Table of Contents

Trustmanager authentication and certificate handling library

Description

TrustManager is an implementation of the java TrustManager interface with implementation of cert path checking, grid namespace restrictions and dynamic loading of CA certs, credentials, CRLs and namespace restrictions. Also provided is integration into tomcat, axis and axis2. There are many utility classes and methods for certificate and proxy creation and handling in trustmanager. It can be used both in the server side for the server ssl handler and on the client side for the opening of ssl connections.

Restructuring

In version 3.0.3 the tomcat integration classes were moved from trustmanager to trustmanager-tomcat. Thus the trustmanager becomes the main library that has most of the code and functionality. The trustmanager-tomcat has the classes needed to integrate the trustmanager to tomcat. Also in the trustmanager 3.0.3 version the axis related code was moved to trustmanager-axis component. This way the basic code resides in trustmanager, tomcat integration in trustmanager-tomcat and axis integration code in trustmanager-axis. There is also the trustmanager-test that has a test servlet and web application, which are used also for trustmanger certification. There is trustmanager-axis2 component that has classes to integrate trustmanager with axis2. Trustmanager-test-certs is a component that has test certificate generation scripts, that are used for example to test and certify trustmanager.

Supported platforms

The Trustmanager (at version 3.1.3) is supported on Scientific Linux 5 and 6. Packages are available also on Debian and code works, but installation system isn't available at the moment. Trustmanager can be used with Tomcat with the trustmanager-tomcat plugin, with Axis (e.g. 1.4) with trustmanager-axis plugin and with axis2 with trustmanager-axis2 plugin. Trustmanager has also been used with Jetty (See Argus for example).

Manual usage with Tomcat

This is just for testing purposes, in real systems the Trustmanager is configured using yaim, espacially the yaim-core method configure-secure-tomcat.

First of all make sure you have the host credentials! (hostcert.pem and hostkey.pem in /etc/grid-security). Also make sure you have all the necessary CA certificates, CRLs and possibly namespace definitions in /etc/grid-security/certificates.

Get and install tomcat. Get and install the trustmanager and trustmanager-tomcat rpms from one emi repository, these depend on bouncycastle and log4j, so you need them too.

Run:

```
export CATALINA_HOME=<path to tomcat home directory>
```

Edit the /var/lib/glite-security-trustmanager/config.properties and run:

```
/opt/glite/etc/glite-security-trustmanager/configure.sh
```

And if everything is ok, run:

```
Service tomcat5 start
```

To check that the tomcat starts up nicely you can check the logs:

```
less /opt/glite/externals/tomcat-5.0.28/logs/catalina.out
```

```
less /opt/glite/externals/tomcat-5.0.28/logs/glite-security-trustmanager.log
```

If the installation was successful the trustmanager log should be an empty file, but it should exist.

Client Integration

In the program

The axis web service integration:

On the service you'll need to include bcprov, log4j, (axis,) security.trustmanager and security.util-java jars to your webapp and call org.glite.security.util.axis.InitSecurityContext.init() in your code to set up the org.glite.security.SecurityContext. After that you can use the org.glite.security.SecurityInfoContainer to get a class that implements SecurityInfo interface and there you can find the cert chain and user DN. the security.util-java rpm contains javadocs for the securitycontext/info stuff.

On the startup

You'll have to add the BoncyCastle (bcprov), log4j, trustmanger and util-java jars to the classpath.

After that you will have to set socket factory the Axis uses to create sockets by command line option -Daxis.socketSecureFactory=org.glite.security.trustmanager.axis.AXISSocketFactory

You also have to set up the credentials with the following options: either using user proxy: -DgridProxyFile=/tmp/x509up_u500

or user's real credentials: -DsslCertFile=/the/path/to/usercert.pem -DsslKey=/the/path/to/userkey.pem -DsslKeyPasswd=password

These are the minimum settings.

Useful is also the setting: -DtrustStoreDir=/etc/grid-security/certificates

Which defines where to get the trustanchors, their crls and namespace definitions.

Notes

Important setting is the crlUpdateInterval. It defines the interval at which the CRL files are polled for changes. Setting it to 0 will prevent the updater thread being started. It is important on clients that use several user credentials for connecting to services to prevent the crl updater from being started as it will lead to memory leaks and unnecessary threads being run. On the client side there is relly no need to start the updater thread.

Note on intervals: format: n {s,m,h,d} where n is number and (s=seconds, m=minutes, h=hours, d=days). Only first letter is considered, so for clarity's sake full word is recommended. (for example "2 hours")

Properties

This is the list of configuration variables the trustmanager takes during the ssl context creation. They can be given through properties when generating the contextwrapper, in the thread local properties or through system properties (thus also through the -D switch during the java command execution).

trustStoreDir

- the directory containing all the CA certificates, CRLs and namespace definitions.If neither this sslCAStore nor sslCAFiles are defined, /etc/grid-security/certificates is assumed as the value for this variable. The directory contains the CA files named as the hex of 4 least significant bytes of the MD5 hash of the CA subject name in binary ASN1 format and the suffix of a number e.g. 1dab23fe.1. The CRLs have the same filename except the suffix is prepended by a 'r' e.g. 1dab23fe.r1 corresponds to the previous CA. The namespace files are named with the same filename but with a suffix .namespace or .signin_policy depending on whether the namespace is defined using International Grid Trust Federation format in the first case or using globus format in the latter case. The IGTF format is used if both are present.

sslCertFile

- the file containing the local certificate.

sslKey

- The file containing the local private key.

sslKeyPasswd

- The password to access the private key if it is encrypted.

gridProxyFile

- Defines the file that contains the gridproxy to use for authentication. Also used for kerberos credentials where the user cert and key are in the same file in pem format.

credentialsUpdateInterval

- The time interval used to reload the local credentials from filesystem into memory. (default 0 (=never))

sslCertStore

- The keystore that holds the local credentials.

sslCertStoreType

- The keystore type "JKS" and "PKCS12" are supported (default "JKS")

sslCertStorePasswd

- The password to use when accessing the keystore

sslCAFiles (deprecated)

- The files containing the CA certificates

sslCAStore

- The keystore that holds the CA certificates

sslCAStoreType

- The keystore type of the CA keystore "JKS" and "PKCS12" are supported (default "JKS")

sslCAStorePasswd

- The password to use to access the CA keystore

crlFiles (deprecated)

- the CRL files to use (default /etc/grid-security/certificates/*.r0)

crlEnabled (deprecated, assumed true)

- defines whether the system tries to use CRLs (default true)

crlRequired

- defines what happens when CRL is not fournd for a CA (default true) - if true, all certificates from a CA that doesn't have a valid CRL are rejected - if false, all certificates from a CA that doesn't have a valid CRL are accepted

crlUpdateInterval

- the interval to poll updated of the CRL, CA or namespace files from the filesystem and reaload them into the memory cache in case of changes. If the interval is set to 0 the update thread is not started. (default "0 hours")

log4jConfFile

- The file to use to configure the log4j logging facility.

logFile

- The file to log to. To be used instead log4jConfFile if defaults are fine, but log file needs to be configured.

sslProtocol

- The ssl protocol to use (default "TLSv1") (SSLv3, SSLv2, SSLv2HELLO also supported)

sslConfigFile

- Given just this option, the actual configuration options are read from this file.

sslTimeout

- defines the read timeout in milliseconds

sslConnectTimeout

- defines the connect timeout in milliseconds

internalOverrideExpirationCheck

- only for testing, overrides the expiration check when loading credentials.

Changes

see TrustManagerChanges

Testing

CA testing

- put empty CA file (any . number file, eg. 9182374918.0) /etc/grid-security/certificates. Tomcat should start, and java clients work, warning in log file.
- renamve one crl file into CA .number file in /etc/grid-security/certificates. Tomcat should start, and java clients work, warning in log file.
- put expired CA file into /etc/grid-security/certificates. Tomcat should start, and java clients work, warning in log file.

CRL testing

- CA without CRL. The certs from the CA are rejected.
- CA with expired CRL. The certs from the CA are rejected.
- CA with invalid CRL. The certs from the CA are rejected.
- CA without CRL at the start of tomcat. After adding a valid CRL and waiting the update period (default 2h), the certs from the CA start to work.
- CA with expired CRL at the start of tomcat. After adding a valid CRL and waiting the update period (default 2h), the certs from the CA start to work.
- CA with invalid CRL at the start of tomcat. After adding a valid CRL and waiting the update period (default 2h), the certs from the CA start to work.
- CA with valid CRL. the certs from the CA are accepted.

Certificate testing

- v1 x509 certs are accepted.
- v3 x509 certs are accepted.
- not yet valid certificate is rejected.
- expired certificate is rejected.

Namespace testing

- CA with only .namespaces file. Certs following the namespace are accepted, certs outside the namespace are rejected.
- CA with only .signing_policy file. Certs following the policy are accepted, certs outside the policy are rejected.
- CA with both .namespaces and .signing_policy files. Certs following the namespace are accepted, certs outside the namespace are rejected, .signing_policy file ahs no effect.
- CA without .namespaces and .signing_policy file. All valid certificates irrespective of the DN are accepted.

Proxy testing

- Valid legacy proxies are accepted.

- Valid draft proxies are accepted.
- Valid rfc proxies are accepted.
- Valid proxies of depth of 15 are accepted.
- Proxy with invalid DN (not starting with the DN of the issuer and adding a CN component) are rejected.
- Proxy with invalid signature (for example signed by a private key of unrelated cert) are rejected.
- Proxy with invalid DN/singature in the middle of the chain is rejected.
- not yet valid proxy is rejected.
- expired proxy is rejected.

Log file testing

- Not yet valid certificate error message is correct.
- Expired certificate error message is correct.
- Error messages from all the above rejected certificates are correct.
- Timestamps are correct.
- Every valid cert produces a line in the logs with DN of the user.

see TrustmanagerTestplan. see TrustManagerTestReports. see tests in cvs.

Tomcat classloader change

see EGEETomcatTrustmanager.

Funding acknowledgement

Trustmanager has been developed with funding from Helsinki Institute of Physics and the following EU Projects:

- European Data Grid (EDG): initial development
- Enabling Grids in EuropE I, II and III (EGEE): further development and support
- European Middleware Initiative (EMI): Support and porting

This topic: EGEE > TrustManager
Topic revision: r11 - 2012-06-04 - JoniHahkala