EUROPEAN

MIDDLEWARE INITIATIVE

PSEUDONYMITY

SYSTEM ADMINISTRATOR GUIDE

Document Version:	**1.0**
EMI Component Version:	**1.1.0**
Date:	**26.04.2012**

This work is co-funded by the EC EMI project under the FP7 Collaborative Projects Grant Agreement Nr. INFSO-RI-261611.

Pseudonymity Server: Configuration

Introduction

EMI Pseudonymity System is meant for providing its users a way to hide their true identity behind a pseudonymous identity. The term refers to a lesser degree of anonymity, as the relationships between the pseudonyms and the true identities can be revealed eg. in the cases of misuse. Users obtain pseudonymous identities from the Pseudonymity Service by using their existing Grid identity for authentication and authorization. The service registers the newly issued pseudonyms to the VO's attribute authority as aliases to the users' existing certificates. The service is accessed with a dedicated client tool.

This guide explains how to install and configure the server-side.

Quickstart

The nature of the pseudonymity server is to integrate very closely with two external parties: the online CA and attribute authority (VOMS). For this reason, there is no quick guide for making the software work without interoperating closely with the administrators of the external parties.

Installation

Make sure that the host has NTP properly configured. Clock skew between the clients, server and external parties may cause many types of failures.

The server can be installed from the EMI repositories. The package name `pseudonymity-server`.

For SL5/SL6, the installation can be done using the command `yum install pseudonymity-server`.

Main configuration File

The pseudonymity server is configured through the use of the `pseudonymity-server.ini` file. This file is a standard INI file with different defined sections.

The `StandaloneServer` section contains properties related the service as a whole and how it listens for incoming requests. The `DNBuilder` section controls how the DNs are built for the outgoing certificates. The `AttributeDefinitions` section controls how the VO attributes are configured. The `CertificatePolicy` section controls what kind of extensions are included in the outgoing certificates. The `CAClient` section configures the communication with an external online Certification Authority, as the `AAClient` section configures the external Attribute Authority (VOMS). The `Auditor` and `AuditAdmin` sections control how the audit data is stored into a database as the `ACL` and `AdministratorACL` sections finally control who are authorized to use and administrate the service.

Main configuration Options

StandaloneServer section

Property	Description	Required?	Default Value
hostname	hostname of the host running the service	Y	None
port	port number that the	Y	8443

	service is listening to		
webapp_context	Web application context	Y	/pseudo
warfile	full pathname to the Web application archive (WAR)	Y	/usr/share/pseudonymity/server/wars/pseudo.war
certificate	full pathname to the certificate protecting the endpoint	Y	/etc/grid-security/hostcert.pem
private_key	full pathname to the private key protecting the endpoint	Y	/etc/grid-security/hostkey.pem
crl_enabled	is revocation list check enabled for the incoming certificates	N	true
crl_update_interval	interval for updating the revocation lists	N	30m (= 30minutes)
trust_store_dir	full pathname to the directory containing the certificates	N	/etc/grid-security/certificates
max_request_queue_size	maximum queue size for the incoming requests	N	-1 (= unbounded)
max_connections	maximum amount of parallel connections	N	64
shutdown_command	(internal) command for shutting down the service	N	shutdown
shutdown_port	port for shutting down the service	Y	8444

DNBuilder section

Property	Description	Required?	Default Value
implementation	class implementing the required interface	Y	org.glite.pseudo.server.dn.impl.PseudoDNBuilder
DNTemplate	Template for building the DNs	Y	C=FI,O=HIP,OU=Tech,DC=pseudotest,CN=${pseudoIdentifier}
IDLength	The length of the pseudoIdentifer -part of the DN	Y	8

AttributeDefinitions section

Property	Description	Required?	Default Value
implementation	class implementing the required interface	Y	org.glite.pseudo.server.attribute.AttributeDefinitionsImpl

ConfigFile	full pathname for the configuration file	Y	/etc/pseudonymity/server/attribute-defs.xml

CertificatePolicy section

Property	Description	Required?	Default Value
implementation	class implementing the required interface	Y	org.glite.pseudo.server.policy.impl.MandatoryCertificateExtensionsPolic
extensionId	identifier for the extension	N	e.g. KeyUsage
extensionCritical	is the extension critical or not	N	e.g. true
extensionValue	the value for the extension	N	e.g. DigitalSignature,KeyEncipherment

CAClient section

Property	Description	Required?	Default Value
implementation	class implementing the required interface	Y	org.glite.pseudo.server.caclient.impl.CMPClient
Endpoint	endpoint for the CMP service	Y	None
CertificateFile	full pathname for the certificate file used in the client authentication	Y	/etc/grid-security/servicecert.pem
KeyFile	full pathname for the private key file used in the client authentication	Y	/etc/grid-security/servicekey.pem
KeyPasswd	password protecting the private key	N	None
RequireCRLCheck	is revocation list check required for the service endpoint	Y	false
TruststoreFile	full pathname for the truststore file	Y	None
TruststorePasswd	password protecting the truststore	Y	None
CADN	CMP-related configuration parameter	Y	None
SenderDN	CMP-related configuration parameter	Y	None
RecipientDN	CMP-related configuration parameter	Y	None
SenderKID	CMP-related configuration parameter	Y	None
SharedSecret	CMP-related configuration parameter	Y	None
OwfAlgId	CMP-related configuration parameter	N	1.3.14.3.2.26

IterCount	CMP-related configuration parameter	N	1
MacAlgId	CMP-related configuration parameter	N	1.3.6.1.5.5.8.1.2
SaltString	CMP-related configuration parameter	N	None
ProtectiongAlgId	CMP-related configuration parameter	N	1.2.840.113533.7.66.13

AAClient section

Property	Description	Required?	Default Value
implementation	class implementing the required interface	Y	org.glite.pseudo.server.aaclient.impl.VOMSClient
Endpoint	endpoint for the VOMSCertificates service	Y	None
CertificateFile	full pathname for the certificate file used in the client authentication	Y	/etc/grid-security/servicecert.pem
KeyFile	full pathname for the private key file used in the client authentication	Y	/etc/grid-security/servicekey.pem
KeyPasswd	password protecting the private key	N	None
RequireCRLCheck	is revocation list check required for the service endpoint	Y	false

Auditor section

Property	Description	Required?	Default Value
implementation	class implementing the required interface	Y	org.glite.pseudo.server.auditor.impl.HibernateAuditor
HibernateConfiguration	full pathname for the Hibernate configuration file	Y	/etc/pseudonymity/server/hibernate/auditeventdatabase.hibernate
HibernateMappings	full pathname for the Hibernate mappings configuration file	Y	/etc/pseudonymity/server/hibernate/HibernateAuditEventDataba
PersonImplementation	class implementating the required Person interface	Y	org.glite.pseudo.server.auditor.impl.AuditPersonEntryImpl
EventImplementation	class	Y	org.glite.pseudo.server.auditor.impl.AuditEventEntryImpl

	implementing the required Event interface		

AuditAdmin section

Property	Description	Required?	Default Value
implementation	class implementing the required interface	Y	org.glite.pseudo.server.auditor.impl.HibernateAuditAdmin
HibernateConfiguration	full pathname for the Hibernate configuration file	Y	/etc/pseudonymity/server/hibernate/auditeventdatabase.hibernate
HibernateMappings	full pathname for the Hibernate mappings configuration file	Y	/etc/pseudonymity/server/hibernate/HibernateAuditEventDataba
PersonImplementation	class implementating the required Person interface	Y	org.glite.pseudo.server.auditor.impl.AuditPersonEntryImpl
EventImplementation	class implementing the required Event interface	Y	org.glite.pseudo.server.auditor.impl.AuditEventEntryImpl

ACL section

Property	Description	Required?	Default Value
implementation	class implementing the required interface	Y	org.glite.pseudo.server.acl.impl.XMLFileAccessControlList
ACLFile	full pathname for the configuration file	Y	/etc/pseudonymity/server/acl.xml

AdministratorACL section

Property	Description	Required?	Default Value
implementation	class implementing the required interface	Y	org.glite.pseudo.server.acl.impl.XMLFileAccessControlList
ACLFile	full pathname for the configuration file	Y	/etc/pseudonymity/server/acl-admin.xml

Example pseudonymity-server.ini file

The following example file contain an example for the configuration file:

```
[StandaloneServer]
hostname = example.org
port = 8443
webapp_context = /pseudo
warfile = /usr/share/pseudonymity/server/wars/pseudo.war
certificate = /etc/grid-security/hostcert.pem
private_key = /etc/grid-security/hostkey.pem
crl_enabled = true
crl_update_interval = 30m
trust_store_dir = /etc/grid-security/certificates
max_request_queue_size = 10
max_connections = 20
shutdown_command = shutdown
shutdown_port = 8444

[DNBuilder]
implementation = org.glite.pseudo.server.dn.impl.PseudoDNBuilder
DNTemplate = C=FI,O=HIP,OU=Tech,DC=pseudotest,CN=${pseudoIdentifier}
IDLength = 8

[AttributeDefinitions]
implementation = org.glite.pseudo.server.attribute.AttributeDefinitionsImpl
ConfigFile = /etc/pseudonymity/server/attribute-defs.xml

[CertificatePolicy]
implementation = org.glite.pseudo.server.policy.impl.MandatoryCertificateExtensionsPolicy
extensionId = KeyUsage
extensionCritical = true
extensionValue = DigitalSignature,KeyEncipherment
extensionId = CertificatePolicies
extensionCritical = false
extensionValue = 2.16.756.1.2.6.4.1.0
extensionId = ExtendedKeyUsage
extensionCritical = false
extensionValue = ClientAuth

[CAClient]
implementation = org.glite.pseudo.server.caclient.impl.CMPClient
Endpoint = https://example.org/CA
CertificateFile = /etc/grid-security/servicecert.pem
KeyFile = /etc/grid-security/servicekey.pem
KeyPasswd = changeit
TruststoreFile = /etc/grid-security/truststore.jks
TruststorePasswd = changeit
RequireCRLCheck = false
CADN = CN=HIP-TEK Pseudo Test CA,OU=Tech,O=HIP,C=FI
SenderDN = C=FI
RecipientDN = C=FI
SenderKID = 1234567890
SharedSecret = 0987654321
OwfAlgId = 1.3.14.3.2.26
IterCount = 1
MacAlgId = 1.3.6.1.5.5.8.1.2
SaltString =
ProtectionAlgId = 1.2.840.113533.7.66.13

[AAClient]
implementation = org.glite.pseudo.server.aaclient.impl.VOMSClient
Endpoint = https://voms.example.org/endpoint
CertificateFile = /etc/grid-security/servicecert.pem
KeyFile = /etc/grid-security/servicekey.pem
```

```
RequireCRLCheck = false

[Auditor]
implementation = org.glite.pseudo.server.auditor.impl.HibernateAuditor
HibernateConfiguration = /etc/pseudonymity/server/hibernate/auditeventdatabase.hibernate.cfg.xml
HibernateMappings = /etc/pseudonymity/server/hibernate/HibernateAuditEventDatabase.hbm.xml
PersonImplementation = org.glite.pseudo.server.auditor.impl.AuditPersonEntryImpl
EventImplementation = org.glite.pseudo.server.auditor.impl.AuditEventEntryImpl

[AuditAdmin]
implementation = org.glite.pseudo.server.auditor.impl.HibernateAuditAdmin
HibernateConfiguration = /etc/pseudonymity/server/hibernate/auditeventdatabase.hibernate.cfg.xml
HibernateMappings = /etc/pseudonymity/server/hibernate/HibernateAuditEventDatabase.hbm.xml
PersonImplementation = org.glite.pseudo.server.auditor.impl.AuditPersonEntryImpl
EventImplementation = org.glite.pseudo.server.auditor.impl.AuditEventEntryImpl

[AttributeDefinitions]
implementation = org.glite.pseudo.server.attribute.AttributeDefinitionsImpl

[ACL]
implementation = org.glite.pseudo.server.acl.impl.XMLFileAccessControlList
ACLFile = /etc/pseudonymity/server/acl.xml

[AdministratorACL]
implementation = org.glite.pseudo.server.acl.impl.XMLFileAccessControlList
ACLFile = /etc/pseudonymity/server/acl-admin.xml
```

Other configuration files

The following sections contains examples of other configuration files that are referenced from the main configuration file's options (see above).

Attribute definitions

Attributes are defined in an XML file containing a list of attribute definitions that can be referenced from the access control lists (see next sections). The AttributeDefinition has four self-explaining mandatory elements: name, header, required and displayName. Currently certificate DN and VOMS FQANs are supported.

By default, this file exists at `/etc/pseudonyity/server/attribute-defs.xml`.

Example attribute-defs.xml file

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<AttributeDefinitions>
   <AttributeDefinition
      name="VOMS:FQAN"
      header="VOMS-FQAN"
      required="false"
      displayName="VOMS Fully Qualified Attribute Name" />
   <AttributeDefinition
      name="Certificate:DN"
      header="Certificate-DN"
      required="false"
      displayName="X.509 certificate subject DN" />
</AttributeDefinitions>
```

Access control lists

Two types of access control lists exists for the pseudonymity service: the first one controlling the normal users accessing the service and the other one controlling the administrator access.

User access control list

The access control list contains the list of attributes that the users MUST have in order to access the service. If the list is null, then all the users with a valid trusted certificate are authorized to use the service.

By default, this file exists at `/etc/pseudonymity/server/acl.xml`.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<AccessControlList>
   <AccessControlRule id="1" group="test_oracle">
      <Attribute name="VOMS-FQAN">/test_oracle/g1&lt;/Attribute>
   </AccessControlRule>
</AccessControlList>
```

The example above would authorize all the users from the VO group `/test_oracle/g1` to use the service.

Administrator access control list

The administrator access control list contains the list of attributes from which the administrator must have at least one in order to administer the service. If the list is null, then nobody is authorized to use the administrator interface of the service.

By default, this file exists at `/etc/pseudonymity/server/acl-admin.xml`.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<AccessControlList>
   <AccessControlRule id="1" group="admins">
      <Attribute name="Certificate-DN">CN=John Doe\, CN=123456\, CN=johndoe\, OU=Users\, OU=Organ
   </AccessControlRule>
</AccessControlList>
```

The example above would authorize the admin with the corresponding subject DN in the trusted certificate to access the administrator interface.

Database configurations

The pseudonymity service requires a relational database to store the audit information of the usage. All the databases supported by Hibernate are supported by the service.

The database connection configuration

This file defines how the service can access the database. It is a standard Hibernate XML configuration file, and the most important configuration properties include:

- `connection.driver_class` The full class name for the JDBC driver
- `connection.url` The url to access the database
- `connection.username` The username to use the database
- `connection.password` The password corresponding to the username

The configuration file below is a very simple example using MySQL, not meant for production environments:

```
<?xml version='1.0' encoding='utf-8'?>
<hibernate-configuration>
    <session-factory>
        <!-- Database connection settings -->
        <property name="connection.driver_class">org.gjt.mm.mysql.Driver</property>
        <property name="connection.driver_class">com.mysql.jdbc.Driver</property>
        <property name="connection.url">jdbc:mysql:///glite_pseudo</property>
```

```
        <property name="connection.username">pseudo</property>
        <property name="connection.password">service</property>
        <!-- JDBC connection pool (use the built-in) -->
        <property name="connection.pool_size">0</property>
        <!-- SQL dialect -->
        <property name="dialect">org.hibernate.dialect.MySQLDialect</property>
        <property name="dialect">org.hibernate.dialect.MySQLInnoDBDialect</property>
        <property name="dialect">org.hibernate.dialect.MySQLMyISAMDialect</property>
        <!-- Echo all executed SQL to stdout -->
        <property name="show_sql">true</property>
        <!-- Drop and re-create the database schema on startup -->
        <property name="hbm2ddl.auto">create</property>
        <property name="hibernate.current_session_context_class">org.hibernate.context.ThreadLoca
    </session-factory>
</hibernate-configuration>
```

This example also takes care of creating the corresponding tables to the database with the property `hbm2ddl.auto` set to `create`. After creating them with this option, this should be changed to `update` as the other option always re-creates the tables.

Object mappings configuration

This file defines how the Java objects are stored into the database. This file needs to be modified only by the developers who wants to customize the Java objects first.

Include the database JDBC connector to the classpath

In order for the service to use the desired JDBC connector, the JAR file must be located in `/var/lib/pseudonymity/server/lib` directory.

-- HenriMikkonen - 22-Apr-2012

This topic: EMI > PseudonymityServerConfiguration
Topic revision: r5 - 25-Apr-2012 - 18:05:18 - HenriMikkonen

Pseudonymity Server: Operation

Starting & Stopping the Service

`/etc/init.d/pseudonymity-service start`
: to start the pseudonymity service

`/etc/init.d/pseudonymity-service stop`
: to stop the pseudonymity service

`/etc/init.d/pseudonymity-service restart`
: to restart the pseudonymity service

`/etc/init.d/pseudonymity-service status`
: to show status of the pseudonymity service

Service Information

Service Endpoints

This service contains the following endpoint URLs:

- `https://pseudo.example.org:8443/pseudo/Login` - This endpoint is the recipient of login requests from the client tool
- `https://pseudo.example.org:8443/pseudo/certificate` - This endpoint is the recipient of certificate requests from the client tool after login
- `https://pseudo.example.org:8443/pseudo/admin` - This endpoint provides the admin Web interface

Logging and Logs

This service uses the logback logging library. Java developers are probably familiar with Apache Log4J, logback is written by the developer who initially wrote Log4J and contains a cleaner API and is much more performant. The configuration file for the logging system can be found in `/etc/pseudonymity/server/logging.xml`.

Enable Debug Logging

To enable debug logging follow:

1. Locate the line that contains `logger name="org.glite.pseudo"`
2. On that line, change `INFO` to `DEBUG`

NOTE always change your logging levels back to their original values once you are done debugging a problem. Keeping the system on the debug logging level could fill up your disk partition in a short time.

This topic: EMI > PseudonymityServerOperation
Topic revision: r5 - 26-Apr-2012 - 13:25:03 - HenriMikkonen