

GT 4.0: Security: Delegation Service

GT 4.0: Security: Delegation Service

Table of Contents

1. Key Concepts	1
1. Overview	1
2. Conceptual Details	1
3. Related Documents	4
2. Admin Guide	8
1. Introduction	8
2. Building and Installing	8
3. Configuring	8
4. Deploying	10
5. Testing	10
6. Security Considerations	10
7. Troubleshooting	11
3. User's Guide	12
1. Introduction	12
2. Command-line tools	12
3. Graphical user interfaces	12
4. Troubleshooting	12
4. Developer's Guide	14
1. Introduction	14
2. Before you begin	14
3. Architecture and design overview	15
4. Public interface	18
5. Usage scenarios	18
6. Tutorials	19
7. Debugging	19
8. Troubleshooting	19
9. Related Documentation	20
5. Fact Sheet	21
1. Brief component overview	21
2. Summary of features	21
3. Usability summary	21
4. Backward compatibility summary	21
5. Technology dependencies	21
6. Tested platforms	22
7. Associated standards	22
8. For More Information	22
6. Public Interfaces	23
1. Semantics and syntax of APIs	23
2. Semantics and syntax of the WSDL	23
3. Command-line tools	24
4. Overview of Graphical User Interface	24
5. Semantics and syntax of domain-specific interface	24
6. Configuration interface	26
7. Environment variable interface	27
7. Quality Report	28
1. Test coverage reports	28
2. Code analysis reports	28
3. Outstanding bugs	28
4. Bug Fixes	28
5. Performance reports	29
8. Migrating Guide	30

1. Migrating from GT2	30
2. Migrating from GT3	30
I. Delegation Service Reference	31
globus-credential-delegate	32
globus-credential-refresh	33
9. 4.0.8 Release Notes	34
1. Introduction	34
2. Changes Summary	34
3. Bug Fixes	34
4. Known Problems	34
5. For More Information	34
10. 4.0.7 Release Notes	35
1. Introduction	35
2. Changes Summary	35
3. Bug Fixes	35
4. Known Problems	35
5. For More Information	35
11. 4.0.6 Release Notes	36
1. Introduction	36
2. Changes Summary	36
3. Bug Fixes	36
4. Known Problems	36
5. For More Information	36
12. 4.0.5 Release Notes	37
1. Introduction	37
2. Changes Summary	37
3. Bug Fixes	37
4. Known Problems	37
5. For More Information	37
13. 4.0.4 Release Notes	38
1. Introduction	38
2. Changes Summary	38
3. Bug Fixes	38
4. Known Problems	38
5. For More Information	38
14. 4.0.3 Release Notes	39
1. Introduction	39
2. Changes Summary	39
3. Bug Fixes	39
4. Known Problems	39
5. For More Information	39
15. 4.0.2 Release Notes	40
1. Introduction	40
2. Changes Summary	40
3. Bug Fixes	40
4. Known Problems	40
5. For More Information	40
16. 4.0.1 Release Notes	41
1. Introduction	41
2. Changes Summary	41
3. Bug Fixes	41
4. Known Problems	41
5. For More Information	41
17. 4.0.0 Release Notes	42

1. Component Overview	42
2. Feature Summary	42
3. Changes Summary	42
4. Bug Fixes	43
5. Known Problems	43
6. Technology Dependencies	43
7. Tested Platforms	44
8. Backward Compatibility Summary	44
9. For More Information	44
GT 4.0 Security Glossary	45

List of Tables

1. globus-credential-delegate options	32
2. globus-credential-refresh options	33

Chapter 1. GT 4.0 Security: Key Concepts

1. Overview

GSI uses public key cryptography (also known as asymmetric cryptography) as the basis for its functionality. Many of the terms and concepts used in this description of GSI come from its use of public key cryptography.

For a good overview of GSI contained in the Web Services-based components of GT4, see [Globus Toolkit Version 4 Grid Security Infrastructure: A Standards Perspective](#)¹.

A reference for detailed information about public key cryptography is available in the book [Handbook of Applied Cryptography](#)², by A. Menezes, P. van Oorschot, and S. Vanstone, CRC Press, 1996. [Chapter 8](#)³ of this book deals exclusively with public key cryptography.

The primary motivations behind GSI are:

- The need for secure communication (authenticated and perhaps confidential) between elements of a computational Grid.
- The need to support security across organizational boundaries, thus prohibiting a centrally-managed security system.
- The need to support "single sign-on" for users of the Grid, including delegation of credentials for computations that involve multiple resources and/or sites.

2. Conceptual Details

2.1. Public Key Cryptography

The most important thing to know about public key cryptography is that, unlike earlier cryptographic systems, it relies not on a single key (a password or a secret "code"), but on two keys. These keys are numbers that are mathematically related in such a way that if either key is used to encrypt a message, the other key must be used to decrypt it. Also important is the fact that it is next to impossible (with our current knowledge of mathematics and available computing power) to obtain the second key from the first one and/or any messages encoded with the first key.

By making one of the keys available publicly (a public key) and keeping the other key private (a [private key](#)⁴), a person can prove that he or she holds the private key simply by encrypting a message. If the message can be decrypted using the public key, the person must have used the private key to encrypt the message.

Important: It is critical that private keys be kept private! Anyone who knows the private key can easily impersonate the owner.

2.2. Digital Signatures

Using public key cryptography, it is possible to digitally "sign" a piece of information. Signing information essentially means assuring a recipient of the information that the information hasn't been tampered with since it left your hands.

¹ GT4-GSI-Overview.pdf

² <http://www.cacr.math.uwaterloo.ca/hac/>

³ <http://www.cacr.math.uwaterloo.ca/hac/about/chap8.pdf>

⁴ #priv-key

To sign a piece of information, first compute a mathematical hash of the information. (A hash is a condensed version of the information. The algorithm used to compute this hash must be known to the recipient of the information, but it isn't a secret.) Using your private key, encrypt the hash, and attach it to the message. Make sure that the recipient has your public key.

To verify that your signed message is authentic, the recipient of the message will compute the hash of the message using the same hashing algorithm you used, and will then decrypt the encrypted hash that you attached to the message. If the newly-computed hash and the decrypted hash match, then it proves that you signed the message and that the message has not been changed since you signed it.

2.3. Certificates

A central concept in GSI authentication is the *certificate*. Every user and service on the Grid is identified via a certificate, which contains information vital to identifying and authenticating the user or service.

A GSI certificate includes four primary pieces of information:

- A subject name, which identifies the person or object that the certificate represents.
- The public key belonging to the subject.
- The identity of a Certificate Authority (CA) that has signed the certificate to certify that the public key and the identity both belong to the subject.
- The digital signature of the named CA.

Note that a third party (a CA) is used to certify the link between the public key and the subject in the certificate. In order to trust the certificate and its contents, the CA's certificate must be trusted. The link between the CA and its certificate must be established via some non-cryptographic means, or else the system is not trustworthy.

GSI certificates are encoded in the X.509 certificate format, a standard data format for certificates established by the Internet Engineering Task Force (IETF). These certificates can be shared with other public key-based software, including commercial web browsers from Microsoft and Netscape.

2.4. Mutual Authentication

If two parties have certificates, and if both parties trust the CAs that signed each other's certificates, then the two parties can prove to each other that they are who they say they are. This is known as *mutual authentication*. GSI uses the Secure Sockets Layer (SSL) for its mutual authentication protocol, which is described [below](#)⁵. (SSL is also known by a new, IETF standard name: Transport Layer Security, or TLS.)

Before mutual authentication can occur, the parties involved must first trust the CAs that signed each other's certificates. In practice, this means that they must have copies of the CAs' certificates--which contain the CAs' public keys--and that they must trust that these certificates really belong to the CAs.

To mutually authenticate, the first person (*A*) establishes a connection to the second person (*B*).

To start the authentication process, *A* gives *B* his certificate.

The certificate tells *B* who *A* is claiming to be (the identity), what *A*'s public key is, and what CA is being used to certify the certificate.

⁵ #s-security-key-delegation

B will first make sure that the certificate is valid by checking the CA's digital signature to make sure that the CA actually signed the certificate and that the certificate hasn't been tampered with. (This is where *B* must trust the CA that signed *A*'s certificate.)

Once *B* has checked out *A*'s certificate, *B* must make sure that *A* really is the person identified in the certificate.

B generates a random message and sends it to *A*, asking *A* to encrypt it.

A encrypts the message using his private key, and sends it back to *B*.

B decrypts the message using *A*'s public key.

If this results in the original random message, then *B* knows that *A* is who he says he is.

Now that *B* trusts *A*'s identity, the same operation must happen in reverse.

B sends *A* her certificate, *A* validates the certificate and sends a challenge message to be encrypted.

B encrypts the message and sends it back to *A*, and *A* decrypts it and compares it with the original.

If it matches, then *A* knows that *B* is who she says she is.

At this point, *A* and *B* have established a connection to each other and are certain that they know each others' identities.

2.5. Confidential Communication

By default, GSI does not establish confidential (encrypted) communication between parties. Once mutual authentication is performed, GSI gets out of the way so that communication can occur without the overhead of constant encryption and decryption.

GSI can easily be used to establish a shared key for encryption if confidential communication is desired. Recently relaxed United States export laws now allow us to include encrypted communication as a standard optional feature of GSI.

A related security feature is communication integrity. Integrity means that an eavesdropper may be able to read communication between two parties but is not able to modify the communication in any way. GSI provides communication integrity by default. (It can be turned off if desired). Communication integrity introduces some overhead in communication, but not as large an overhead as encryption.

2.6. Securing Private Keys

The core GSI software provided by the Globus Toolkit expects the user's private key to be stored in a file in the local computer's storage. To prevent other users of the computer from stealing the private key, the file that contains the key is encrypted via a password (also known as a passphrase). To use GSI, the user must enter the passphrase required to decrypt the file containing their private key.

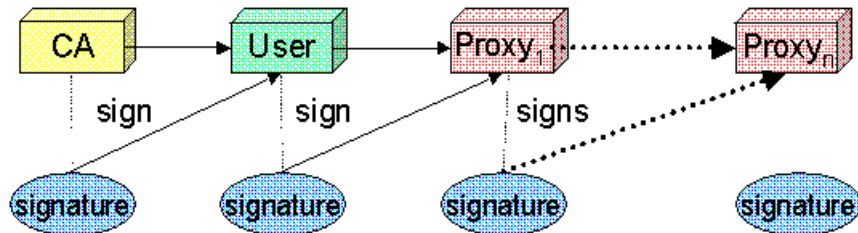
We have also prototyped the use of cryptographic smartcards in conjunction with GSI. This allows users to store their private key on a smartcard rather than in a file system, making it still more difficult for others to gain access to the key.

2.7. Delegation, Single Sign-On and Proxy Certificates

GSI provides a delegation capability: an extension of the standard SSL protocol which reduces the number of times the user must enter his passphrase. If a Grid computation requires that several Grid resources be used (each requiring

mutual authentication), or if there is a need to have agents (local or remote) requesting services on behalf of a user, the need to re-enter the user's passphrase can be avoided by creating a *proxy*.

A proxy consists of a new certificate and a private key. The key pair that is used for the proxy, i.e. the public key embedded in the certificate and the private key, may either be regenerated for each proxy or obtained by other means. The new certificate contains the owner's identity, modified slightly to indicate that it is a proxy. The new certificate is signed by the owner, rather than a CA. (See diagram below.) The certificate also includes a time notation after which the proxy should no longer be accepted by others. Proxies have limited lifetimes.



The proxy's private key must be kept secure, but because the proxy isn't valid for very long, it doesn't have to be kept quite as secure as the owner's private key. It is thus possible to store the proxy's private key in a local storage system without being encrypted, as long as the permissions on the file prevent anyone else from looking at them easily. Once a proxy is created and stored, the user can use the proxy certificate⁶ and private key for mutual authentication without entering a password.

When proxies are used, the mutual authentication process differs slightly. The remote party receives not only the proxy's certificate (signed by the owner), but also the owner's certificate. During mutual authentication, the owner's public key (obtained from her certificate) is used to validate the signature on the proxy certificate. The CA's public key is then used to validate the signature on the owner's certificate. This establishes a chain of trust from the CA to the proxy through the owner.

Note

GSI, and software based on it (notably the Globus Toolkit, GSI-SSH, and GridFTP), is currently the only software which supports the delegation extensions to TLS (a.k.a. SSL). The Globus Alliance has worked in the GGF and the IETF to standardize this extension in the form of Proxy Certificates (RFC 3820) [<http://www.ietf.org/rfc/rfc3820.txt>].

3. Related Documents

- [Globus Toolkit Version 4 Grid Security Infrastructure: A Standards Perspective](#)⁷
- [Handbook of Applied Cryptography](#)⁸

GT 4.0 Security Glossary

C

Certificate Authority (CA) An entity that issues certificates.

⁶ #proxy-cert

⁷ GT4-GSI-Overview.pdf

⁸ <http://www.cacr.math.uwaterloo.ca/hac/>

CA Certificate	The CA's certificate. This certificate is used to verify signature on certificates issued by the CA. GSI typically stores a given CA certificate in <code>/etc/grid-security/certificates/<hash>.0</code> , where <code><hash></code> is the hash code of the CA identity.
CA Signing Policy	The CA signing policy is used to place constraints on the information you trust a given CA to bind to public keys. Specifically it constrains the identities a CA is trusted to assert in a certificate. In GSI the signing policy for a given CA can typically be found in <code>/etc/grid-security/certificates/<hash>.signing_policy</code> , where <code><hash></code> is the hash code of the CA identity. For more information see [add link].
certificate	A public key and 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.
Certificate Revocation List (CRL)	A list of revoked certificates generated by the CA that originally issued them. When using GSI this list is typically found in <code>/etc/grid-security/certificates/<hash>.r0</code> , where <code><hash></code> is the hash code of the CA identity.
certificate subject	A identifier for the certificate owner, e.g. <code>"/DC=org/DC=doegrids/OU=People/CN=John Doe 123456"</code> . The subject is part of the information the CA binds to a public key when creating a certificate.
credentials	The combination of a certificate and the matching private key.

E

End Entity Certificate (EEC)	A certificate belonging to a non-CA entity, e.g. you, me or the computer on your desk.
------------------------------	--

G

GAA Configuration File	A file that configures the Generic Authorization and Access control GAA libraries. When using GSI this file is typically found in <code>/etc/grid-security/gsi-gaa.conf</code> .
grid map file	A file containing entries mapping certificate subjects to local user names. This file can also serve as a access control list for GSI enabled services and is typically found in <code>/etc/grid-security/grid-mapfile</code> . For more information see the Gridmap file ⁹ .
grid security directory	The directory containing GSI configuration files such as the GSI authorization callout configuration and GAA configuration files. Typically this directory is <code>/etc/grid-security</code> . For more information see Grid security directory ¹⁰ .
GSI authorization callout configuration file	A file that configures authorization callouts to be used for mapping and authorization in GSI enabled services. When using GSI this file is typically found in <code>/etc/grid-security/gsi-authz.conf</code> .

⁹ http://www.globus.org/toolkit/docs/4.0/security/prewsaa/Pre_WS_AA_Public_Interfaces.html#prewsaa-env-gridmapfile

¹⁰ http://www.globus.org/toolkit/docs/4.0/security/prewsaa/Pre_WS_AA_Public_Interfaces.html#prewsaa-env-gridsecurity

H

host certificate An EEC belonging to a host. When using GSI this certificate is typically stored in `/etc/grid-security/hostcert.pem`. For more information on possible host certificate locations see the [Credentials](#)¹¹.

host credentials The combination of a host certificate and its corresponding private key..

P

private key The private part of a key pair. Depending on the type of certificate the key corresponds to it may typically be found in `$HOME/.globus/userkey.pem` (for user certificates), `/etc/grid-security/hostkey.pem` (for host certificates) or `/etc/grid-security/<service>/<service>key.pem` (for service certificates). For more information on possible private key locations see the [Credentials](#)¹²

proxy certificate A short lived certificate issued using a EEC. A proxy certificate typically has the same effective subject as the EEC that issued it and can thus be used in its stead. GSI uses proxy certificates for single sign on and delegation of rights to other entities.

For more information about types of proxy certificates and their compatibility in different versions of GT, see <http://dev.globus.org/wiki/Security/ProxyCertTypes>.

proxy credentials The combination of a proxy certificate and its corresponding private key. GSI typically stores proxy credentials in `/tmp/x509up_u<uid>`, where `<uid>` is the user id of the proxy owner.

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

S

service certificate A EEC for a specific service (e.g. FTP or LDAP). When using GSI this certificate is typically stored in `/etc/grid-security/<service>/<service>cert.pem`. For more information on possible service certificate locations see the [Credentials](#)¹³.

service credentials The combination of a service certificate and its corresponding private key.

T

transport-level security Uses transport-level security (TLS) mechanisms.

¹¹ http://www.globus.org/toolkit/docs/4.0/security/prewsaa/Pre_WS_AA_Public_Interfaces.html#prewsaa-env-credentials

¹² http://www.globus.org/toolkit/docs/4.0/security/prewsaa/Pre_WS_AA_Public_Interfaces.html#prewsaa-env-credentials

¹³ http://www.globus.org/toolkit/docs/4.0/security/prewsaa/Pre_WS_AA_Public_Interfaces.html#prewsaa-env-credentials

trusted CAs directory The directory containing the CA certificates and signing policy files of the CAs trusted by GSI. Typically this directory is `/etc/grid-security/certificates`. For more information see [Grid security directory](#)¹⁴.

U

user certificate A EEC belonging to a user. When using GSI this certificate is typically stored in `$HOME/.globus/usercert.pem`. For more information on possible user certificate locations see [Credentials](#)¹⁵.

user credentials The combination of a user certificate and its corresponding private key.

¹⁴ http://www.globus.org/toolkit/docs/4.0/security/prewsaa/Pre_WS_AA_Public_Interfaces.html#prewsaa-env-gridsecurity

¹⁵ http://www.globus.org/toolkit/docs/4.0/security/prewsaa/Pre_WS_AA_Public_Interfaces.html#prewsaa-env-credentials

Chapter 2. GT4 Delegation Service Admin Guide

1. Introduction

This guide contains advanced configuration information for system administrators working with the Delegation Service. It provides references to information on procedures typically performed by system administrators, including installation, configuring, deploying, and testing the installation.

Important

This information is in addition to the basic Globus Toolkit prerequisite, overview, installation, security configuration instructions in the [GT 4.0 System Administrator's Guide](#)¹. Read through this guide before continuing!

2. Building and Installing

Refer to [System Administrator's guide](#)² for installation instructions.

3. Configuring

3.1. Configuration overview

The security settings for Delegation Factory Service and Delegation Service can be configured by modifying the security descriptors. The descriptors allow for configuring the credentials that will be used by the services and the type of authentication and message protection required, as well as the authorization mechanism.

By default, the following configuration is installed:

- Delegation Factory Service:
 - Credentials are determined by the container level security descriptor. If there is no container level security descriptor or if it does not specify what credentials to use then default credentials are used.
 - Authentication and message integrity protection is enforced for the `requestSecurityToken` operation. Other operations do not require authentication. This means that you may use any of GSI *Transport*, GSI Secure Message or GSI Secure Conversation when invoking the `requestSecurityToken` operation on the delegation factory service.
 - Access is authorized using the grid map mechanism and no grid map is configured in the service by default. If a grid map is configured in the container level security descriptor, it is used. To configure a *grid map file* for this service refer to instructions in the next section.
- Delegation Service
 - Credentials are determined by the container level security descriptor. If there is no container level security descriptor or if it does not specify what credentials to use then default credentials are used.

¹ ../../admin/docbook/

² ../../admin/docbook/

- Authentication and message integrity protection is enforced for all operations. This means that you may use any of GSI Transport, GSI Secure Message or GSI Secure Conversation when interacting with the delegation service.
- Access to resources managed by the Delegation Service is managed using the gridmap mechanism. The gridmap used is resource specific and is populated with the subject of the client that originally created the resource. This implies that only the user who delegated can access (and refresh) the delegated credential.



Note

Changing required authentication and authorization methods will require corresponding changes to the clients that contact this service.



Important

If the service is configured to use GSI Secure Transport, then container credentials are used for the handshake, irrespective of whether service level credentials are specified.

3.2. Syntax of the interface

To alter the security descriptor configuration refer to [Security Descriptors](#)³.

To alter the security configuration of the Delegation Factory Service, edit the file `$GLOBUS_LOCATION/etc/globus_delegation_service/factory-security-config.xml`.



Note

To either specify a gridmap file different from the container level configuration or to add one if the container security descriptor does not specify one, the following needs to be added to the Delegation Factory security descriptor.

```
<securityConfig xmlns="http://www.globus.org">
    .
    .
    .
    <gridmap value="path/to/gridmap/file"/>
</securityConfig>
```

To alter the security configuration of the Delegation Service, edit the file `$GLOBUS_LOCATION/etc/globus_delegation_service/service-security-config.xml`

³ http://www.globus.org/toolkit/docs/4.0/security/authzframe/security_descriptor.html

4. Deploying

The Delegation Service is deployed as part of a standard toolkit installation. Please refer to the [System Administrator's Guide](#)⁴ for details.

4.1. Deploying into Tomcat

Delegation Service has been tested to work without any additional setup when deployed into Tomcat. Please follow these [basic instructions](#)⁵ to deploy GT4 services into Tomcat. Note that the Java WS Core module needs to be built and configured as described in the previous sections.

5. Testing

- Install the Delegation Service test package (*gt4-cas-delegation-test-4.0.1-src_bundle.tar.gz*) using GPT build.
- To run the tests do:

```
$ cd $GLOBUS_LOCATION
$ ant -f share/globus_wsrf_test/runtests.xml runServer \
    -Dtests.jar=$GLOBUS_LOCATION/lib/globus_delegation_test.jar \
    -Djunit.jvmarg="-Dorg.globus.wsrf.container.server.id=delegationTest"
```

- The test report can be found in `$GLOBUS_LOCATION/share/globus_wsrf_test/tests/test-reports/TEST-org.globus.delegation.service.PackageTests.xml`.

6. Security Considerations

6.1. Key Pair Reuse

The current design re-uses the keys associated with the delegation service for each of the *proxy certificates* delegated to it. During a security review it was pointed out that while this was fine from a cryptographic standpoint, compromising this single long lived key pair may significantly extend the time for which a single intrusion (presuming an exploitable security flaw making the intrusion possible) is effective.

This can be remedied by either frequently regenerating the key pair used by the delegation service, which can be accomplished with a simple cron job, or by generating a new key pair for each new delegation. The later of these approaches requires changes to the design and may be adopted in future versions of the toolkit. For the time being we recommend the former approach should this issue concern you.

6.2. Authorizing Server prior to delegation

The delegation client that is distributed with the toolkit allows for delegation of credentials even when no authorization of the server is done. Also, when using secure message authentication the authorization of the server is done after the completion of the operation. These two scenarios could lead to the delegation of credentials to a malicious server.

To prevent this users should use secure *transport* (HTTPS) or GSI secure conversation and appropriate client side authorization.

⁴ [../admin/docbook/](#)

⁵ <http://www.globus.org/toolkit/docs/4.0/common/javawscore/admin-index.html#javawscore-admin-tomcat-deploying>

7. Troubleshooting

Refer to the [Globus Toolkit Administrator Guide - Security Overview](#)⁶ troubleshooting section for details on some common security installation issues.

Also, for security related troubleshooting the [CoG FAQ](#)⁷ might prove useful (especially sections on configuring credentials, CAs and so on).

⁶ <http://www.globus.org/toolkit/docs/4.0/admin/docbook/ch06.html#s-security-troubleshooting>

⁷ <http://www.globus.org/cog/distribution/1.2/FAQ.TXT>

Chapter 3. GT4 Delegation Service User's Guide

1. Introduction

The delegation service can be used when a user wants to delegate rights to a service that is hosted in the same container as the delegation service. The delegation service accepts a credential from the user and provides access to that credential to any authorized service that runs in the same container. Upon delegation to the service an endpoint reference to the delegated credential is returned to the client, which can then be furnished to other services as a handle to the credential.

Moreover, the endpoint reference returned on delegation can be used by the client to refresh the credential stored with the delegation service. When the client performs a refresh the service sends notifications to any service that has registered interest in that particular credential.

The generic client `wsrfr-destroy`¹ can be used to remove the delegated credential.



Note

If the service being contacted is using *GSI Secure Transport*, then the container credentials configured for the service will be used, even if service/resource level credentials are configured. Hence authorization needs to be done based on the DN of the container credentials.

2. Command-line tools

Please see the [Delegation Service Command Reference](#).

3. Graphical user interfaces

There is no GUI for the Delegation Service.

4. Troubleshooting

4.1. AuthorizationException: "test DN" is not authorized to use operation: {http://www.globus.org/08/2004/delegationService}requestSecurityToken

This exception can occur when a client whose DN is not in the *grid map file* configured for the delegation factory service attempts to delegate (using `globus-credential-delegate`) a credential to the factory service.



Note

The *test DN* specified in the error message is just a placeholder and will contain the DN of the user attempting to access the credential.

¹ <http://www.globus.org/toolkit/docs/4.0/common/javawscore/mn01re03.html>

4.2. AuthorizationException: "test DN" is not authorized to use operation: {http://www.globus.org/08/2004/delegationService}refresh

This exception can occur when a client attempts to refresh a credential it did not delegate (using `globus-credential-refresh`).



Note

The *test DN* specified in the error message is just a placeholder and will contain the DN of the user attempting to access the credential.

4.3. CoG Configuration and troubleshooting

Also, for security related troubleshooting the [CoG FAQ](http://www.globus.org/cog/distribution/1.2/FAQ.TXT)² might prove useful (especially sections on configuring credentials, CAs and so on.)

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

Chapter 4. GT4 Delegation Service Developer's Guide

1. Introduction

The Delegation Service provides an interface that, given the endpoint reference to the credential resource, allows service developers to retrieve a delegated credential. While there is a remote interface to delegate and refresh a credential, no remote interface is provided for acquiring the delegated credential (that is, all access is through a shared Java state.)

In addition, the component provides a utility API that can be used for developing client side code to generate a credential, delegate it and refresh it.

2. Before you begin

2.1. Feature summary

Features new in GT 4.0

- Provides an interface for the delegation and renewal of credentials to a host.
- Allows for a single delegated credential to be reused across multiple service invocations (e.g. GRAM jobs).

Other Supported Features

- The Delegation Service is a new component in GT 4.0.

Deprecated Features

- The Delegation Service is a new component in GT 4.0.

2.2. Tested platforms

Tested Platforms for Delegation Service

- Windows XP
- Linux (Red Hat 7.3)

Tested Containers for Delegation Service

- Java WS Core container
- Tomcat 5.0.30

2.3. Backward compatibility summary

The Delegation Service is a new component in GT 4.0

2.4. Technology dependencies

The Delegation Service depends on the following GT components:

- WS Authentication and Authorization
- Java WS Core

The Delegation Service depends on the following 3rd party software:

- Apache Axis

2.5. Security considerations

2.5.1. Key Pair Reuse

The current design re-uses the keys associated with the delegation service for each of the *proxy certificates* delegated to it. During a security review it was pointed out that while this was fine from a cryptographic standpoint, compromising this single long lived key pair may significantly extend the time for which a single intrusion (presuming an exploitable security flaw making the intrusion possible) is effective.

This can be remedied by either frequently regenerating the key pair used by the delegation service, which can be accomplished with a simple cron job, or by generating a new key pair for each new delegation. The later of these approaches requires changes to the design and may be adopted in future versions of the toolkit. For the time being we recommend the former approach should this issue concern you.

2.5.2. Authorizing Server prior to delegation

The delegation client that is distributed with the toolkit allows for delegation of credentials even when no authorization of the server is done. Also, when using secure message authentication the authorization of the server is done after the completion of the operation. These two scenarios could lead to the delegation of credentials to a malicious server.

To prevent this users should use secure *transport* (HTTPS) or GSI secure conversation and appropriate client side authorization.

3. Architecture and design overview

3.1. Overview

This component offers an interface for delegating credentials and subsequently managing them. It exposes the delegated credentials as a WS-Resources and allows authorized co-hosted services to access these credentials through a Java API. Furthermore, it gives clients the ability to refresh and manage the lifetime of their delegated credentials.

This component has a Delegation Factory Service and Delegation Service. A delegate call on the factory creates a WS-Resource managed by the Delegation Service that represents the delegated credential. The delegate call returns a Endpoint Reference (EPR) that can be used to later refresh the credentials.

Services that are interested in the delegated credential can register a listener (an object that implements `org.globus.delegation.DelegationRefreshListener`) with the specific delegated credential resource. There currently is no remote interface for this, hence only services that are in the same hosting environment can register interest. The credentials are pushed to the listener anytime a refresh is done.

In practice delegation is done as follows. The Delegation Factory Service publishes its certificate chain, including the service's certificate, as a Resource Property. In the first step of the delegation process the client obtains this certificate chain, validates and authorizes it, and extracts the *public key* from the Delegation Factory *Service certificate*. The client then creates the *proxy certificate* it is going to delegate by binding, i.e. signing, the service's public key to the proxy certificate information using its *private key*. In the third and final step the client passes the certificate chain that starts with the proxy certificate to the Delegation Factory Service, which upon receipt replies with the address to the created delegated credential WS-Resource.

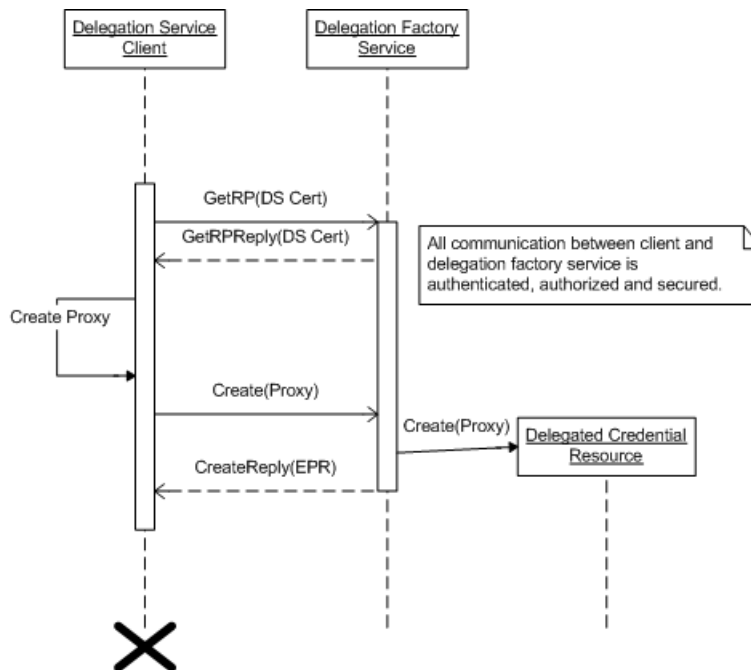
Additionally, the component allows the delegator to renew delegated credentials. Credential renewal follows the same steps as the initial delegation except for the final step, in which the client acts against its WS-Resource instead of the Delegation Service Factory.

3.2. Relationship to WS-Trust

The Delegation service uses WS-Trust messages as described in the WS-Trust specification. However, it should be noted that these messages are underspecified (the contents of the basic WS-Trust envelop are xsd:any) and the contents of these messages for the Delegation Services are simplistic and do not achieve the "spirit" intended by the specification.

3.3. Normal Usage Patterns

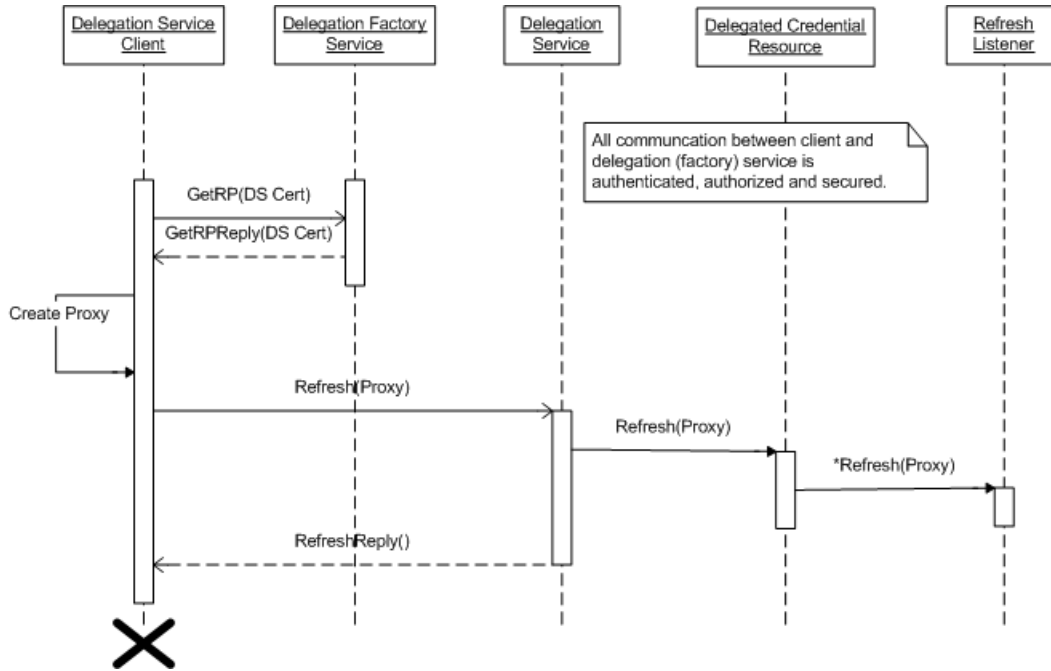
3.3.1. Creation



1. The client does a lookup on the Delegation Factory Service (DFS) for the Resource Property (RP) containing the certificate chain of the DFS. To perform this lookup the client uses the EPR of the DFS obtained from an RP of another application (e.g. GRAM, RFT) or some other OOB method.
 - This lookup is secured via *transport-level security*. The client and server mutually authenticate and authorize each other. Messages are integrity protected.
2. The client creates a proxy certificate by binding the public key of the DFS, obtained from the certificate chain in the previous step, to the proxy identity of the client.

- The lifetime of a proxy certificate defaults to the lifetime of the signing certificate (typically a short-lived proxy).
3. The client sends the proxy certificate (and chain) to the DFS. An endpoint reference (EPR) for the delegated credential is returned to client, which the client may use in subsequent invocations of other services co-hosted with the Delegation Service.
 - The resource expires on proxy expiration.
 - An ACL is created for the delegated credential resource, which contains the identity of the credential delegator.

3.3.2. Refresh



The original delegator of a credential may refresh the credential (i.e. replace it with a different credential, presumably one that has an expiration date farther in the future). The usage pattern for this method is identical to the Creation method described in the previous section, with the exception that the EPR of the previously created Delegation Resource is used as opposed to the EPR of the Delegation Factory Service.

3.4. Credential Storage

The Delegation Factory Service and the Delegation Service use standard Grid service credentials, namely a private key and certificate chain where the *EEC* has a DN containing the FQDN of the host on which the services are running. The private key and certificate chain are stored on local disk, protected by local file system permissions. By default the PEM files `/etc/grid-security/containercert.pem` and `/etc/grid-security/container-key.pem` are used.

When a delegated *proxy credential* is persisted to disk, it is stored as a serialized JAVA object in `~globus/.globus/persisted/{ip addr}/DelegationResource/`. The private key of the Delegation Service is stored with the delegated proxy certificate to ease the use of delegated credentials by services in the hosting environment.

3.5. Access to Delegated Proxy Resources

After delegating a credential to the Delegation Service a client will typically invoke an application service (e.g. GRAM, RFT) that requires the use of such a delegated credential. In such situations the client will pass the EPR of the delegated credential to the service which it is invoking.

The service will use an internal hosting environment API (as opposed to a web services method) to access the delegated credential. This interface provides the identity of the requesting client to the underlying software, which verifies that the client identity is present in the ACL of the proxy resource before returning the requested credential.

3.6. Registration for Renewal Events

Services internal to the hosting environment can register with a resource proxy to receive updated credentials when they are renewed by the client. Such registrations are authorized in the same manner as direct access to the proxy. Registered services will have any renewed credentials pushed to them.

4. Public interface

The semantics and syntax of the APIs and WSDL for the component, along with descriptions of domain-specific structured interface data, can be found in the [Public Interface Guide](#)¹.

5. Usage scenarios

5.1. Client-side scenario

- Prior to delegating, the client needs to get information about the public key of the Delegation Factory Service. A utility API to do that is described [here](#)².
- Once the delegation client has the public key of the Delegation Factory Service, it needs to create a delegated credential using that public key and then invoke the remote interface on the factory to delegate the credential. A utility API to do all of the above is described [here](#)³. The Endpoint Reference that is returned by this operation can be distributed to services that the user would like to delegate its rights to.
- The user may need to refresh the delegated credential. The onus is on the user to do this prior to expiration of the delegated credential. If the user does not refresh the credential, the expired credential will be garbage collected and the Endpoint Reference cannot be reused. A utility API that can be used to refresh is described [here](#)⁴.

5.2. Service-side scenario

This section describes the usage scenario where a service is provided with a delegated credential EPR and needs to access the credential. Typically, as a part of an application the delegated credential EPR is sent to the service and and it is assumed that the delegation service is co-hosted (that is, it runs in the same hosting environment).

The service needs to create a listener object that implements the `org.globus.delegation.DelegationRefreshListener` interface and register the listener with the Delegation Service. Upon registering the listener the Delegation Service checks that the delegator identity matches either the identity passed in the subject object or the

¹ [WS_AA_Delegation_Service_Public_Interfaces.html](#)

² [WS_AA_Delegation_Service_Public_Interfaces.html#getCertChain](#)

³ [WS_AA_Delegation_Service_Public_Interfaces.html#delegate](#)

⁴ [WS_AA_Delegation_Service_Public_Interfaces.html#refresh](#)

identity contained in the peer subject associated with the current message context. Once the listener has been authorized the delegated credential is made available to the listener. Details about the API are described [here](#)⁵.

6. Tutorials

There are no tutorials available at this time.

7. Debugging

Log output from the delegation service is a useful tool for debugging issues. Logging in the delegation service code is based on the [Jakarta Commons Logging](#)⁶ API and is described in more detail [here](#)⁷. As described in the above section, configuration files need to be edited to enable logging at different levels. For example, to see all debug logging statements for the delegation service, the following lines need to be added:

```
log4j.category.org.globus.delegation.service=DEBUG
```

```
log4j.category.org.globus.delegation.factory=DEBUG
```

Debugging information from delegation clients can be obtained by setting the following line in the client side logging configuration file:

```
log4j.category.org.globus.delegation.client=DEBUG
```



Note

Client side logging configuration has to be done in `$GLOBUS_LOCATION/log4j.properties`.

8. Troubleshooting

8.1. Unable to retrieve caller DN, cannot register

This error occurs when attempting to register a listener with a delegated credential resource without a JAAS subject. There are two ways of registering: either the JAAS subject can be explicitly passed using the API or the JAAS subject can be picked up from the current message context (the subject representing the client). If the later mechanism for registering is used and there is no client credential associated with the thread that is calling the register function, then this exception is thrown. If this occurs, make sure to use the API call that explicitly passes the subject.

8.2. *test user DN* is not authorized to access this resource

Only the user who created the delegated credential is allowed to access it. There are two sets of API functions for getting the credential and registering listeners: one in which the caller's DN is picked up from the current thread and the other in which a JAAS subject (containing the caller's DN) is explicitly passed as a function parameter. If the caller's DN (picked up from thread or specified explicitly) does not match the DN of the user who created the credential, this error is thrown. Ensure that the DN explicitly specified or the client DN associated with the thread matches the creator's DN.

⁵ [WS_AA_Delegation_Service_Public_Interfaces.html#registerListener](#)

⁶ <http://jakarta.apache.org/commons/logging/>

⁷ [../common/javawscore/developer-index.html#s-javawscore-developer-debugging](#)



Note

The *test user DN* specified in the error message is just a placeholder and will contain the actual DN of the user attempting to access the credential.

8.3. CoG Configuration and troubleshooting

Also, for security related troubleshooting the [CoG FAQ](#)⁸ might prove useful (especially sections on configuring credentials, CAs and so on).

9. Related Documentation

- [WS-Security](#)⁹
- [WS-Security: X.509 Certificate Tokens](#)¹⁰
- [WS-Trust](#)¹¹
- [RFC 3820](#)¹² Proxy Certificates

⁸ <http://www.globus.org/cog/distribution/1.2/FAQ.TXT>

⁹ <http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf>

¹⁰ <http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0.pdf>

¹¹ <http://www-106.ibm.com/developerworks/library/specification/ws-trust/>

¹² <http://www.faqs.org/rfcs/rfc3820.html>

Chapter 5. GT4 Delegation Service Factsheet

1. Brief component overview

The Delegation Service is a new component in Globus Toolkit 4.0. This component provides an interface for delegation of credentials to a hosting environment. This enables a single delegated credential to be shared across multiple invocations of services on that hosting environment (e.g. it could be used for multiple GRAM job submissions or across GRAM and RFT submissions.) It also provides a means for credential renewal.

2. Summary of features

Features new in GT 4.0

- Provides an interface for the delegation and renewal of credentials to a host.
- Allows for a single delegated credential to be reused across multiple service invocations (e.g. GRAM jobs).

Other Supported Features

- The Delegation Service is a new component in GT 4.0.

Deprecated Features

- The Delegation Service is a new component in GT 4.0.

3. Usability summary

Usability improvements for the Delegation Service:

- Better validation of values passed via command line arguments.
- Use of the client side security descriptor in the utility API for accessing delegated credentials by services in same container.

4. Backward compatibility summary

The Delegation Service is a new component in GT 4.0

5. Technology dependencies

The Delegation Service depends on the following GT components:

- WS Authentication and Authorization
- Java WS Core

The Delegation Service depends on the following 3rd party software:

- Apache Axis

6. Tested platforms

Tested Platforms for Delegation Service

- Windows XP
- Linux (Red Hat 7.3)

Tested Containers for Delegation Service

- Java WS Core container
- Tomcat 5.0.30

7. Associated standards

- [WS-Security](#)¹
- [WS-Security: X.509 Certificate Tokens](#)²
- [WS-Trust](#)³
- [RFC 3820](#)⁴ Proxy Certificates

8. For More Information

Click [here](#)⁵ for more information about this component.

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-x509-token-profile-1.0.pdf>

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

⁴ <http://www.faqs.org/rfcs/rfc3820.html>

⁵ [index.html](#)

Chapter 6. GT4 Delegation Service Public Interfaces

1. Semantics and syntax of APIs

1.1. Programming Model Overview

This component consists of two services: the delegation factory service and the delegation service.

The delegation factory service exposes its public certificate as a resource property and allows clients to delegate credentials bound to that *public key*. Upon delegation an Endpoint Reference(EPR) to the delegated credential, which is implemented as a resource of the delegation service, is returned to the client. The client can use this EPR to provide a reference to the delegated credential to other services.

The delegation service itself has an interface to allow refreshing the credentials remotely. Other co-hosted services can register interest in delegated credentials through listeners and be notified when credentials are refreshed.

1.2. Component API

Some relevant API:

- `org.globus.delegation.DelegationUtil`
- `org.globus.delegation.DelegationRefreshListener`
- `org.globus.delegation.delegationService.DelegationPortType`
- `org.globus.delegation.delegationService.DelegationFactoryPortType`

Complete API:

- [Service API](#)¹
- [Common API](#)²

2. Semantics and syntax of the WSDL

2.1. Protocol overview

The delegation service allows for delegation of credentials and is based on the [WS-Trust](#)³ specification. A WSDL interface to refresh the credentials remotely is also provided. Access to these credentials is restricted to co-hosted services, i.e services that are run in the same container, and is done using shared Java state. Co-hosted services interested in the credentials can register listeners and will be notified upon credential refresh.

¹ http://www.globus.org/api/javadoc-4.0.0/globus_wsrf_delegation_service_java/

² http://www.globus.org/api/javadoc-4.0.0/globus_wsrf_delegation_stubs_java/

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

2.2. Operations

2.2.1. Delegation Factory Service

- `RequestSecurityToken`: This operation allows for a security token to be sent to the service.

2.2.2. Delegation Service

- `refresh`: This operation is used to refresh a delegated credential. When invoked, all services that have registered interest in the credential through listeners are notified.

2.3. Resource properties

2.3.1. Delegation Factory Service

- `CertificateChain`: This resource property is used to expose the certificate used by delegation service.

2.4. Faults

All operations on delegation service and delegation factory service throw `RemoteException` in case of failure.

2.5. WSDL and Schema Definition

- [Delegation Factory Service WSDL](#)⁴
- [Delegation Service WSDL](#)⁵

3. Command-line tools

Please see the [Delegation Service Command Reference](#).

4. Overview of Graphical User Interface

There is no GUI for the Delegation Service.

5. Semantics and syntax of domain-specific interface

5.1. Delegation Service API

The `org.globus.delegation.DelegationUtil` provides an API that allows users to get the certificate chain resource property exposed by the Delegation Factory Service, delegate to a service, and to refresh and register listeners.

⁴ http://viewcvs.globus.org/viewcvs.cgi/ws-delegation/common/schema/delegationService/delegation_factory_flattened.wsdl?rev=1.3&only_with_tag=globus_4_0_0&content-type=text/vnd.viewcvs-markup

⁵ http://viewcvs.globus.org/viewcvs.cgi/ws-delegation/common/schema/delegationService/delegation_flattened.wsdl?rev=1.2&only_with_tag=globus_4_0_0&content-type=text/vnd.viewcvs-markup

5.1.1. To get certificate chain of delegation factory service

```
static X509Certificate[] getCertificateChainRP(String delegationUrl)
```

This takes an endpoint URL to a Delegation Factory Service and queries the CertificateChain resource property. The chain of certificates is returned as an array of X509Certificate. The client needs to delegate on the first certificate in the returned chain.

5.1.2. To delegate

```
public static EndpointReferenceType delegate(String delegationServiceUrl,
                                           GlobusCredential issuingCred,
                                           X509Certificate certificate,
                                           int lifetime,
                                           boolean fullDelegation,
                                           ClientSecurityDescriptor desc)
```

This utility method is used to create the security token to delegate using the *issuingCred* and *certificate* parameters. The lifetime and type of the delegated credential created is determined by the *lifetime* and *fullDelegation* parameters. The security token (delegated credential) thus created is then stored by the Delegation Factory Service specified by the *delegationServiceUrl*. The client security descriptor determines the authentication mechanism, protection and authorization settings to use.

The Endpoint Reference that is returned points to the delegated credential and can be used by co-hosted services (services in the same hosting environment) to retrieve the delegated credential.

5.1.3. To refresh a delegated credential

```
public static void refresh(GlobusCredential issuingCred,
                          X509Certificate certToSign,
                          int lifetime,
                          boolean fullDelegation,
                          ClientSecurityDescriptor desc,
                          EndpointReferenceType epr)
```

This method can be used to refresh a delegated credential that is referred to by the EPR *epr*. A new delegated credential is created using the *issuingCred*, *certToSign*, *lifetime* and *fullDelegation* parameters. The client security descriptor determines the authentication mechanism, protection and authorization type to use.

5.1.4. To register listener

```
static void
    registerDelegationListener(EndpointReferenceType epr,
                              DelegationRefreshListener listener,
                              Subject subject)
```

This method registers the listener *listener* with the delegation resource referenced by *epr*. The operation is permitted only if the identity in the subject object matches that of the user who delegated the credential.

```
static void
    registerDelegationListener(EndpointReferenceType epr,
                              DelegationRefreshListener listener)
```

This method provides the same functionality as the previous one, except that the subject object is picked up from the property `org.globus.wsrp.security.Constants.PEER_SUBJECT` in the current message context. If the identity of the user who delegated the credential matches that of the subject object referred to by the property, then the operation is permitted.

6. Configuration interface

6.1. Configuration overview

The security settings for Delegation Factory Service and Delegation Service can be configured by modifying the security descriptors. The descriptors allow for configuring the credentials that will be used by the services and the type of authentication and message protection required, as well as the authorization mechanism.

By default, the following configuration is installed:

- Delegation Factory Service:
 - Credentials are determined by the container level security descriptor. If there is no container level security descriptor or if it does not specify what credentials to use then default credentials are used.
 - Authentication and message integrity protection is enforced for the `requestSecurityToken` operation. Other operations do not require authentication. This means that you may use any of GSI *Transport*, GSI Secure Message or GSI Secure Conversation when invoking the `requestSecurityToken` operation on the delegation factory service.
 - Access is authorized using the grid map mechanism and no grid map is configured in the service by default. If a grid map is configured in the container level security descriptor, it is used. To configure a *grid map file* for this service refer to instructions in the next section.
- Delegation Service
 - Credentials are determined by the container level security descriptor. If there is no container level security descriptor or if it does not specify what credentials to use then default credentials are used.
 - Authentication and message integrity protection is enforced for all operations. This means that you may use any of GSI Transport, GSI Secure Message or GSI Secure Conversation when interacting with the delegation service.
 - Access to resources managed by the Delegation Service is managed using the gridmap mechanism. The gridmap used is resource specific and is populated with the subject of the client that originally created the resource. This implies that only the user who delegated can access (and refresh) the delegated credential.



Note

Changing required authentication and authorization methods will require corresponding changes to the clients that contact this service.



Important

If the service is configured to use GSI Secure Transport, then container credentials are used for the handshake, irrespective of whether service level credentials are specified.

6.2. Syntax of the interface

To alter the security descriptor configuration refer to [Security Descriptors](#)⁶.

To alter the security configuration of the Delegation Factory Service, edit the file `$GLOBUS_LOCATION/etc/globus_delegation_service/factory-security-config.xml`.



Note

To either specify a gridmap file different from the container level configuration or to add one if the container security descriptor does not specify one, the following needs to be added to the Delegation Factory security descriptor.

```
<securityConfig xmlns="http://www.globus.org">
    .
    .
    .
    <gridmap value="path/to/gridmap/file"/>
</securityConfig>
```

To alter the security configuration of the Delegation Service, edit the file `$GLOBUS_LOCATION/etc/globus_delegation_service/service-security-config.xml`

7. Environment variable interface

Refer to the [environment variable interface](#)⁷ for details.

The environment variables described above only affect the selection of credentials if no credentials are specified in any of the applicable security descriptors.

⁶ http://www.globus.org/toolkit/docs/4.0/security/authzframe/security_descriptor.html

⁷ http://www.globus.org/toolkit/docs/4.0/security/message/WS_AA_Message_Level_Public_Interfaces.html#s-message-public-env

Chapter 7. GT4 Delegation Service Quality Report

1. Test coverage reports

- [Clover test coverage reports](#)¹

2. Code analysis reports

- [PMD Report](#)²
- [FindBugs](#)³

3. Outstanding bugs

- [Bug 2973](#)⁴: Inconsistent arguments
- [Bug 3145](#)⁵
- [Bug 3077](#)⁶

4. Bug Fixes

- File permissions for persisted delegated credentials are now set before writing the credentials to disk.
- [Bug 2537](#)⁷: globus-credential-delegate should honor -help
- [Bug 2581](#)⁸: NPE in DelegationResource
- [Bug 2575](#)⁹: ConcurrentModificationException
- [Bug 2964](#)¹⁰: Command line option "-m" in delegate client
- [Bug 2966](#)¹¹: Command line option "-d" in delegate client
- [Bug 3076](#)¹²: ArrayOutOBounds in delegate client when no argument is passed

¹ <http://www.globus.org/Security/delegationService/>

² http://www.globus.org/Security/delegationService/HEAD/pmd/pmd_report.html

³ <http://www.globus.org/Security/delegationService/>

⁴ http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=2973

⁵ http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=3145

⁶ http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=3077

⁷ http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=2537

⁸ http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=2581

⁹ http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=2575

¹⁰ http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=2964

¹¹ http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=2966

¹² http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=3076

5. Performance reports

Only scalability tests have been done on the delegation service. It has been established that it can scale up to 10K credentials without failing. This number is not limiting (i.e it does not fail on delegating more than 10K), but is the maximum it has been tested with.

Chapter 8. GT 4.0 Migrating Guide for WS A &A Delegation Service

The following provides available information about migrating from previous versions of the Globus Toolkit.

1. Migrating from GT2

This is a new component in Globus 4.0.

2. Migrating from GT3

This is a new component in Globus 4.0.

Delegation Service Reference

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]

Table 1. globus-credential-delegate options

-help/--help	Displays usage information.
-h <host>	Host on which Delegation Factory Service is running. Defaults to localhost.
-p <port>	Port on which Delegation Factory Service is running. Defaults to 8443.
-c <credFile>	Filename to read credential from. If not specified, then the default proxy location is used.
-l <lifetime>	Lifetime for the created delegated credential, specified in seconds. Defaults to 12 hours.
-d <true/false>	Determines the type of delegation. If set to true full delegation is done, while with any other value limited delegation is done. If the option is not specified, it defaults to full delegation.
-m <security mechanism>	Sets the security mechanism type. If set to 'msg' Secure Message is used, if set to 'conv' Secure Conversation is used or if set to 'trans' Secure Transport is used. Defaults to Secure Transport.
-n <protection type>	Sets the protection type. If set to 'sig' signature is used, while if set to 'enc' encryption is used. Defaults to signature.
-a <authz>	Type of client authorization to use. If set to "none" then no authorization is done, if set to "host" then host authorization is done, if set to "self" then self authorization is done. If set to none of the above then the string specified is used as the expected identity. Defaults to host authorization.
<filename>	Filename to write the EPR of delegated credential to.

Name

globus-credential-refresh -- Delegation refresh client

globus-credential-refresh

Tool description

Used to refresh delegated credentials pointed to by the specified EPR. A new credential is generated and the one previously created by the delegation service is overwritten.

Command syntax

globus-credential-refresh [options]

Table 2. globus-credential-refresh options

-help/--help	Displays usage information.
-c <credFile>	Filename to read credential from. If not specified, then the default proxy location is used.
-l <lifetime>	Lifetime for the created delegated credential, specified in seconds. Defaults to 12 hours.
-d <true/false>	Determines type of delegation that needs to be done. If set to true full delegation is done, while if set to any other value limited delegation is done. If the option is not specified, it defaults to full delegation
-m <security mechanism>	Sets the security mechanism type. If set to 'msg' Secure Message is used, if set to 'conv' Secure Conversation is used or if set to 'trans' Secure Transport is used. Defaults to Secure Transport.
-n <protection type>	Sets the protection type. If set to 'sig' signature is used, while if set to 'enc' encryption is used. Defaults to signature.
-a <authz>	Type of client authorization to use. If set to "none" then no authorization is done, if set to "host" then host authorization is done, if set to "self" then self authorization is done. If set to none of the above then the string specified is used as the expected identity. Defaults to host authorization.
-e <filename>	Filename to read the EPR of the delegated credential resource from. Defaults to "delegatedCredEPR".

Chapter 9. GT 4.0.8 Incremental Release Notes: Delegation Service

1. Introduction

These release notes are for the incremental release 4.0.8. It includes a summary of changes since 4.0.7, bug fixes since 4.0.7 and any known problems that still exist at the time of the 4.0.8 release. This page is in addition to the top-level 4.0.8 release notes at <http://www.globus.org/toolkit/releasenotes/4.0.8>.

For release notes about 4.0 (including feature summary, technology dependencies, etc) go to the [Delegation Service 4.0 Release Notes](#)¹.

2. Changes Summary

No changes have been made to Delegation Service since GT 4.0.7.

3. Bug Fixes

No bugs have been fixed for Delegation Service since GT 4.0.7.

4. Known Problems

- Limitations: No new limitations have been identified.
- Known Bugs: No new bugs have been identified.

5. For More Information

Click [here](#)² for more information about this component.

¹ http://www.globus.org/toolkit/docs/4.0/security/delegation/WS_AA_Delegation_Service_Release_Notes.html

² [index.html](#)

Chapter 10. GT 4.0.7 Incremental Release Notes: Delegation Service

1. Introduction

These release notes are for the incremental release 4.0.7. It includes a summary of changes since 4.0.6, bug fixes since 4.0.6 and any known problems that still exist at the time of the 4.0.7 release. This page is in addition to the top-level 4.0.7 release notes at <http://www.globus.org/toolkit/releasenotes/4.0.7>.

For release notes about 4.0 (including feature summary, technology dependencies, etc) go to the [Delegation Service 4.0 Release Notes](#)¹.

2. Changes Summary

No changes have been made to Delegation Service since GT 4.0.6.

3. Bug Fixes

No bugs have been fixed for Delegation Service since GT 4.0.6.

4. Known Problems

- Limitations: No new limitations have been identified.
- Known Bugs
 - [Bug 4300](#):² Delegation Service does not implement GetRP interface

5. For More Information

Click [here](#)³ for more information about this component.

¹ http://www.globus.org/toolkit/docs/4.0/security/delegation/WS_AA_Delegation_Service_Release_Notes.html

² http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=4300

³ [index.html](#)

Chapter 11. GT 4.0.6 Incremental Release Notes: Delegation Service

1. Introduction

These release notes are for the incremental release 4.0.6. It includes a summary of changes since 4.0.5, bug fixes since 4.0.5 and any known problems that still exist at the time of the 4.0.6 release. This page is in addition to the top-level 4.0.6 release notes at <http://www.globus.org/toolkit/releasenotes/4.0.6>.

For release notes about 4.0 (including feature summary, technology dependencies, etc) go to the [Delegation Service 4.0 Release Notes](#)¹.

2. Changes Summary

No changes have been made to Delegation Service since GT 4.0.5.

3. Bug Fixes

No bugs have been fixed for Delegation Service since GT 4.0.5.

4. Known Problems

- Limitations: No new limitations have been identified.
- Known Bugs
 - [Bug 4300](#):² Delegation Service does not implement GetRP interface

5. For More Information

Click [here](#)³ for more information about this component.

¹ http://www.globus.org/toolkit/docs/4.0/security/delegation/WS_AA_Delegation_Service_Release_Notes.html

² http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=4300

³ [index.html](#)

Chapter 12. GT 4.0.5 Incremental Release Notes: Delegation Service

1. Introduction

These release notes are for the incremental release 4.0.5. It includes a summary of changes since 4.0.4, bug fixes since 4.0.4 and any known problems that still exist at the time of the 4.0.5 release. This page is in addition to the top-level 4.0.5 release notes at <http://www.globus.org/toolkit/releasenotes/4.0.5>.

For release notes about 4.0 (including feature summary, technology dependencies, etc) go to the [Delegation Service 4.0 Release Notes](#)¹.

2. Changes Summary

No changes have been made to Delegation Service since GT 4.0.4.

3. Bug Fixes

No bugs have been fixed for Delegation Service since GT 4.0.4.

4. Known Problems

No new problems are known to exist for the Delegation Service at the time of GT 4.0.5 release.

5. For More Information

Click [here](#)² for more information about this component.

¹ http://www.globus.org/toolkit/docs/4.0/security/delegation/WS_AA_Delegation_Service_Release_Notes.html

² [index.html](#)

Chapter 13. GT 4.0.4 Incremental Release Notes: Delegation Service

1. Introduction

These release notes are for the incremental release 4.0.4. It includes a summary of changes since 4.0.3, bug fixes since 4.0.3 and any known problems that still exist at the time of the 4.0.4 release. This page is in addition to the top-level 4.0.4 release notes at <http://www.globus.org/toolkit/releasenotes/4.0.4>.

For release notes about 4.0 (including feature summary, technology dependencies, etc) go to the [Delegation Service 4.0 Release Notes](#)¹.

2. Changes Summary

No changes have been made to Delegation Service since GT 4.0.3.

3. Bug Fixes

No bugs have been fixed for Delegation Service since GT 4.0.3.

4. Known Problems

No new problems are known to exist for the Delegation Service at the time of GT 4.0.4 release.

5. For More Information

Click [here](#)² for more information about this component.

¹ http://www.globus.org/toolkit/docs/4.0/security/delegation/WS_AA_Delegation_Service_Release_Notes.html

² [index.html](#)

Chapter 14. GT 4.0.3 Incremental Release Notes: Delegation Service

1. Introduction

These release notes are for the incremental release 4.0.3. It includes a summary of changes since 4.0.2, bug fixes since 4.0.2 and any known problems that still exist at the time of the 4.0.3 release. This page is in addition to the top-level 4.0.3 release notes at <http://www.globus.org/toolkit/releasenotes/4.0.3>.

For release notes about 4.0 (including feature summary, technology dependencies, etc) go to the [Delegation Service 4.0 Release Notes](#)¹.

2. Changes Summary

No changes have been made for the Delegation Service since 4.0.2.

3. Bug Fixes

No bugs have been fixed for the Delegation Service since GT 4.0.2.

4. Known Problems

No new problems are known to exist for the Delegation Service at the time of the GT 4.0.3 release.

5. For More Information

Click [here](#)² for more information about this component.

¹ http://www.globus.org/toolkit/docs/4.0/security/delegation/WS_AA_Delegation_Service_Release_Notes.html

² [index.html](#)

Chapter 15. GT 4.0.2 Incremental Release Notes: Delegation Service

1. Introduction

These release notes are for the incremental release 4.0.2. It includes a summary of changes since 4.0.1, bug fixes since 4.0.1 and any known problems that still exist at the time of the 4.0.2 release. This page is in addition to the top-level 4.0.2 release notes at <http://www.globus.org/toolkit/releasenotes/4.0.2>.

For release notes about 4.0 (including feature summary, technology dependencies, etc) go to the [Delegation Service 4.0 Release Notes](#)¹.

2. Changes Summary

Other than listed bug fixes, synchronization in delegation resource was improved.

3. Bug Fixes

The following bugs were fixed for Delegation Service:

- [Bug 3864](#):² Fixes to prevent deadlock when notification listeners are not removed upon resource destroy.

4. Known Problems

No new problems are known to exist for the Delegation Service.

5. For More Information

Click [here](#)³ for more information about this component.

¹ http://www.globus.org/toolkit/docs/4.0/security/delegation/WS_AA_Delegation_Service_Release_Notes.html

² http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=3864

³ [index.html](#)

Chapter 16. GT 4.0.1 Incremental Release Notes: Delegation Service

1. Introduction

These release notes are for the incremental release 4.0.1. It includes a summary of changes since 4.0.0, bug fixes since 4.0.0 and any known problems that still exist at the time of the 4.0.1 release. This page is in addition to the top-level 4.0.1 release notes at <http://www.globus.org/toolkit/releasenotes/4.0.1>.

For release notes about 4.0 (including feature summary, technology dependencies, etc) go to the [Delegation Service 4.0 Release Notes](#)¹.

2. Changes Summary

No changes have occurred for the Delegation Service.

3. Bug Fixes

No bugs were fixed for Delegation Service.

4. Known Problems

Following known issue exists:

- [Bug 4126](#):²Wrong method invoked when Delegation Service is used

5. For More Information

Click [here](#)³ for more information about this component.

¹ http://www.globus.org/toolkit/docs/4.0/security/delegation/WS_AA_Delegation_Service_Release_Notes.html

² http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=4126

³ [index.html](#)

Chapter 17. GT4 Delegation Service Release Notes

1. Component Overview

The Delegation Service is a new component in Globus Toolkit 4.0. This component provides an interface for delegation of credentials to a hosting environment. This enables a single delegated credential to be shared across multiple invocations of services on that hosting environment (e.g. it could be used for multiple GRAM job submissions or across GRAM and RFT submissions.) It also provides a means for credential renewal.

2. Feature Summary

Features new in GT 4.0

- Provides an interface for the delegation and renewal of credentials to a host.
- Allows for a single delegated credential to be reused across multiple service invocations (e.g. GRAM jobs).

Other Supported Features

- The Delegation Service is a new component in GT 4.0.

Deprecated Features

- The Delegation Service is a new component in GT 4.0.

3. Changes Summary

3.1. New command line client names

The command line scripts have changed names to "globus-credential-delegate" and "globus-credential-refresh".

3.2. API Changes

- The function `getTokenFromRequest()` was removed from the public API.
- A function for delegating a credential whose lifetime is equal to that of the issuing credential was added to the `DelegationUtil` class.
- Function signatures were changed to take a client security descriptor parameter.

3.3. Transport security is used by default

The *transport security* (HTTPS) mechanism is now assumed as the default security mechanism. Delegation service clients will now automatically use this mechanism and will fall back to GSI Secure Message if pointed at an HTTP endpoint.

3.4. Internationalization

The delegation service has been internationalized.

3.5. Grid map file not specified by default

Delegation service is not configured with a *grid map file* location by default.

4. Bug Fixes

- File permissions for persisted delegated credentials are now set before writing the credentials to disk.
- [Bug 2537](http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=2537)¹: globus-credential-delegate should honor -help
- [Bug 2581](http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=2581)²: NPE in DelegationResource
- [Bug 2575](http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=2575)³: ConcurrentModificationException
- [Bug 2964](http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=2964)⁴: Command line option "-m" in delegate client
- [Bug 2966](http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=2966)⁵: Command line option "-d" in delegate client
- [Bug 3076](http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=3076)⁶: ArrayOutOBounds in delegate client when no argument is passed

5. Known Problems

- Persisted credentials that have expired and are never accessed are not cleaned up from disk. [Bug 3145](http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=3145)⁷.
- Inconsistency in delegation client command line arguments. [Bug 3077](http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=3077)⁸ and [Bug 2973](http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=2973)⁹.

6. Technology Dependencies

The Delegation Service depends on the following GT components:

- WS Authentication and Authorization
- Java WS Core

The Delegation Service depends on the following 3rd party software:

- Apache Axis

¹ http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=2537

² http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=2581

³ http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=2575

⁴ http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=2964

⁵ http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=2966

⁶ http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=3076

⁷ http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=3145

⁸ http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=3077

⁹ http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=2973

7. Tested Platforms

Tested Platforms for Delegation Service

- Windows XP
- Linux (Red Hat 7.3)

Tested Containers for Delegation Service

- Java WS Core container
- Tomcat 5.0.30

8. Backward Compatibility Summary

The Delegation Service is a new component in GT 4.0

9. For More Information

Click [here](#)¹⁰ for more information about this component.

¹⁰ [index.html](#)

GT 4.0 Security Glossary

C

Certificate Authority (CA)	An entity that issues certificates.
CA Certificate	The CA's certificate. This certificate is used to verify signature on certificates issued by the CA. GSI typically stores a given CA certificate in <code>/etc/grid-security/certificates/<hash>.0</code> , where <code><hash></code> is the hash code of the CA identity.
CA Signing Policy	The CA signing policy is used to place constraints on the information you trust a given CA to bind to public keys. Specifically it constrains the identities a CA is trusted to assert in a certificate. In GSI the signing policy for a given CA can typically be found in <code>/etc/grid-security/certificates/<hash>.signing_policy</code> , where <code><hash></code> is the hash code of the CA identity. For more information see [add link].
certificate	A public key and 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.
Certificate Revocation List (CRL)	A list of revoked certificates generated by the CA that originally issued them. When using GSI this list is typically found in <code>/etc/grid-security/certificates/<hash>.r0</code> , where <code><hash></code> is the hash code of the CA identity.
certificate subject	A identifier for the certificate owner, e.g. <code>"/DC=org/DC=doegrids/OU=People/CN=John Doe 123456"</code> . The subject is part of the information the CA binds to a public key when creating a certificate.
credentials	The combination of a certificate and the matching private key.

E

End Entity Certificate (EEC)	A certificate belonging to a non-CA entity, e.g. you, me or the computer on your desk.
------------------------------	--

G

GAA Configuration File	A file that configures the Generic Authorization and Access control GAA libraries. When using GSI this file is typically found in <code>/etc/grid-security/gsi-gaa.conf</code> .
grid map file	A file containing entries mapping certificate subjects to local user names. This file can also serve as a access control list for GSI enabled services and is typically found in <code>/etc/grid-security/grid-mapfile</code> . For more information see the Gridmap file ¹¹ .

¹¹ http://www.globus.org/toolkit/docs/4.0/security/prewsaa/Pre_WS_AA_Public_Interfaces.html#prewsaa-env-gridmapfile

grid security directory The directory containing GSI configuration files such as the GSI authorization callout configuration and GAA configuration files. Typically this directory is `/etc/grid-security`. For more information see [Grid security directory](#)¹².

GSI authorization callout configuration file A file that configures authorization callouts to be used for mapping and authorization in GSI enabled services. When using GSI this file is typically found in `/etc/grid-security/gsi-authz.conf`.

H

host certificate An EEC belonging to a host. When using GSI this certificate is typically stored in `/etc/grid-security/hostcert.pem`. For more information on possible host certificate locations see the [Credentials](#)¹³.

host credentials The combination of a host certificate and its corresponding private key..

P

private key The private part of a key pair. Depending on the type of certificate the key corresponds to it may typically be found in `$HOME/.globus/userkey.pem` (for user certificates), `/etc/grid-security/hostkey.pem` (for host certificates) or `/etc/grid-security/<service>/<service>key.pem` (for service certificates). For more information on possible private key locations see the [Credentials](#)¹⁴

proxy certificate A short lived certificate issued using a EEC. A proxy certificate typically has the same effective subject as the EEC that issued it and can thus be used in its stead. GSI uses proxy certificates for single sign on and delegation of rights to other entities.

For more information about types of proxy certificates and their compatibility in different versions of GT, see <http://dev.globus.org/wiki/Security/ProxyCertTypes>.

proxy credentials The combination of a proxy certificate and its corresponding private key. GSI typically stores proxy credentials in `/tmp/x509up_u<uid>`, where `<uid>` is the user id of the proxy owner.

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

S

service certificate A EEC for a specific service (e.g. FTP or LDAP). When using GSI this certificate is typically stored in `/etc/grid-security/<service>/<service>cert.pem`. For more information on possible service certificate locations see the [Credentials](#)¹⁵.

service credentials The combination of a service certificate and its corresponding private key.

¹² http://www.globus.org/toolkit/docs/4.0/security/prewsaa/Pre_WS_AA_Public_Interfaces.html#prewsaa-env-gridsecurity

¹³ http://www.globus.org/toolkit/docs/4.0/security/prewsaa/Pre_WS_AA_Public_Interfaces.html#prewsaa-env-credentials

¹⁴ http://www.globus.org/toolkit/docs/4.0/security/prewsaa/Pre_WS_AA_Public_Interfaces.html#prewsaa-env-credentials

¹⁵ http://www.globus.org/toolkit/docs/4.0/security/prewsaa/Pre_WS_AA_Public_Interfaces.html#prewsaa-env-credentials

T

transport-level security	Uses transport-level security (TLS) mechanisms.
trusted CAs directory	The directory containing the CA certificates and signing policy files of the CAs trusted by GSI. Typically this directory is <code>/etc/grid-security/certificates</code> . For more information see Grid security directory ¹⁶ .

U

user certificate	A EEC belonging to a user. When using GSI this certificate is typically stored in <code>\$HOME/.globus/usercert.pem</code> . For more information on possible user certificate locations see Credentials ¹⁷ .
user credentials	The combination of a user certificate and its corresponding private key.

¹⁶ http://www.globus.org/toolkit/docs/4.0/security/prewsaa/Pre_WS_AA_Public_Interfaces.html#prewsaa-env-gridsecurity

¹⁷ http://www.globus.org/toolkit/docs/4.0/security/prewsaa/Pre_WS_AA_Public_Interfaces.html#prewsaa-env-credentials