

# GT4 Java WS A&A User's Guide

DRAFT

---

## GT4 Java WS A&A User's Guide

### Introduction

**Authorization.** Users who run clients can programmatically set up the authorization scheme to be enforced on a per invocation basis. The properties and configuration information required depends on the configured authorization scheme.

**Authentication & message-level security.** Typical user configuration deals with configuring authentication mechanisms and credentials for the clients. These could be client applications, including command line clients or client configuration within services that contact other services. There are multiple mechanisms for doing this:

- Command line options (these are application-specific)
- Client security descriptors
- CoG properties
- Environment variables
- Relying on default behavior. The only default behaviors available concern the proxy file and trusted certificates locations.

More information on these mechanisms can be found in the [public interface guide](#).

---

## Table of Contents

1. Client-side authorization .....	1
2. Configuring client authentication and message/transport security .....	2
1. Interface introduction .....	2
2. Syntax of the interface .....	4
3. Debugging .....	9
1. Logging in Java WS Core .....	9
4. Troubleshooting .....	11
1. Credential Troubleshooting .....	11
2. Error Messages For Java WS A&A .....	14
Glossary .....	17

DRAFT

## List of Tables

2.1. Client side security properties .....	5
4.1. Credential Errors .....	12
4.2. Java WS A&A Errors .....	15

DRAFT

# Chapter 1. Client-side authorization

The client side authorization can be configured for each invocation.

- Security descriptors using files. Refer to [Section 1, “Configuring Using Files”](#), specifically [Section 1.2.2, “Configuring authorization mechanism”](#).
- Security descriptors programmatically. Refer to [Section 2, “Configuring Programmatically”](#)
- Properties on the Stub. Refer to [Section 2.1, “Configuring client-side authorization on the stub”](#)

To write and configure custom authorization mechanism refer to [Section 2.2, “Writing custom client-side authorization scheme”](#).

If no authorization mechanism has been specified, HostOrSelf authorization is used. In this scheme host authorization is tried first, if it fails, self authorization is attempted

DRAFT

# Chapter 2. Configuring client authentication and message/transport security

## 1. Interface introduction

Client-side security is set up by either setting individual properties on the `javax.xml.rpc.Stub` object used for the web service method invocation or by setting properties on a client-side security descriptor object, which in turn is propagated to client-side security handlers by making it available as a stub object property. Here are examples of the two approaches:

- Setting a property on the stub:

```
// Create endpoint reference
EndpointReferenceType endpoint = new EndpointReferenceType();
// Set address of service
String counterAddr =
    "http://localhost:8080/wsrf/services/CounterService";
// Get handle to port
CounterPortType port =
    locator.getCounterPortTypePort(endpoint);
// set client authorization to self
((Stub)port)._setProperty(Constants.AUTHORIZATION,
    SelfAuthorization.getInstance());
```

- Setting properties using a client descriptor:

```
// Client security descriptor file
String CLIENT_DESC =
    "org/globus/wsrf/samples/counter/client/client-security-config.xml";
// Create endpoint reference
EndpointReferenceType endpoint = new EndpointReferenceType();
// Set address of service
String counterAddr =
    "http://localhost:8080/wsrf/services/CounterService";
// Get handle to port
CounterPortType port =
    locator.getCounterPortTypePort(endpoint);
//Set descriptor on Stub
((Stub)port)._setProperty(Constants.CLIENT_DESCRIPTOR_FILE, CLIENT_DESC);
```

The descriptor file is described in detail in [Chapter 1, Security Descriptors Introduction](#).

**Note**

If the client needs to use transport security, the following API must be used to register the Axis transport handler for https:

```
import org.globus.axis.util.Util;
static {
    Util.registerTransport();
}
```

DRAFT

## 2. Syntax of the interface

DRAFT

**Table 2.1. Client side security properties**

Number	Task	Stub Configuration	De- Co- tion
1.	Allows for configuration of credentials for authentication.	Property: <code>org.globus.axis.gsi.GSIConstants.GSI_CREDENTIALS</code> Value equals the Instance of <code>org.ietf.jgss.GSSCredential</code> .	Sec- tion "C ing tial
2.	Allows for configuring client-side authorization.	Property: <code>org.globus.wsrf.security.Constants.AUTHORIZATION</code> Value equals the Instance of <code>org.globus.wsrf.security.authorization.Authorization</code> If GSI Secure Transport or GSI Secure Conversation is used, the value should be an instance of <code>org.globus.gsi.gssapi.auth.Authorization</code> . But this translation is done automatically by the toolkit.	Rel- Sec- tion "C ing atic ani
3.	Enable GSI Secure Conversation with specified message protection level.	1. Property: <code>org.globus.wsrf.security.Constants.GSI_SEC_CONV</code> Values equal one of the following: <ul style="list-style-type: none"> <li>• <code>Constants.ENCRYPTION</code></li> <li>• <code>Constants.SIGNATURE</code></li> </ul> 2. Property: <code>org.globus.wsrf.security.Constants.GSI_SEC_CONV_SECREPLY_UNNECESSARY</code> If the value is set to <code>Boolean.TRUE</code> , the GSI Secure conversation protection is not required in the reply message. By default, if the request was secured with GSI Secure Conversation, the response is also required to have the same protection.           3. Property: You can set the SOAP Actor of the GSI signed/encrypted SOAP message by using the <code>gssActor</code> property. We recommend that you <i>not</i> do this unless you <i>really</i> know what you are doing.	Rel- tion "C ing cur ver

4.	Sets the GSI delegation mode. <i>Used for GSI Secure Conversation only.</i> If limited or full delegation is chosen, then some form of client-side authorization needs to be done (i.e client-side authorization cannot be set to none).	<p>Property:</p> <pre>org.globus.axis.gsi.GSIConstants.GSI_MODE</pre> <p>Value equals one of following:</p> <ol style="list-style-type: none"> <li>1. <code>GSIConstants.GSI_MODE_NO_DELEG</code>: No delegation is performed.</li> <li>2. <code>GSIConstants.GSI_MODE_LIMITED_DELEG</code>: Limited delegation is performed.</li> <li>3. <code>GSIConstants.GSI_MODE_FULL_DELEG</code>: Full delegation is performed.</li> </ol>	Rel tion "C ing cur ver
5.	Enables GSI Secure Transport with some protection level.	<p>Property:</p> <pre>org.globus.gsi.GSIConstants.GSI_TRANSPORT</pre> <p>Values equal one of the following:</p> <ul style="list-style-type: none"> <li>• <code>Constants.ENCRYPTION</code></li> <li>• <code>Constants.SIGNATURE</code></li> </ul>	Rel tion "C ing cur por
6.	Enables anonymous authentication. <i>This option only applies to GSI Secure Conversation and GSI Transport.</i>	<p>Property:</p> <pre>org.globus.wsrp.security.Constants.GSI_ANONYMOUS</pre> <p>Value equals one of following:</p> <ol style="list-style-type: none"> <li>1. <code>Boolean.FALSE</code>: Anonymous authentication is disabled.</li> <li>2. <code>Boolean.TRUE</code>: Anonymous authentication is enabled.</li> </ol>	Rel tion "C ing cur por

7.	Enable GSI Secure Message with specified message protection level.	<p>1. Property:  <code>org.globus.wsrp.security.Constants.GSI_SEC_MSG</code></p> <p>Values equal one of the following:</p> <ul style="list-style-type: none"> <li>• <code>Constants.ENCRIPTION</code></li> <li>• <code>Constants.SIGNATURE</code></li> </ul> <p>2. Property:  <code>org.globus.wsrp.security.Constants.GSI_SEC_MSG_SECREPLY_UNNECESSARY</code></p> <p>If the value is set to <code>Boolean.TRUE</code>, the GSI Secure Message protection is not required in the reply message. By default, if the request was secured with GSI Secure Message, the response is also required to have the same protection.</p> <p>3. Property:  <code>org.globus.wsrp.security.Constants.GSI_SEC_MSG_SINGLECERT</code></p> <p>If the value is set to <code>Boolean.TRUE</code>, only a single certificate is used for the GSI Secure Message request. By default, the whole certificate chain is sent.</p> <p>4. Property:</p> <p>You can set the SOAP Actor of the signed message using the <code>x509Actor</code> property, but we do <i>not</i> recommend this unless you know what you are doing.</p>	Ret tion "C ing cur sag
8.	Enable WS-Security username/password authentication.	<p>Properties:</p> <p><code>org.globus.wsrp.security.Constants.USERNAME</code></p> <p>Value equals the username.</p> <p><code>org.globus.wsrp.security.Constants.PASSWORD</code></p> <p>Value equals the password.</p>	Ret tion "C ing nar wo

9.	Sets the credential that is used to encrypt the message (typically, the recipient's <i>public key</i> ). <i>Used for GSI Secure Message only.</i>	<p>Property:</p> <pre>org.globus.wsrfl.impl.security.authentication     .Constants.PEER_SUBJECT</pre> <p>Value equals the instance of <code>javax.security.auth.Subject</code>.</p> <p>The credential object needs to be wrapped in <code>org.globus.wsrfl.impl.security.authentication.encrypted</code> and added to the set of public credentials of the Subject object.</p> <p>For example, if <code>publicKeyFilename</code> was the file that had the recipient's public key:</p> <pre>Subject subject = new Subject(); X509Certificate serverCert =     CertUtil.loadCertificate(publicKeyFilename); EncryptionCredentials encryptionCreds =     new EncryptionCredentials(         new X509Certificate[] { serverCert }); subject.getPublicCredentials().add(encryptionCreds); stub._setProperty(Constants.PEER_SUBJECT, subject);</pre>	Rel tion "C ing cur sag
10.	Sets the trusted certificates location.	<p>Property:</p> <pre>org.globus.wsrfl.security.TRUSTED_CERTIFICATES</pre> <p>Value should be a comma-separated list of directories and file names.</p>	Rel tion "C ing cre " -
11.	Sets the SAML Authorization Assertion to embed in SOAP Header.	<p>Property:</p> <pre>org.globus.wsrfl.impl.security.authentication.Constants.SAML_AUTHZ_ASSERTION</pre> <p>Value should be an instance of <code>org.opensaml.SAMLAssertion</code>.</p>	Car con usi des

# Chapter 3. Debugging

## 1. Logging in Java WS Core

The following information applies to Java WS Core and all services built on Java WS Core.

Java WS Core server side has two types of loggers. One logger is used for development logging and by default writes to standard out. The other logger includes system administration information and is [CEDPs best practices](#)<sup>1</sup> compliant.

On client side, only developer logging is available and is configured using `log4j.properties`.

### 1.1. Development Logging in Java WS Core

The following information applies to Java WS Core and those services built on it.

Logging in the Java WS Core is based on the [Jakarta Commons Logging](#)<sup>2</sup> API. Commons Logging provides a consistent interface for instrumenting source code while at the same time allowing the user to plug-in a different logging implementation. Currently we use [Log4j](#)<sup>3</sup> as a logging implementation. Log4j uses a separate configuration file to configure itself. Please see Log4j documentation for details on the [configuration file format](#)<sup>4</sup>.

#### 1.1.1. Configuring server side developer logs

Server side logging can be configured in `$GLOBUS_LOCATION/container-log4j.properties`, when the container is stand alone container. For tomcat level logging, refer to [Logging for Tomcat](#)<sup>5</sup>. The logger `log4j.appender.A1` is used for developer logging and by default writes output to the system output. By default it is set for all warnings in the Globus Toolkit package to be displayed.

Additional logging can be enabled for a package by adding a new line to the configuration file. Example:

```
#for debug level logging from org.globus.package.FooClass
log4j.category.org.globus.package.name.FooClass=DEBUG
#for warnings from org.some.warn.package
log4j.category.org.some.warn.package=WARN
```

#### 1.1.2. Configuring client side developer logs

Client side logging can be configured in `$GLOBUS_LOCATION/log4j.properties`. The logger `log4j.appender.A1` is used for developer logging and by default writes output to the system output. By default it is set for all warnings in the Globus Toolkit package to be displayed.

---

<sup>1</sup> <http://cedps.net/index.php/LoggingBestPractices>

<sup>2</sup> <http://jakarta.apache.org/commons/logging/>

<sup>3</sup> <http://logging.apache.org/log4j/>

<sup>4</sup> [http://logging.apache.org/log4j/docs/api/org/apache/log4j/PropertyConfigurator.html#doConfigure\(java.lang.String,org.apache.log4j.spi.LoggerRepository\)](http://logging.apache.org/log4j/docs/api/org/apache/log4j/PropertyConfigurator.html#doConfigure(java.lang.String,org.apache.log4j.spi.LoggerRepository))

<sup>5</sup> <http://tomcat.apache.org/tomcat-5.5-doc/logging.html>

## 1.2. Configuring system administration logs

The specific logger to edit will be `log4j.logger.sysadmin` in `$GLOBUS_LOCATION/container-log4j.properties`. There you can configure the following properties:

```
log4j.appender.infoCategory=org.apache.log4j.RollingFileAppender
log4j.appender.infoCategory.Threshold=INFO
log4j.appender.infoCategory.File=var/containerLog
log4j.appender.infoCategory.MaxFileSize=10MB
log4j.appender.infoCategory.MaxBackupIndex=2
```

Above implies the logging file is rolling with each file size limited to 10MB and the logging information is stored in `$GLOBUS_LOCATION/var/containerLog`.

## 1.3. Sample log file

The [sample log file](#)<sup>6</sup> contains many log entries for various scenarios in the Java WS container.

---

<sup>6</sup> <http://www.globus.org/toolkit/docs/4.2/4.2.0/common/javawscore/sample-container-log.txt>

# Chapter 4. Troubleshooting

For a list of common errors in GT, see [Error Codes](#).

For information about system administrator logs, see [Chapter 7, Troubleshooting](#).

## 1. Credential Troubleshooting

### 1.1. Credential Errors

The following are some common problems that may cause clients or servers to report that credentials are invalid:

For a list of common errors in GT, see [Error Codes](#).

DRAFT

**Table 4.1. Credential Errors**

<b>Error Code</b>	<b>Definition</b>	<b>Possible Solutions</b>
Your proxy credential may have expired	Your proxy credential may have expired.	Use <code>grid-proxy-info</code> to check whether the proxy credential has actually expired. If it has, generate a new proxy with <code>grid-proxy-init</code> .
The system clock on either the local or remote system is wrong.	This may cause the server or client to conclude that a credential has expired.	Check the system clocks on the local and remote system.
Your end-user certificate may have expired	Your end-user certificate may have expired	Use <code>grid-cert-info</code> to check your certificate's expiration date. If it has expired, follow your CA's procedures to get a new one.
The permissions may be wrong on your proxy file	If the permissions on your proxy file are too lax (for example, if others can read your proxy file), Globus Toolkit clients will not use that file to authenticate.	You can "fix" this problem by changing the permissions on the file or by destroying it (with <code>grid-proxy-destroy</code> ) and creating a new one (with <code>grid-proxy-init</code> ).  <b>Important:</b> However, it is still possible that someone else has made a copy of that file during the time that the permissions were wrong. In that case, they will be able to impersonate you until the proxy file expires or your permissions or end-user certificate are revoked, whichever happens first.
The permissions may be wrong on your private key file	If the permissions on your end user certificate private key file are too lax (for example, if others can read the file), <code>grid-proxy-init</code> will refuse to create a proxy certificate.	You can "fix" this by changing the permissions on the private key file.  <b>Important:</b> However, you will still have a much more serious problem: it is possible that someone has made a copy of your private key file. Although this file is encrypted, it is possible that someone will be able to decrypt the private key, at which point they will be able to impersonate you as long as your end user certificate is valid. You should contact your CA to have your end-user certificate revoked and get a new one.
The remote system may not trust your CA	The remote system may not trust your CA	Verify that the remote system is configured to trust the CA that issued your end-entity certificate. See <a href="#">Installing GT 4.2.0</a> for details.
You may not trust the remote system's CA	You may not trust the remote system's CA	Verify that your system is configured to trust the remote CA (or that your environment is set up to trust the remote CA). See <a href="#">Installing GT 4.2.0</a> for details.
There may be something wrong with the remote service's credentials	There may be something wrong with the remote service's credentials	It is sometimes difficult to distinguish between errors reported by the remote service regarding your credentials and errors reported by the client interface regarding the remote service's credentials. If you cannot find anything wrong with your credentials, check for the same conditions on the remote system (or ask a remote administrator to do so).

## 1.2. Some tools to validate certificate setup

### 1.2.1. Check that the user certificate is valid

```
openssl verify -CApath /etc/grid-security/certificates
-purpose sslclient ~/.globus/usercert.pem
```

### 1.2.2. Connect to the server using s\_client

```
openssl s_client -ssl3 -cert ~/.globus/usercert.pem -key
~/.globus/userkey.pem -CApath /etc/grid-security/certificates
-connect <host:port>
```

Here `<host:port>` denotes the server and port you connect to.

If it prints an error and puts you back at the command prompt, then it typically means that the *server* has closed the connection, i.e. that the server was not happy with the client's certificate and verification. Check the SSL log on the server.

If the command "hangs" then it has actually opened a telnet style (but secure) socket, and you can "talk" to the server.

You should be able to scroll up and see the subject names of the server's verification chain:

```
depth=2 /DC=net/DC=ES/O=ESnet/OU=Certificate Authorities/CN=ESnet Root CA 1
verify return:1
depth=1 /DC=org/DC=DOEGrids/OU=Certificate Authorities/CN=DOEGrids CA 1
verify return:1
depth=0 /DC=org/DC=doegrids/OU=Services/CN=wiggum.mcs.anl.gov
verify return:1
```

In this case, there were no errors. Errors would give you an extra line next to the subject name of the certificate that caused the error.

### 1.2.3. Check that the server certificate is valid

Requires root login on server:

```
openssl verify -CApath /etc/grid-security/certificates -purpose sslserver
/etc/grid-security/hostcert.pem
```

## 2. Error Messages For Java WS A&A

DRAFT

**Table 4.2. Java WS A&A Errors**

Error Code	Definition	Possible Solutions
[JWSSEC-248] Secure container requires valid credentials	This error occurs when <code>globus-start-container</code> is run without any valid credentials. Either a proxy certificate or service/host certificate needs to be configured for the container to start up.	<ol style="list-style-type: none"> <li>1. If you are not looking to start up a container that uses GSI Secure Transport, which is used by the container by default, use <code>globus-start-container -nosec</code>. You will be able to use insecure clients and services. However, this also implies that if you have not configured individual services with credentials, you will not be able to securely access the service.</li> <li>2. If you are running a personal container, generate a proxy certificate with <code>grid-proxy-init</code>. If the proxy certificate is not in the default location, configure the container security descriptor as described in <a href="#">Configuring Container Security Descriptor</a>.</li> <li>3. If you want to use host certificates, configure the container security descriptor as described <a href="#">Configuring Credentials</a>.</li> </ol>
Failed to start container: Container failed to initialize [Caused by: [JWSSEC-250] Failed to load certificate/key file]	This error occurs if the file path to the container certificate and key configured are invalid.	<ol style="list-style-type: none"> <li>1. The path to the container certificate and key are configured in <code>\$GLOBUS_LOCATION/etc/globus_wsrf_core/global_security_descriptor.xml</code>. This file is loaded as described [here - fixme link]. Ensure that the path is correct.</li> </ol>
Failed to start container: Container failed to initialize [Caused by: [JWSSEC-249] Failed to load proxy file]	This error occurs if container proxy file configured is invalid.	<ol style="list-style-type: none"> <li>1. The path to the container proxy certificates are configured in <code>\$GLOBUS_LOCATION/etc/globus_wsrf_core/global_security_descriptor.xml</code>. This file is loaded as described [here - fixme link]. Ensure that the path is correct.</li> </ol>
Failed to start container: Container failed to initialize [Caused by: [JWSSEC-245] Error parsing file: "etc/globus_wsrf_core/global_security_descriptor.xml" [Caused by: ...]	This error occurs if the container security descriptor configured is invalid.	<ol style="list-style-type: none"> <li>1. The container security descriptor should conform to the <a href="#">Container Security Descriptor Schema</a>.<sup>1</sup></li> <li>2. Refer to the "Caused by: " section for details on the specific element that is not correct.</li> </ol>

<sup>1</sup> [http://www.globus.org/toolkit/docs/4.2.0/security/container\\_security\\_descriptor.xsd](http://www.globus.org/toolkit/docs/4.2.0/security/container_security_descriptor.xsd)

Error Code	Definition	Possible Solutions
[JGLOBUS-77] Unknown CA	This error occurs if the CA certificate for the credentials being used is not installed correctly.	<ol style="list-style-type: none"><li data-bbox="805 243 1336 401">1. If this issue occurs on the server side, the container is not configured with CA certificates. The container looks for trusted certificates in the default location as described <a href="#">Java CoG Toolkit FAQ</a><sup>2</sup></li><li data-bbox="805 428 1336 520">2. On the server side, the trusted certificates can be configured as described in <a href="#">Trusted Certificates</a></li><li data-bbox="805 548 1336 640">3. On the client side, trusted certificates can be configured as described in <a href="#">Configuring Trusted Credentials</a></li></ol>

---

<sup>2</sup> <http://www.globus.org/cog/distribution/1.2/FAQ.TXT>

