```

15.2.2.2 *Configuration*

In order to use secured connections to access the SSS, you need to configure the Apache Server to authorize HTTPS/SSL connections. To do that, you can refer to the following instructions: Apache Tomcat SSL How To.

15.2.3 Java VM

The SSS requires Java version 1.6:

```
apt-get install sun-java6-jdk openjdk-6-jdk
```

15.2.4 Mozilla Firefox

The SSS is web-based and the user interface should be displayed in a browser. We recommand the usage of a Firefox borwser Firefox

15.3 Installation

15.3.1 Installation Procedures

The Secure Storage Service is deployed directly on Tomcat as a .war file. To do that, copy the SecureStorageService.war file in the */<path-to-apache-tomcat-directory>/webapps/* folder. This can be done with the following command :

```
cp SecureStorageService.war /<path-to-apache-tomcat-
directory>/webapps/
```

The client Applet connecting to the SSS is deployed in the */<path-to-apache-tomcat-directory>/webapps/SecureStorageService/* folder with the HTML page Form.html. This can be done with the following command :

```
cp SignedApplet.jar /<path-to-apache-tomcat-
directory>/webapps/SecureStorageService/

cp Form.html /<path-to-apache-tomcat-
directory>/webapps/SecureStorageService/
```

You need to import the applet certificate *sssapplet.crt* in your web browser in order to use the signed applet connecting to the SSS web services.

15.3.2 Database Deployement

First you need to connect to mysql with the login determined when installing mysql, and the password associated, then to create the database and grant access to it for your user :

```
#mysql -u "login" -p"password"

>CREATE DATABASE sss;
```

```
>GRANT USAGE on *.* to sss_user@localhost identified by
'sss_user_pw';

>GRANT ALL PRIVILEGES on sss.* to sss_user@localhost;

>quit
```

Finally you create the tables needed by the SSS:

```
#mysql -u sss_user -p'sss_user_pw' sss

>CREATE TABLE user (username VARCHAR(30) NOT NULL UNIQUE, password
VARCHAR(50) NULL DEFAULT NULL, active BOOLEAN NOT NULL DEFAULT 1,
PRIMARY KEY (username));

>CREATE TABLE data (username VARCHAR(30) NOT NULL, id INT(11) NOT
NULL AUTO_INCREMENT, filename VARCHAR(50) NOT NULL, xmldata LONGTEXT
NOT NULL, PRIMARY KEY (id));

>CREATE TABLE sp (spname VARCHAR(30) NOT NULL UNIQUE, password
VARCHAR(50) NULL DEFAULT NULL, list TEXT NULL DEFAULT NULL, PRIMARY
KEY (spname));

>quit
```

15.3.3 Access to SSS server

The Secure Storage Service can be accessed at
https://localhost:8443/SecureStorageService/Form.html

There are two ports listening to requests:

- **8080**: HTTP access to SSS User registration demo,
- **8443**: HTTPS access to SSS.

It should be made sure that these ports are not blocked by a firewall in your organization.

15.4 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.

15.4.1 End to End testing

To verify that the SSS application was correctly deployed on the application server:

- check whether, at the URL https://localhost:8443/SecureStorageService/Form.html, an HTML frontend with a FIWARE logo is properly displayed;

To verify that the SSS works correctly:

- verify that logging in a random user (with userID and password) would return the following message:

```
The user 'userID' does not exist.
```

This would ensure that DB connection is properly set. Any other message could represent a possible deployment issue.

15.4.2 List of Running Processes

On Unix systems:

- java (for Apache Tomcat, or similar entries for the Java application server in use)
- mysqld
- apache

15.4.3 Network interfaces Up & Open

- Java application server (Apache Tomcat)

http: port 8080

https: port 8443

- MySQL database: port 3306.

15.4.4 Databases

- check whether MySQL instance is up and running, and reachable from the SSS Server;
- check whether the required MySQL database schema "sss" is created, and the username supplied has granted all rights on it;
- check whether the required MySQL table is properly created into "sss".

```
SELECT * FROM sss.user;
```

It should return no results if the query is executed before the first Secure Storage Service execution.

15.5 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.

15.5.1 Resource availability

Resource load of SSS strongly depends on the size of incoming data, and from the number of concurrent requests received. There are different aspects to consider on 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. A number of CPU cores could be assigned to MySQL DBMS as well.

15.5.2 Remote Service Access

N/A

15.5.3 Resource consumption

Resource consumption strongly depends on inputs.

15.5.4 I/O flows

The only expected I/O flow is of type HTTPS (or HTTP), and inbound only. Requests interactivity should be low.

16 Identity Management Global Customer Platform (GCP) - Software Release and Installation and Administration guide

The Global Customer Platform GE, which is since January 2014 an official product of Deutsche Telekom (product name: Internet Business Suite), 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[1].

In accordance with the privacy restrictions requested by GE owner, both the software (binaries) and the Installation and Administration Guides were not reviewed 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 on a remote server located at http://internet-business-suite.com/de/home after registration with email address and password, the software and the documentation is accessible. The partner who is the GE owner will ultimately provide this access.

17 Identity Management DigitalSelf GE - Software Release and Installation and Administration guide

The Identity Management DigitalSelf 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[1].

In accordance with the privacy restrictions requested by GE owner, both the software (binaries) and the Installation and Administration Guides were not reviewed 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 on a remote server located at http://idmlab03.extranet.nokiasiemensnetworks.com/fiware. Username and password will be disclosed upon explicit request to the project coordinator. The partner who is the GE owner will ultimately provide this access.

18 Malware Detection Service - Software Release and Installation and Administration guide

The Malware Detection Service 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[1].

In accordance with the privacy restrictions requested by GE owner, both the software (binaries) and the Installation and Administration Guides were not reviewed 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 Installation and Administration Guides are accessible to the EC for reviewing purposes on simple request at lhs@loria.fr or guillaume.bonfante@loria.fr.