



Deliverable Number: D33.3

Factsheet Number: #2

*Software prototype name: “Federated SSO
Utility Service - Local Security Authority
Service”*

Document Owner: Jerry Dimitriou (SINGULAR)

Contributors:

Dissemination: Public

Contributing to: WP33

Date: 29/03/2013

Revision: 1.0

List of Abbreviations

IDE	Integrated Development Environment
JDK	Java Development Kit
SVN	Subversion (versioning control system)
SSO	Single Sign-On
LSA	Local Security Authority

Table of Contents

1	AVAILABILITY AND CONTACTS	4
2	ARCHITECTURE AND FUNCTIONALITIES	5
2.1	INTRODUCTION	5
2.2	SUMMARY OF REQUIREMENTS	5
2.3	TECHNOLOGIES	5
3	TECHNICAL INFORMATION.....	6
3.1	SERVICE REQUIREMENTS	6
3.1.1	<i>Operating systems</i>	6
3.1.2	<i>Java development kit</i>	6
3.1.3	<i>Database</i>	6
3.2	DEVELOPMENT ENVIRONMENT	6
3.3	TECHNICAL DETAILS	6
4	LICENSING	7
4.1	SERVICE LICENSE	7
4.2	THIRD PARTY LICENSES	7
5	TECHNICAL MANUAL	8
5.1	INSTALLATION	8
5.1.1	<i>Database setup</i>	8
5.1.2	<i>Deploy the WAR file</i>	8
5.1.3	<i>Configure database access</i>	8
5.2	RETRIEVING SOURCES FROM SVN	8
6	USER MANUAL.....	9
7	FUTURE PLANS.....	10

1 Availability and Contacts

Version	1.0
Availability	https://msee.eng.it/sso/login
Accompanying specification and design document	D33.3 FI Utility Services first prototype – M18
Source control	<code>svn://repo.nimbus-ware.com/MSEE/SP3/WP33/D33.3/trunk/MSEE_WP33_LsaService</code>
Contact person	Jerry Dimitriou – ep6@singularlogic.eu

2 Architecture and Functionalities

2.1 Introduction

The purpose of the Local Security Authority Service is to provide an implementation of authentication and attribute release features for MSEE organizations. A number of LSA Service instances will be federated by SSO Utility Service to provide federated SSO to MSEE ecosystems.

2.2 Summary of Requirements

The LSA Service is a two-faced object. On one side it exposes a public endpoint for user credential validation and user attribute release. On the other side it connects with some pluggable user data repository.

The public endpoint is a RESTful service that accepts GET requests with username and password parameters; the response body is a SAML document stating the authenticated identity and user attributes.

2.3 Technologies

The LSA Service provides a REST API frontend served by a Java web container (eg. Apache Tomcat) based on common Java frameworks for web and enterprise applications development, including Spring, JPA/Hibernate and Apache CXF framework.

We take advantage of existing open source implementations provided by Jasig CAS server project (<http://www.jasig.org/>) to provide the authentication and attribute release backends. In the current prototype, LSA Service uses a JDBC-based authentication backend to provide username/credentials authentication and attribute release based on a relational database.

3 Technical Information

3.1 Service requirements

The LSA Service is a deployable WAR that can be used in any java web app server (e.g. Apache Tomcat). It requires a database to support the JDBC-based authentication backend. It does not depend on other external services.

3.1.1 Operating systems

No specific constraint. (This module has been successfully installed and tested on Windows 7 Enterprise Edition and Mac OS X).

3.1.2 Java development kit

This module has been developed, compiled and tested with Oracle JDK versions 6 and 7.

3.1.3 Database

LSA Service has been tested using MySQL Database engine, however any database with a JDBC driver available should work.

3.2 Development environment

LSA Service has been developed using Maven for build & dependency management.

3.3 Technical details

Nature	Source code
Programming Language	Java
Development Tools	Maven
Additional libraries	Spring Apache CXF JPA/Hibernate Jasig CAS libraries
Application Server	Apache Tomcat 7 (any Java webapp server should be fine)
Database	MySQL (other relational DBs should work)

4 Licensing

4.1 Service license

To be defined.

4.2 Third party licenses

Third party software	License
Spring	Apache License version 2.0
Hibernate	LGPL v2.1
Apache CXF	Apache License version 2.0
Jasig CAS	Vedi http://www.jasig.org/cas/license

5 Technical Manual

5.1 Installation

5.1.1 Database setup

After having successfully setup a MySQL instance using the appropriate installer from <http://www.mysql.com/downloads/>, a new database must be created. A sample schema for MySQL is provided in `src/main/sql/dbSMYSQLsample.sql`, which creates a basic schema and inserts some test data for users with test passwords and a set of attributes according to MSEE security model.

5.1.2 Deploy the WAR file

The Federated SSO Utility Service package can be downloaded from the MSEE portal: <http://www.msee-ip.eu/intranet/sp3-workspace/sp3-wp33/d3.3.3-fi-utility-services-first-prototype-m18>

The package *D33.3 FI Utility Services first prototype - M18 - FederatedSSO.bin.zip* contains the file `sso-1sa.war`. Copy the distributed binary `sso-1sa.war` to `${TOMCAT}/webapps/` and start Tomcat server. Tomcat will deploy the web application. The preconfigured default settings for database access are:

- MySQL server at localhost
- Database name is `msee_sp3_1sa`
- Username & password for accessing the database is “msee” – “msee”.

In case any of the above defaults are not correct, stop Apache Tomcat server, reconfigure the server as described below and restart Apache Tomcat.

5.1.3 Configure database access

In case you need to customize database access, open file `${TOMCAT}/webapps/sso-1sa/WEB-INF/classes/database.properties` in a plain text editor and change the following lines appropriately:

Property name	Value
<code>jdbc.url</code>	JDBC URL of MySQL instance, including hostname and port on which MySQL is installed, as well as database name in the format below: <code>jdbc:mysql://HOSTNAME[:PORT]/DB_NAME</code>
<code>jdbc.username</code>	Username of MySQL user with read access to database
<code>jdbc.password</code>	Password of MySQL user

5.2 Retrieving sources from SVN

Source code for the LSE Service is published on the MSEE SVN repository: svn://repo.nimbus-ware.com/MSEE/SP3/WP33/D33.3/trunk/MSEE_WP33_LsaService

6 User Manual

Assuming the LSA Service is deployed on a Tomcat instance on localhost, port 8080, the REST service is accessible at

<http://localhost:8080/sso-lsa/api/authenticate?uid={UID}&password={PASSWORD}>

This endpoint can be invoked by any standard HTTP client via GET method, e.g. via curl command line tool. The server responds to the GET request by issuing a response with status 200-OK and a SAML2 document as response body or status 404-NOT FOUND in case of any other failure.

7 Future plans

For future iterations of the LSA, we plan to support other authentication backends that can plug to existing organizations infrastructure such as Active Directory and/or LDAP repositories.