

GT4 C WS A&A Public Interfaces

GT4 C WS A&A Public Interfaces

Table of Contents

1. APIs	1
1. Programming Model Overview	1
2. Component API	1
2. Services and WSDL	2
1. Secure Conversation Service	2
3. Framework-level Protocols	5
1. WS-Security	5
2. Transport (HTTPS) Security	5
I. Command-line tools	6
globus-credential-delegate	7
4. Domain-specific interface	8
1. Interface introduction	8
2. Syntax of the interface	9
5. Configuring	14
1. Configuration overview	14
2. Syntax of the interface	14
6. Environmental variables	16
1. Environmental variables for C WS A&A	16
A. Errors	17
Glossary	22

List of Tables

1. globus-credential-delegate options	7
4.1. Client side security properties	10
5.1. Configuring server side authentication and message/transport security	15
A.1. C WS A&A Errors	18

Chapter 1. APIs

1. Programming Model Overview

The security programming model differs between the client and server side. The client side model is programmatic in nature, i.e. security-related code is driven by making actual function calls, whereas the server-side model is declarative, i.e. security-related settings are declared in a security descriptor. For more information on the available client side calls see ????. More information about the security descriptor can be found in [Java WS A&A Security Descriptor Framework](#).

2. Component API

- Stable interfaces:
 - `org.globus.wsrp.security.Constants`
 - `org.globus.wsrp.security.SecureResource`
 - `org.globus.wsrp.security.SecurityManager`
 - `org.globus.wsrp.security.SecurityException`
- Less stable interfaces:
 - `org.globus.wsrp.impl.security.descriptor.ClientSecurityDescriptor`
 - `org.globus.wsrp.impl.security.descriptor.ResourceSecurityDescriptor`

Documentation for these interfaces can be found [here](#)¹.

¹ http://www.globus.org/api/javadoc-4.2.1/globus_java_ws_core

Chapter 2. Services and WSDL

1. Secure Conversation Service

1.1. Protocol overview

This service provides a mechanism for generating a security session, i.e the negotiation of a shared secret which may be used to secure a set of subsequent messages. It is based on the [WS-Trust](http://www.ibm.com/developerworks/library/ws-trust/)¹ and [WS-SecureConversation](http://www-106.ibm.com/developerworks/library/ws-secon/)² specifications.

1.2. Operations

- `RequestSecurityToken`: This operation initiates a new security session negotiation. Furthermore, since the actual schema for this message is not unambiguously defined by the specifications, this is the actual schema used:

```
<xs:element name='RequestSecurityToken'>
  <xs:complexType name='RequestSecurityTokenType'>
    <xs:sequence>
      <xs:element ref='wst:TokenType' />
      <xs:element ref='wst:RequestType' />
      <xs:element ref='wst:BinaryExchange' />
    </xs:sequence>
    <xs:attribute name='Context' type='xs:anyURI' />
  </xs:complexType>
</xs:element>
```

```
<xs:element name='RequestSecurityTokenResponse'>
  <xs:complexType name='RequestSecurityTokenResponseType'>
    <xs:sequence>
      <xs:element ref='wst:TokenType' />
      <xs:element ref='wst:RequestType' />
      <xs:element ref='wst:BinaryExchange' />
    </xs:sequence>
    <xs:attribute name='Context' type='xs:anyURI' />
  </xs:complexType>
</xs:element>
```

- `RequestSecurityTokenResponse`: This operation continues a security session negotiation. Furthermore, since the actual schema for this message is not unambiguously defined by the specifications, this is the actual schema used:

```
<xs:element name='RequestSecurityTokenResponse'>
  <xs:complexType name='RequestSecurityTokenResponseType'>
    <xs:sequence>
      <xs:element ref='wst:TokenType' />
      <xs:element ref='wst:RequestType' />
      <xs:element ref='wst:BinaryExchange' />
    </xs:sequence>
    <xs:attribute name='Context' type='xs:anyURI' />
  </xs:complexType>
</xs:element>
```

¹ <http://www.ibm.com/developerworks/library/ws-trust/>

² <http://www-106.ibm.com/developerworks/library/ws-secon/>

```
</xs:complexType>
</xs:element>

<xs:element name='RequestSecurityTokenResponse'>
  <xs:complexType name='RequestSecurityTokenResponseType'>
    <xs:sequence>
      <xs:element ref='wst:TokenType' />
      <xs:element ref='wst:RequestType' />
      <xs:element ref='wst:BinaryExchange'
        minOccurs="0" />
      <xs:element ref='wsc:SecurityContextToken' />
    </xs:sequence>
    <xs:attribute name='Context' type='xs:anyURI' />
  </xs:complexType>
</xs:element>
```

In the above schema, the second RequestSecurityTokenResponse element refers to the final message in the exchange.

1.3. Resource properties

This service has no associated resource properties.

1.4. Faults

Both RequestSecurityToken and RequestSecurityTokenResponse throw the following faults:

- `ValueTypeNotSupportedFault`: This fault indicates that the value type attribute on the binary exchange token element is not supported by the service.
- `EncodingTypeNotSupportedFault`: This fault indicates that the encoding type attribute on the binary exchange token element is not supported by the service.
- `RequestTypeNotSupportedFault`: This fault indicates that the request type specified in the request type element is not supported by the service.
- `TokenTypeNotSupportedFault`: This fault indicates that the token type specified in the token type element is not supported by the service.
- `MalformedMessageFault`: This fault indicates that the message content received by the service does not conform to the expected content. This is necessary since the schema does not give a well defined content model.
- `BinaryExchangeFault`: This fault indicates that a failure occurred during the in the underlying security constant responsible for the session negotiation.
- `InvalidContextIdFault`: This fault indicates that the context id passed in the message is not valid within the context of this service or negotiation.

1.5. WSDL and Schema Definitions

- [WS-Trust WSDL](#)³

³ <http://www-106.ibm.com/developerworks/library/specification/ws-trust/ws-trust.wsdl>

- [WS-Trust XSD](#)⁴
- [WS-SecureConversation XSD](#)⁵
- Secure Conversation WSDL [fixme - link]

⁴ <http://www-106.ibm.com/developerworks/library/specification/ws-trust/ws-trust.xsd>

⁵ <http://www-106.ibm.com/developerworks/library/specification/ws-secon/ws-secureconversation.xsd>

Chapter 3. Framework-level Protocols

1. WS-Security

The framework implements the Web Services Security: SOAP Message Security¹, Web Services Security: Username Token Profile² and Web Services Security: X.509 Token Profile³ specifications.

2. Transport (HTTPS) Security

The transport security solution used by the framework consists of HTTP over SSL/TLS (HTTPS) using X.509 certificates. The path validation step has been augmented to support the Proxy Certificate Profile (RFC3820⁴).

¹ <http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf>

² <http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0.pdf>

³ <http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0.pdf>

⁴ <ftp://ftp.rfc-editor.org/in-notes/rfc3820.txt>

Command-line tools

Name

globus-credential-delegate -- Delegation client

globus-credential-delegate

Tool description

Used to contact a Delegation Factory Service and store a delegated credential. A delegated credential is created and stored in a delegated credential WS-Resource, and the Endpoint Reference(EPR) of the credential is written out to a file for further use.

Command syntax

```
globus-credential-delegate [options] <eprFilename>
```

Table 1. globus-credential-delegate options

[option1]	Enables anonymous authentication. Only supported with transport security or the GSI Secure Conversation authentication mechanism.
[option1]	Specifies the server's <i>certificate</i> file used for encryption. Only needed for the GSI Secure Message authentication mechanism.

Chapter 4. Domain-specific interface

1. Interface introduction

Client-side security is set up by 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);
```



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();
}
```

2. Syntax of the interface

Table 4.1. Client side security properties

Number	Task	Stub Configuration
1.	Allows for configuration of credentials for authentication.	<p>Property:</p> <p><code>org.globus.axis.gsi.GSIConstants.GSI_CREDENTIALS</code></p> <p>Value equals the Instance of <code>org.ietf.jgss.GSSCredential</code>.</p>
2.	Allows for configuring client-side authorization.	<p>Property:</p> <p><code>org.globus.wsrsecurity.Constants.AUTHORIZATION</code></p> <p>Value equals the Instance of <code>org.globus.wsrsecurity.authorization.Authorization</code></p> <p>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.</p>
3.	Enable GSI Secure Conversation with specified message protection level.	<p>1. Property:</p> <p><code>org.globus.wsrsecurity.Constants.GSI_SEC_CONV</code></p> <p>Values equal one of the following:</p> <ul style="list-style-type: none"> • <code>Constants.ENCRYPTION</code> • <code>Constants.SIGNATURE</code> <p>2. Property:</p> <p><code>org.globus.wsrsecurity.Constants.GSI_SEC_CONV_SECREPLY_UNNECESSARY</code></p> <p>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.</p> <p>3. Property:</p> <p>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.</p>

4.	<p>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>	<p>Property: <code>org.globus.axis.gsi.GSIConstants.GSI_MODE</code></p> <p>Value equals one of following:</p> <ol style="list-style-type: none"> 1. <code>GSIConstants.GSI_MODE_NO_DELEG</code>: No delegation is performed. 2. <code>GSIConstants.GSI_MODE_LIMITED_DELEG</code>: Limited delegation is performed. 3. <code>GSIConstants.GSI_MODE_FULL_DELEG</code>: Full delegation is performed.
5.	<p>Enables GSI Secure Transport with some protection level.</p>	<p>Property: <code>org.globus.gsi.GSIConstants.GSI_TRANSPORT</code></p> <p>Values equal one of the following:</p> <ul style="list-style-type: none"> • <code>Constants.ENCRYPTION</code> • <code>Constants.SIGNATURE</code>
6.	<p>Enables anonymous authentication. <i>This option only applies to GSI Secure Conversation and GSI Transport.</i></p>	<p>Property: <code>org.globus.wsrp.security.Constants.GSI_ANONYMOUS</code></p> <p>Value equals one of following:</p> <ol style="list-style-type: none"> 1. <code>Boolean.FALSE</code>: Anonymous authentication is disabled. 2. <code>Boolean.TRUE</code>: Anonymous authentication is enabled.

7.	<p>Enable GSI Secure Message with specified message protection level.</p>	<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"> • <code>Constants.ENCRIPTION</code> • <code>Constants.SIGNATURE</code> <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>
8.	<p>Enable WS-Security username/password authentication.</p>	<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>

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>
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>
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>

Can
con
usin
des

Chapter 5. Configuring

1. Configuration overview

Configuration of service-side security settings can be achieved by using container or service security descriptor. Some of the security configuration, like the credential to use and trusted certificates location, can also be configured using CoG properties or rely on default location. **The preferred way is to provide these settings in a security descriptor.**

The next section provides details on the relevant properties. An overview of the syntax of security descriptors can be found in [Java WS A&A Security Descriptor Framework](#). Available CoG security properties can be found in [Chapter 2, Configuring](#)

2. Syntax of the interface

The following properties are relevant to authentication and message/transport security:

Table 5.1. Configuring server side authentication and message/transport security

Number	Task	Descriptor Configuration	Alternate Configuration
1	Credentials	<u>Container or service descriptor configuration</u> ¹	<ul style="list-style-type: none"> • X509_USER_CERT or <u>CoG Configuration</u>²: User certificate configuration • X509_USER_KEY or <u>CoG Configuration</u>³: User key configuration • X509_USER_PROXY or <u>CoG Configuration</u>⁴: User proxy configuration <p>If no explicit configuration is found, the default proxy is read from /tmp/x509_up_<uid>.</p>
2	Trusted Certificates	<u>Container security descriptor configuration</u> ⁵	<u>CoG Configuration</u> ⁶
3	Limited proxy policy configuration	<u>Container or service descriptor configuration</u> ⁷	None.
4	Replay Attack Window	<u>Container or service descriptor configuration</u> ⁸	None.
5	Replay Attack Filter	<u>Container or service descriptor configuration</u> ⁹	None.
6	Replay timer interval	<u>Container descriptor configuration</u> ¹⁰	None.
7	Context timer interval	<u>Container descriptor configuration</u> ¹¹	None.

¹ <http://www.globus.org/toolkit/docs/4.2/4.2.1/security/wsaajava/wsaajava-secdesc.html#wsaajava-secdesc-configCred>

² <http://www.globus.org/toolkit/docs/4.2/4.2.1/common/javacog/admin/javacog-admin-configuring.html#javacog-admin-configuring-user-certificate>

³ <http://www.globus.org/toolkit/docs/4.2/4.2.1/common/javacog/admin/javacog-admin-configuring.html#javacog-admin-configuring-user-key>

⁴ <http://www.globus.org/toolkit/docs/4.2/4.2.1/common/javacog/admin/javacog-admin-configuring.html#javacog-admin-configuring-user-proxy>

⁶ <http://www.globus.org/toolkit/docs/4.2/4.2.1/common/javacog/admin/javacog-admin-configuring.html#javacog-admin-configuring-trusted-certs>

⁵ <http://www.globus.org/toolkit/docs/4.2/4.2.1/security/wsaajava/wsaajava-secdesc.html#wsaajava-secdesc-container-trusted>

⁷ <http://www.globus.org/toolkit/docs/4.2/4.2.1/security/wsaajava/wsaajava-secdesc.html#wsaajava-secdesc-rejectLimProxy>

⁸ <http://www.globus.org/toolkit/docs/4.2/4.2.1/security/wsaajava/wsaajava-secdesc.html#wsaajava-secdesc-replayAttack>

⁹ <http://www.globus.org/toolkit/docs/4.2/4.2.1/security/wsaajava/wsaajava-secdesc.html#wsaajava-secdesc-replayAttack>

¹⁰ <http://www.globus.org/toolkit/docs/4.2/4.2.1/security/wsaajava/wsaajava-secdesc.html#wsaajava-secdesc-container-replay>

¹¹ <http://www.globus.org/toolkit/docs/4.2/4.2.1/security/wsaajava/wsaajava-secdesc.html#wsaajava-secdesc-container-context>

Chapter 6. Environmental variables

1. Environmental variables for C WS A&A

Refer to [Chapter 2, Configuring](#) for environment variables. Note that the above environment variable [fixme - not clear which envvar you mean] does not supersede any settings provided in security descriptors.

Appendix A. Errors

Table A.1. C WS A&A Errors

Error Code	Definition
<p>ERROR: Couldn't read user key: Bad passphrase key file location: /Users/bester/.globus/userkey.pem</p> <p>globus_credential: Error reading user credential: Can't read credential's private key from PEM OpenSSL Error: pem_lib.c:423: in library: PEM routines, function PEM_do_header: bad decrypt OpenSSL Error: evp_enc.c:509: in library: digital envelope routines, function EVP_DecryptFinal: bad decrypt</p> <p>Use -debug for further information.</p>	<p>Unable to decrypt private key</p>
<p>globus_gsi_gssapi: Error with gss credential handle globus_credential: Valid credentials could not be found in any of the possible locations specified by the credential search order. Valid credentials could not be found in any of the possible locations specified by the credential search order.</p> <p>Attempt 1 globus_credential: Error reading host credential globus_sysconfig: Error with certificate filename globus_sysconfig: Error with certificate filename globus_sysconfig: File is not owned by current user: /etc/grid-security/hostcert.pem is not owned by current user</p> <p>Attempt 2 globus_credential: Error reading proxy credential globus_sysconfig: Could not find a valid proxy certificate file location globus_sysconfig: Error with key filename globus_sysconfig: File does not exist: /tmp/x509up_u501 is not a valid file</p> <p>Attempt 3 globus_credential: Error reading user credential globus_credential: Key is password protected: GSI does not currently support password protected private keys. OpenSSL Error: pem_lib.c:401: in library: PEM routines, function PEM_do_header: bad password read</p>	<p>No user proxy could be found</p>
<p>globus_gsi_gssapi: Error with GSI credential globus_gsi_gssapi: Error with gss credential handle globus_credential: Error with credential: The proxy credential: /tmp/x509up_u1499 with subject: /DC=org/DC=example/DC=grid/OU=People/CN=Joe User/CN=1235439010 expired 44 minutes ago.</p>	<p>Proxy has expired.</p>

Error Code	Definition
<p>globus_xio: The GSI XIO driver failed to establish a secure connection. The failure occurred during a handshake read. globus_xio: An end of file occurred</p>	<p>Communication disrupted during SSL handshake</p>
<p>globus_gsi_gssapi: Unable to verify remote side's credentials globus_gsi_gssapi: Unable to verify remote side's credentials: Couldn't verify the remote certificate OpenSSL Error: s3_pkt.c:1052: in library: SSL routines, function SSL3_READ_BYTES: sslv3 alert bad certificate SSL alert number 42</p>	<p>Unable to verify remote certificate. Often a clock-synchronization problem where the service clock is behind that of the client.</p>
<p>OpenSSL Error: s3_clnt.c:894: in library: SSL routines, function SSL3_GET_SERVER_CERTIFICATE: certificate verify failed globus_gsi_callback_module: Could not verify credential globus_gsi_callback_module: The certificate is not yet valid: Cert with subject: /DC=org/DC=example/DC=grid/OU=People/CN=Joe User/CN=464555355 is not yet valid- check clock skew between hosts.</p>	<p>Unable to verify remote certificate. Often a clock-synchronization problem where the client clock is behind that of the service.</p>

Error Code	Definition
<pre> globus_gsi_callback_module: Error with signing policy globus_sysconfig: Error getting signing policy file globus_sysconfig: File does not exist: /etc/grid-security/certificates/2b0e42b2.signing_policy is not a valid file </pre>	<p>The service's certificate is not trusted by the client</p>
<pre> globus_gsi_callback_module: Could not verify credential globus_gsi_callback_module: Error with signing policy globus_gsi_callback_module: Error in OLD GAA code: CA policy violation: <no reason given> </pre>	<p>Service certificate is not trusted because the CA signing policy does not trust the CA to sign the subject name of the certificate.</p>
<pre> Error: globus_soap_message_module: SOAP Fault Fault code: Client Fault string: globus_handler_ws_secure_message: Server Request handling failed globus_handler_ws_secure_message: Failed to verify the message: Unable to get Security header element from message attributes. </pre>	<p>The client sent a request to a service which message security without properly invoking the security handlers</p>

Error Code	Definition
<p>Error: globus_soap_message_module: SOAP Fault Fault code: Client Fault string: globus_soap_message_module: Loaded message handlers do not understand required header element: {http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd}Security</p>	<p>The client sent a request protected with message-level security but the server did not understand the required security headers</p>

Glossary

C

certificate

A public key plus information about the certificate owner bound together by the digital signature of a CA. In the case of a CA certificate, the certificate is self signed, i.e. it was signed using its own private key.

P

public key

The public part of a key pair used for cryptographic operations (e.g. signing, encrypting).