EUROPEAN MIDDLEWARE INITIATIVE

EUROPEAN

MIDDLEWARE INITIATIVE

SECURITY TOKEN SERVICE

SYSTEM ADMINISTRATOR GUIDE

Document Version:	**1.0.0**
EMI Component Version:	**1.0.0**
Date:	**11.02.2013**

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

STS Server: Configuration

Introduction

EMI Security Token Service (STS) is a partial implementation of the OASIS WS-Trust specification. It is a service that can be used for transforming an existing security token into another security token format. Security token, on the other hand, is defined in the WS-Security specifications as a collection of claims that can be attached into a Web Service message.

The incoming token formats that are supported by our STS implementation include username and password that is validated against an LDAP directory, and SAML assertion. From these types of tokens, STS can issue an X.509 certificate or a proxy certificate containing the users' VO attributes.

The current implementation of the STS focus on a subset of all the possible functionality outlined with the WS-Trust specification. Specifically it will implement the issuance of the supported security tokens. It will not support the renewal, cancellation, or validation functionality, nor it will support all possible key and token parameter extensions.

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

Quickstart

The nature of the STS 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 `sts`.

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

Main configuration File

The STS server is configured through the use of the `sts-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 `Messages` section contains properties related to the incoming WS-Trust messages. The `SAML` section contains properties related to the consumation of the incoming SAML messages. The `LDAP` section defines how the STS server interacts with the LDAP directory for validating username tokens and collecting user attributes. The `CACLient` section configures the communication with an external online Certification Authority, as the `VOMS` section configures the VOMS communication.

The checking mode configuration options for namespaceCheckingMode, crlCheckingMode, ocspCheckingMode and proxySupport that exist in multiple section can be found from the `CA & certificate checking modes` section of this document.

Main configuration Options

StandaloneServer section

Property	Description	Default Value
hostname	hostname of the host running the service	localhost
port	port number that the service is listening to	8443
scheme	the connection scheme, http or https	https
webappContext	the Web application context in the URL	sts
warfile	full pathname to the Web application archive (WAR)	/usr/share/sts/wars/sts-server-1.0.0.war
certificate	full pathname to the certificate protecting the endpoint	/etc/grid-security/hostcert.pem
private_key	full pathname to the private key protecting the endpoint	/etc/grid-security/hostkey.pem
crlUpdateInterval	interval for updating the revocation lists in milliseconds	600000
trustStoreDir	full pathname to the directory containing the certificates	/etc/grid-security/certificates
maxRequestQueueSize	maximum queue size for the incoming requests	-1 (= unbounded)
maxConnections	maximum amount of parallel connections	64
shutdownCommand	(internal) command for shutting down the service	shutdown
shutdownPort	port for shutting down the service	8444
namespaceCheckingMode	CA namespace checking mode for the incoming connections	EUGRIDPMA
crlCheckingMode	CRL checking mode for the incoming connections	REQUIRE
ocspCheckingMode	OCSP checking mode for the incoming connections	IF_AVAILABLE
proxySupport	proxy allowance for the incoming connections	ALLOW
wantClientAuth	is client certificate authentication wanted for the incoming connections	false
requireClientAuth	is client certificate authentication required for the incoming connections	false

Messages section

Property	Description	Default Value
clockSkew	the clock skew in milliseconds allowed in the incoming message timestamps	300000
lifetime	how long the timestamps are valid in milliseconds in the incoming messages	300000
requireIssueInstant	is issue instant required in the incoming messages	true
requireFrameworkVersion	is sbf:Framework version 2.0 required in the incoming messages	true
requireMessageId	is message identifier required in the incoming messages	true
requiredRecipientId	which recipient identifier is required in the incoming messages (empty means not required)	

schemaDirectory	the directory containing the XML schema definition files	/usr/share/sts/schema
requireSignature	is signature required in the incoming messages	false

SAML section

Property	Description	Default Value
entityId	the SAML entity id of the STS	https://sts.example.org/saml
consumerServiceUrl	the consumer service URL of the STS	/soap
metadataFile	directory containing the metadata for the trusted Identity Providers	/usr/share/sts/metadata
metadataFileFrequency	how often the metadata files are reloaded in milliseconds	300000
certFile	the certificate file used for signing the SAML messages	/etc/grid-security/servicecert.pem
keyFile	the private key file used for signing the SAML messages	/etc/grid-security/servicekey.pem
keyPass	optional password protecting the private key	
assertionLifetime	lifetime in minutes how long the assertions are considered valid	30
requireRequestId	is authentication request id required to exist in the local storage	true
requestIdLifetime	how long the request ids are valid in the local storage in milliseconds	300000
requestIdSweepFrequency	how often the expired request ids are sweeped from the local storage	300000
authnLifetime	how long the authentication instant is valid in the assertion	300000
requireAuthnStatement	is authentication statement required in the assertions	true
requireAuthnLocality	is authentication locality required in the assertions	false
requireConditions	are conditions required in the assertions	true
requireAudience	is audience element required in the assertions	true
remoteAddressHeader	the HTTP header containing the client IP address (for locality check)	X-Forwarded-For

LDAP section

Property	Description	Default Value
enableLdap	is LDAP enabled, configuration is ignored if false	false
serverUrl	the URL for the LDAP endpoint	ldap://127.0.0.1
useStartTls	is start TLS used in the connection	false
baseDn	base DN for the users in the directory	dc=people,dc=organization,dc=com
bindDN	the DN of the administrator user used for binding	
bindCredential	the credential of the administrator user	
uid	the attribute name for the uid	
userPassword	the attribute name for the hashed password	
hashAlgorithm	the hash algorithm, only SHA1 is supported at the moment	
blockWaitTime	the maximum waiting time in milliseconds for a connection	5000

minPoolSize	the minimum connection pool size	5
maxPoolSize	the maximum connection pool size	20
validatePeriodically	is connection validated periodically	false
validatePeriod	how often the connection is validated	30

CAClient section

Property	Description	Default Value
serverUrl	endpoint for the CMP service	https://ca.example.org:443/ejbca/publicweb/cmp
dnPattern	the pattern for building the DN, attribute name inside ${ }	C=xx,DC=xx,O=xx,OU=xx,CN=${cn}
certFile	full pathname for the certificate file used in the client authentication	/etc/grid-security/servicecert.pem
keyFile	full pathname for the private key file used in the client authentication	/etc/grid-security/servicekey.pem
keyPass	password protecting the private key	
trustStoreDir	the directory containing the CA certificate files	/etc/grid-security/certificates
trustStoreUpdateInterval	how often the truststore is updated	600000
namespaceCheckingMode	CA namespace checking mode for the CA connection	EUGRIDPMA
crlCheckingMode	CRL checking mode for the CA connection	REQUIRE
ocspCheckingMode	OCSP checking mode for the CA connection	IF_AVAILABLE
proxySupport	is proxy allowed in the CA connection	ALLOW
caDn	CMP-related configuration parameter	CN=xx CA,OU=xx,O=xx,C=xx
senderDn	CMP-related configuration parameter	C=xx
recipientDn	CMP-related configuration parameter	C=xx
senderKid	CMP-related configuration parameter	keyIdentifier
sharedSecret	CMP-related configuration parameter	sharedSecret
keysize	the keysize for the key-pair if generated by the STS	2048

VOMS section

Property	Description	Default Value
vomses	the directory containing the vomses files	/etc/grid-security/vomses
vomsDir	the directory containing the CA cerfificates	/etc/grid-security/vomsdir
updateInterval	how often the directories are updated	300000
namespaceCheckingMode	CA namespace checking mode for the VOMS connection	EUGRIDPMA

crlCheckingMode	CRL checking mode for the VOMS connection	REQUIRE
ocspCheckingMode	OCSP checking mode for the VOMS connection	IF_AVAILABLE
keysize	the keysize for the key-pair if generated by the STS	2048

CA & certificate checking modes

The following options exists for the checking modes, as documented in the EMI common authentication library:

namespaceCheckingMode:

- `GLOBUS_EUGRIDPMA`: A Globus EACL is checked first. If found for the issuing CA then it is used and enforced. If not found then EuGridPMA namespaces definition is searched. If found for the issuing CA then it is enforced. If no definition is present then namespaces check is considered to be passed.
- `EUGRIDPMA_GLOBUS`: An EuGridPMA namespaces definition is checked first. If found for the issuing CA then it is enforced. If not found then Globus EACL definition is searched. If found for the issuing CA then it is enforced. If no definition is present then namespaces check is considered to be passed.
- `GLOBUS`: A Globus EACL is checked only. If found for the issuing CA then it is used and enforced. If no definition is present then namespaces check is considered to be passed.
- `EUGRIDPMA`: An EuGridPMA namespaces definition is checked only. If found for the issuing CA then it is enforced. If no definition is present then namespaces check is considered to be passed.
- `GLOBUS_EUGRIDPMA_REQUIRE`: A Globus EACL is checked first. If found for the issuing CA then it is used and enforced. If not found then EuGridPMA namespaces definition is searched. If found for the issuing CA then it is enforced. If no definition is present then namespaces check is considered to be failed.
- `EUGRIDPMA_GLOBUS_REQUIRE`: An EuGridPMA namespaces definition is checked first. If found for the issuing CA then it is enforced. If not found then Globus EACL definition is searched. If found for the issuing CA then it is enforced.If no definition is present then namespaces check is considered to be failed.
- `GLOBUS_REQUIRE`: A Globus EACL is checked only. If found for the issuing CA then it is used and enforced. If no definition is present then namespaces check is considered to be failed.
- `EUGRIDPMA_REQUIRE`: An EuGridPMA namespaces definition is checked only. If found for the issuing CA then it is enforced. If no definition is present then namespaces check is considered to be failed.
- `EUGRIDPMA_AND_GLOBUS`: Both EuGridPMA namespaces definition and Globus EACL are enforced for the issuer. If no definition is present then namespaces check is considered to be passed.
- `EUGRIDPMA_AND_GLOBUS_REQUIRE`: Both EuGridPMA namespaces definition and Globus EACL are enforced for the issuer. If no definition is present then namespaces check is considered to be failed.
- `IGNORE`: CA namespaces are fully ignored, even if present.

crlCheckingMode:

- `REQUIRE`: A CRL for CA which issued a certificate being validated must be present and valid and the certificate must not be on the list.
- `IF_VALID`: If a CRL for CA which issued a certificate being validated is present and valid then the certificate must not be listed on the CRL. If the CRL is present but it is outdated (or anyhow else corrupted) then the validation fails. If CRL is missing then validation is successful.
- `IGNORE`: CRL is not checked even if it exists.

ocspCheckingMode:

- `REQUIRE`: Require, for each checked certificate, that at least one valid OCSP responder is defined and that at least one responder of those defined returns a correct certificate status. If all OCSP responders return error or unknown status, the last one received is treated as a critical validation error. Not suggested, unless it is guaranteed that well configured responder(s) is(are) defined and can handle all queries without timeouts.
- `IF_VALID`: Use OCSP for each certificate if a responder is available. OCSP 'unknown' status and query errors (as timeout) do not cause the validation to fail. Also a lack of defined responder doesn't cause the validation to fail.
- `IGNORE`: Do not use OCSP.

proxySupport:

- `ALLOW`: All kinds of proxies are allowed.
- `DENY`: All kinds of proxies are denied.

Example sts-server.ini file

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

```
[StandaloneServer]
hostname = localhost
port = 8443
scheme = https
webappContext = sts
warfile = /usr/share/sts/wars/sts-server-1.0.0.war
certificate = /etc/grid-security/hostcert.pem
private_key = /etc/grid-security/hostkey.pem
crlUpdateInterval = 600000
trustStoreDir = /etc/grid-security/certificates
maxRequestQueueSize =
maxConnections =
shutdownCommand = shutdown
shutdownPort = 8444
namespaceCheckingMode = EUGRIDPMA
crlCheckingMode = REQUIRE
ocspCheckingMode = IF_AVAILABLE
proxySupport = ALLOW
wantClientAuth = false
requireClientAuth = false

[Messages]
clockSkew = 300000
lifetime = 300000
requireIssueInstant = true
requireFrameworkVersion = true
requireMessageId = true
requiredRecipientId =
schemaDirectory = /usr/share/sts/schema
requireSignature = false

[SAML]
entityId = https://sts.example.org/saml
consumerServiceUrl = /soap
metadataFile = /usr/share/sts/metadata
metadataFileFrequency = 300000
certFile = /etc/grid-security/servicecert.pem
keyFile = /etc/grid-security/servicekey.pem
keyPass =
assertionLifetime = 30
requireRequestId = true
requestIdLifetime = 300000
requestIdSweepFrequency = 300000
```

```
authnLifetime = 300000
requireAuthnStatement = true
requireAuthnLocality = false
requireConditions = true
requireAudience = true
remoteAddressHeader = X-Forwarded-For

[LDAP]
enableLdap = false
serverUrl = ldap://127.0.0.1
useStartTls = false
baseDn = dc=people,dc=organization,dc=com
blockWaitTime = 5000
minPoolSize = 5
maxPoolSize = 20
validatePeriodically = false
validatePeriod = 30

[CAClient]
serverUrl = https://ca.example.org:443/ejbca/publicweb/cmp
dnPattern = C=xx,DC=xx,O=xx,OU=xx,CN=${cn}
certFile = /etc/grid-security/servicecert.pem
keyFile = /etc/grid-security/servicekey.pem
keyPass =
trustStoreDir = /etc/grid-security/certificates
trustStoreUpdateInterval = 600000
namespaceCheckingMode = EUGRIDPMA
crlCheckingMode = REQUIRE
ocspCheckingMode = IF_AVAILABLE
proxySupport = ALLOW
caDn = CN=xx CA,OU=xx,O=xx,C=xx
senderDn = C=xx
recipientDn = C=xx
senderKid = keyIdentifier
sharedSecret = sharedSecret
keysize = 2048

[VOMS]
vomses = /etc/grid-security/vomses
vomsDir = /etc/grid-security/vomsdir
updateInterval = 300000
namespaceCheckingMode = EUGRIDPMA
crlCheckingMode = REQUIRE
ocspCheckingMode = IF_AVAILABLE
keysize = 2048
```

Token-specific configuration

UsernamePassword

Username and passwords can be validated against an LDAP directory using two different approaches: binding the incoming user directly to the directory or reading the user attributes from the LDAP using administrator user, and comparing the password value against the given one. The first option is the recommended one.

Binding the incoming user

The following options are used from the `LDAP` configuration section:

- baseDn
- uid

The password sent in the token must be in plaintext. User is binded to the directory using `baseDn` and adding the username value from the token to the `uid` attribute with the password value from the token.

Use of administrator user

The following options are used from the `LDAP` configuration section:

- baseDn
- uid
- bindUser
- bindCredential
- userPassword

The password must be sent in SHA1 -hashed format, without nonce or created elements in the token. The user is validated by first obtaining `uid` and `userPassword` values from the directory using the `bindUser` account with `bindCredential`. The value of the `userPassword` attribute is then compared against the password given in the username token.

SAML assertion

The metadata for the STS can be constructed by using the following skeleton:

```
<?xml version="1.0" encoding="UTF-8"?>
<md:EntityDescriptor xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata"
      entityID="https://sts.example.org/saml">
   <md:SPSSODescriptor protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol"
      xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata">
      <md:KeyDescriptor xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata">
         <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
            <ds:X509Data xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
               <ds:X509Certificate xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
               </ds:X509Certificate>
            </ds:X509Data>
         </ds:KeyInfo>
      </md:KeyDescriptor>
      <md:AssertionConsumerService Binding="urn:oasis:names:tc:SAML:2.0:bindings:PAOS"
         Location="/soap" index="0" isDefault="true"
         xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata"/>
   </md:SPSSODescriptor>
</md:EntityDescriptor>
```

The contents from the file defined in the `certFile` must be included inside the X509Certificate element. Also, the Location in the assertion consumer service must match with the configured `consumerServiceUrl`.

The STS supports the initiation of the ECP profile for generating the authentication requests. The ECP endpoint (`/sts/ecp`) must be contacted with the PAOS client as defined in the SAML 2.0 ECP profile specification. The option `requireRequestId` can force the check of the identifier during the assertion validation process, as `requestIdLifetime` controls how long the authentication request identifier is valid after the initiation of the ECP profile.

X.509 certificate

The CMP protocol is supported for the communication with the online CA. STS has been successfully tested to be compatible with the open source implementation called EJBCA, using RA mode of its CMP support.

VOMS certificate

The certificates that are issued by the STS must be already registered to the requested VO. The VO must also trust the CA certificate of the CA behind the STS.

Also, a corresponding vomses file must exist in the directory configured with the `vomses` configuration option.

This topic: EMI > STSConfiguration
Topic revision: r4 - 08-Feb-2013 - HenriMikkonen

STS: Operation

Starting & Stopping the Service

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

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

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

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

Service Information

Service Endpoints

This service contains the following endpoint URLs:

- `https://sts.example.org:8443/sts/wstrust` - This endpoint is the recipient of the WS-Trust request messages
- `https://sts.example.org:8443/sts/ecp` - This endpoint is the recipient of SAML 2.0 ECP profile initiation requests

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/sts/logging.xml`.

Enable Debug Logging

To enable debug logging follow:

1. Locate the line that contains `logger name="org.glite.sts"`
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 > STSOperation
Topic revision: r1 - 18-Jan-2013 - HenriMikkonen