GT 4.0: Credential Management: MyProxy

GT 4.0: Credential Management: MyProxy

Table of Contents

List of Tables

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.

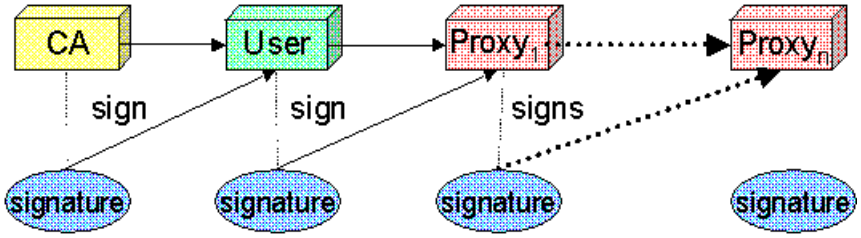
We have also prototyped the use of cryptographic smartcards in conjunction with GSI. This allows users to store their private key on a smartcard rather than in a file system, making it still more difficult for others to gain access to the key.

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 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 certificate[6] 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 `/etc/grid-security/certificates/<hash>.0`, where <hash> 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 `/etc/grid-security/certificates/<hash>.signing_policy`, where <hash> 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 `/etc/grid-security/certificates/<hash>.r0`, where <hash> is the hash code of the CA identity.
certificate subject	A identifier for the certificate owner, e.g. "/DC=org/DC=doegrids/OU=People/CN=John Doe 123456". 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 `/etc/grid-security/gsi-gaa.conf`.
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 `/etc/grid-security/grid-mapfile`. 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 `/etc/grid-security`. 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 `/etc/grid-security/gsi-authz.conf`.

[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.

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.

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].

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

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.

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

Chapter 2. GT 4.0 Release Notes: MyProxy

1. Component Overview

MyProxy is an online credential repository. You can store X.509 *proxy credentials* in the MyProxy repository, protected by a passphrase, for later retrieval over the network. This eliminates the need for manually copying *private key* and certificate files between machines. MyProxy can also be used for authentication to grid portals and credential renewal with job managers.

2. Feature Summary

Features new in release 4.0

- This is the first Globus Toolkit release that includes MyProxy.

Other Supported Features

- Users can store and retrieve multiple X.509 *proxy credentials* using **myproxy-init** and **myproxy-logon**.
- Users can store and retrieve multiple X.509 end-entity credentials using **myproxy-store** and **myproxy-retrieve**.
- Administrators can load the repository with X.509 end-entity credentials on the users' behalf using **myproxy-admin-load-credential**.
- Administrators can use the **myproxy-admin-adduser** command to create *user credentials* and load them into the MyProxy repository.
- Users and administrators can set access control policies on the credentials in the repository.
- If allowed by policy, job managers (such as Condor-G) can renew credentials before they expire.
- The MyProxy server enforces local site passphrase policies using a configurable external call-out.

Deprecated Features

- None

3. Bug Fixes

This is the first release of the Globus Toolkit that includes MyProxy.

4. Known Problems

None.

5. Technology Dependencies

MyProxy depends on the following GT component:

- Pre-WS Authentication and Authorization

MyProxy depends on the following 3rd party software:

- None

6. Tested Platforms

Tested Platforms for MyProxy

- Mac OS X 10.3
- i686 GNU/Linux
- ia64 GNU/Linux

7. Backward Compatibility Summary

Protocol changes since GT 3.2

- MyProxy was not included in GT 3.2.

API changes since GT 3.2

- MyProxy was not included in GT 3.2.

Exception changes since GT 3.2

- Not applicable

Schema changes since GT 3.2

- Not applicable

8. For More Information

Click here[1] for more information about this component.

[1] index.html

Chapter 3. GT 4.0.1 Incremental Release Notes: MyProxy

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 MyProxy 4.0 Release Notes[1].

2. Changes Summary

The following changes have occurred for MyProxy:

- Updated GPT package from 1.17 to 2.3.
- Fixed myproxy-init compatibility with Java CoG grid-proxy-init in PATH (using grid-proxy-init -hours instead of -valid).
- Fixed error message on server-side authorized_renewers check.
- Added client-side reverse lookup on server IP address for authorization consistency with other GSI clients.
- Added native PAM support (not requiring SASL).
- Added support for managing trusted *CA certificates* using myproxy-logon/myproxy-retrieve -T option
- Added the myproxy-replicate utility for managing multiple myproxy-server repository replicas for high availability.
- Added myproxy_version() and myproxy_check_version() functions to verify headers match the shared library in use.
- Fixed bug in 1.17 that caused myproxy-get-delegation not to be included in binary GPT installs.
- Use system getopt_long() if available; otherwise, use included getopt_long from NetBSD instead of GNU version as previous.
- Fixed server side bug where the default_key_retrievers policy was always applied even if the credential had a key retriever policy.
- Fixed myproxy_get_delegation() signature to maintain API-level compatibility.
- Added dynamic buffer management for improved handling of large messages.

3. Bug Fixes

The following bug was fixed for MyProxy:

[1] http://www.globus.org/toolkit/docs/4.0/security/myproxy/Cred_Mgmt_MyProxy_Release_Notes.html

- Bug 2939:[2] The MyProxy commands ignore invalid options rather than returning an error message.

4. Known Problems

The following problem is known to exist for MyProxy at the time of the 4.0.1 release:

- Bug 2709:[3] The MyProxy package isn't internationalized.

5. For More Information

Click here[4] for more information about this component.

[2] http://bugzilla.globus.org/globus/show_bug.cgi?id=2939
[3] http://bugzilla.globus.org/globus/show_bug.cgi?id=2709
[4] index.html

Chapter 4. GT 4.0.2 Incremental Release Notes: MyProxy

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 MyProxy 4.0 Release Notes[1].

2. Changes Summary

The following changes have occurred for MyProxy:

- Updated GPT package from 2.3 to 3.5.
- API changes: new trusted_retrievers member for struct myproxy_creds and struct myproxy_request_t
- Added ability for myproxy-server to act as a CA by setting certificate_issuer options in myproxy-server.config.
- Added support for CA username to DN resolution via LDAP using myproxy-server.config ca_ldap options.
- Dropped support for certificate_issuer option in myproxy-server.config; use certificate_issuer_cert instead.
- Added support for certificate-only authentication to the myproxy-server via myproxy-init -Z and myproxy-server.config trusted_retrievers options.
- Added myproxy-server authentication support for Pubcookie (http://www.pubcookie.org/) granting cookie via pubcookie_granting_cert and pubcookie_app_server_key myproxy-server.config options.
- Added myproxy-init --local_proxy option to create a local proxy credential after storing a credential on the myproxy-server.
- Added myproxy-init --keyfile/--certfile options.
- Install example myproxy-server.config to $GLOBUS_LOCATION/share/myproxy instead of potentially overwriting existing version in $GLOBUS_LOCATION/etc.
- In myproxy-logon, added support for writing credential to standard output via '-o -' option; also added --quiet option.
- Added myproxy-logon --no_credentials option to authenticate without retrieving credentials.
- Added certificate_mapapp call-out for mapping usernames to certificate subject distinguished names for the CA module.
- added certificate_extfile and certificate_extapp (call-out) for setting extensions in certificates issued by CA module.

[1] http://www.globus.org/toolkit/docs/4.0/security/myproxy/Cred_Mgmt_MyProxy_Release_Notes.html

3. Bug Fixes

The following bugs were fixed for MyProxy:

- Fixed memory errors found with myproxy-test -valgrind.
- Fixed segmentation fault on reverse lookup in client.
- Fixed SASL build problems on Darwin.
- Fixed client-side memory leak in GSI_SOCKET_authentication_init().
- Fixed compilation problem on platforms without setenv().
- Fixed static builds using 'gpt-build -static'.
- Fixes for SASL support.

4. Known Problems

The following problem is known to exist for MyProxy at the time of the 4.0.2 release:

- Bug 2709:[2] The MyProxy package isn't internationalized.

5. For More Information

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

[2] http://bugzilla.globus.org/globus/show_bug.cgi?id=2709
[3] index.html

Chapter 5. GT 4.0.3 Incremental Release Notes: MyProxy

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 MyProxy 4.0 Release Notes[1].

2. Changes Summary

The following changes have occurred for MyProxy since GT 4.0.2:

- Updated GPT package from 3.5 to 3.6.
- Added support for credential renewal via the MyProxy CA using myproxy-server.config authorized_renewers/default_renewers options
- Provided an access control list for storing credentials via the myproxy-server.config accepted_credentials_mapfile option
- Added support for VOMS attributes in myproxy-server.config policies
- Changed MyProxy CA toissue certificates valid starting 5 minutes in the past to account for possible clock skew between hosts
- Added check_multiple_credentials option in myproxy-server.config to allow clients to retrieve a credential for a given username even if the associated "credential name" isn't provided (via myproxy-logon --credname)

3. Bug Fixes

The following bugs have been fixed for MyProxy since GT 4.0.2:

- Bug 4647:[2] Fixed insecure temporary file handling in myproxy-admin-adduser.
- Fixed insecure input handling in myproxy-get-delegation.cgi example.

4. Known Problems

The following problem is known to exist for MyProxy at the time of the 4.0.3 release:

- Bug 2709:[3] The MyProxy package isn't internationalized.

[1] http://www.globus.org/toolkit/docs/4.0/security/myproxy/Cred_Mgmt_MyProxy_Release_Notes.html
[2] http://bugzilla.globus.org/globus/show_bug.cgi?id=4647
[3] http://bugzilla.globus.org/globus/show_bug.cgi?id=2709

5. For More Information

Click here[4] for more information about this component.

[4] index.html

Chapter 6. GT 4.0.4 Incremental Release Notes: MyProxy

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 MyProxy 4.0 Release Notes[1].

2. Changes Summary

- Updated GPT package from 3.6 to 3.7.
- Verify that credentials in the myproxy-server repository are still valid (i.e., not revoked) before performing delegation
- Added myproxy-admin-query --invalid option for listing, locking, or removing invalid credentials from repository
- Optionally check OCSP status of stored credentials before performing delegation via myproxy-server.config ocsp settings; requires GT 3.2 (OpenSSL 0.9.7) or later
- Updated etc.init.d.myproxy script to use pidfile to locate server to stop (rather than searching ps output), include a restart option, and exit with error if $GLOBUS_LOCATION isn't set
- If the myproxy-server hostname given by $MYPROXY_SERVER or the -s option resolves to multiple IP addresses, clients will connect to each address until a connection is established or all fail
- Added accepted_credentials_mapapp call-out version of accepted_credentials_mapfile in myproxy-server.config
- Now support unencrypted, although still signed, Pubcookie granting cookies as passwords
- Added myproxy-server.config syslog_ident option
- Improved MyProxy CA logging
- Added myproxy-server --listen to specify host/ip to bind to

3. Bug Fixes

- Bug 280:[2] fix handling of usernames containing '/', '-', and '.' characters; note this required a change to the myproxy-server repository format, so credential data files written by a new myproxy-server won't be readable by an older myproxy-server

[1] http://www.globus.org/toolkit/docs/4.0/security/myproxy/Cred_Mgmt_MyProxy_Release_Notes.html
[2] http://bugzilla.ncsa.uiuc.edu/show_bug.cgi?id=280

4. Known Problems

- Bug 2709:[3] The MyProxy package isn't internationalized.

5. For More Information

Click here[4] for more information about this component.

[3] http://bugzilla.globus.org/globus/show_bug.cgi?id=2709
[4] index.html

Chapter 7. GT 4.0 MyProxy: System Administrator's Guide

1. Introduction

This guide contains advanced configuration information for system administrators working with MyProxy. 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

MyProxy is built and installed as part of a default GT 4.0 installation. For basic installation instructions, see the GT 4.0 System Administrator's Guide[2]. No extra installation steps are required for this component.

2.1. Building and Installing only MyProxy

If you wish to install MyProxy without installing the rest of the Globus Toolkit, follow the instructions in the GT 4.0 System Administrator's Guide[3] with the following changes:

1. First, you do not need Ant, a JDK, or a JDBC database to build only MyProxy.
2. Second, instead of running "make", run:

```
globus$ make gsi-myproxy
```

This will install the MyProxy client and server programs. For client-only installations, simply do not configure or use the installed server.

3. Configuring

A typical MyProxy configuration has one dedicated myproxy-server for the site, with MyProxy clients installed on all systems where other Globus Toolkit client software is installed.

No additional configuration is required to use MyProxy clients after they are installed, although you may want to set the MYPROXY_SERVER environment variable to the hostname of your myproxy-server in the default user environment on your systems.

To configure the myproxy-server you must modify the myproxy-server.config template provided at `$GLOBUS_LOCATION/share/myproxy/myproxy-server.config` and copy it to `/etc/myproxy-server.config` (if you have root access) or `$GLOBUS_LOCATION//etc/myproxy-server.config` (if you don't have root

[1] ../../admin/docbook/
[2] http://www.globus.org/toolkit/docs/4.0/admin/docbook/
[3] http://www.globus.org/toolkit/docs/4.0/admin/docbook/

access). *If you skip this step, your myproxy-server will not start.* To enable all myproxy-server features uncomment the provided sample policy at the top of the myproxy-server.config config file, as follows:

```
#
# Complete Sample Policy
#
# The following lines define a sample policy that enables all
# myproxy-server features.  See below for more examples.
accepted_credentials      "*"
authorized_retrievers     "*"
default_retrievers        "*"
authorized_renewers       "*"
default_renewers          "none"
authorized_key_retrievers  "*"
default_key_retrievers    "none"
trusted_retrievers        "*"
default_trusted_retrievers "none"
```

Please see below for additional documentation on the myproxy-server.config options.

The `myproxy-server.config` file sets the policy for the **myproxy-server(8)**, specifying what credentials may be stored in the server's repository and who is authorized to retrieve credentials. By default, the **myproxy-server(8)** looks for this file in `/etc/myproxy-server.config` and if it is not found there, it looks in `$GLOBUS_LOCATION/etc/myproxy-server.config`. The **myproxy-server -c** option can be used to specify an alternative location. The file installed by default does not allow any requests.

The file also supports a **passphrase_policy_program** command for specifying an external program for evaluating the quality of users' passphrases. A sample program is installed in **$GLOBUS_LOCATION/share/myproxy/myproxy-passphrase-policy** but is not enabled by default.

Lines in the configuration file use limited regular expressions for matching the distinguished names (DNs) of classes of users. The limited regular expressions support the shell-style characters '*' and '?', where '*' matches any number of characters and '?' matches any single character.

The DN limited regexes should be delimited with double quotes ("DN regex").

The configuration file has the following types of lines:

Table 7.1. myproxy-server.config lines

accepted_credentials "DNregex"	Each of these lines allows any clients whose DNs match the given limited regex to connect to the myproxy-server and store credentials with it for future retrieval. Any number of these lines may appear. For backwards compatibility, these lines can also start with `allowed_clients` instead of `accepted_credentials`.
authorized_retrievers "DN regex"	Each of these lines allows the server administrator to set server-wide policies for authorized retrievers. If the client DN does not match the given limited regex, the client is not allowed to retrieve the credentials previously stored by a client. In addition to the server-wide policy, MyProxy also provides support for per-credential policy. The user can specify the regex DN of the allowed retrievers of the credential when uploading the credential (using **myproxy-init(1)**). The retrieval client DN must also match the user specified regex. In order to retrieve credentials the client also needs to know the name and pass phrase provided by the client when the credentials were stored. Any number of these lines may appear. For backwards compatibility, these lines can also start with `allowed_services` instead of `authorized_retrievers`.
default_retrievers "DN regex"	Each of these lines allows the server administrator to set server-wide default policies. The regex specifies the clients who can access the credentials. The default retriever policy is enforced if a per-credential policy is not specified on upload (using **myproxy-init(1)**). In other words, the client can override this policy for a credential on upload. The per-credential policy is enforced in addition to the server-wide policy specified by the `authorized_retrievers` line (which clients can not override). Any number of these lines may be present. For backwards compatibility, if no `default_retrievers` line is specified, the default policy is "*", which allows any client to pass the per-credential policy check. (The client must still pass the `authorized_retrievers` check).
authorized_renewers "DN regex"	Each of these lines allows the server administrator to set server-wide policies for authorized renewers. If the client DN does not match the given limited regex the client is not allowed to renew the credentials previously stored by a client. In addition to the server-wide policy, MyProxy also provides support for per-credential policy. The user can specify the regex DN of the allowed renewers of the credential on upload (using **myproxy-init(1)**). The renewal client DN must match both this regex and the user specified regex. In this case, the client must also already have a credential with a DN matching the DN of the credentials to be retrieved, to be used in a second authorization step (see the `-a` option for **myproxy-logon(1)**).
default_renewers "DN regex"	Each of these lines allows the server administrator to set server-wide default renewer policies. The regex specifies the clients who can renew the credentials. The default renewer policy is enforced if a per-credential policy is not specified on upload (using **myproxy-init(1)**). This is enforced in addition to the server-wide policy specified by the `authorized_renewers` line. Any number of these lines may appear. For backwards compatibility, if no `default_renewers` line is specified, the default policy is "*", which allows any client to pass the per-credential policy check. (The client must still pass the `authorized_renewers` check).
passphrase_policy_program full-path-to-script	This line specifies a program to run whenever a passphrase is set or changed for implementing a local password policy. The program is passed the new passphrase via stdin and is passed the following arguments: username, distinguished name, credential name (if any), per-credential retriever policy (if any), and per-credential renewal policy (if any). If the passphrase is acceptable, the program should exit with status 0. Otherwise, it should exit with non-zero status, causing the operation in progress (credential load, passphrase change) to fail with the error message provided by the program's stdout. Note: You must specify the full path to the external program. `$GLOBUS_LOCATION` can't be used in the `myproxy-server.config` file.

max_proxy_lifetime hours	This line specifies a server-wide maximum lifetime for retrieved proxy credentials. By default, no server-wide maximum is enforced. However, if this option is specified, the server will limit the lifetime of any retrieved proxy credentials to the value given.

4. Deploying

A sample SysV-style boot script for MyProxy is installed at $GLOBUS_LOCATION/share/myproxy/etc.init.d.myproxy. To install on Linux, copy the file to /etc/rc.d/init.d/myproxy and run 'chkconfig --add myproxy'. You will need to edit the file to set the GLOBUS_LOCATION environment variable correctly.

Alternatively, to run the myproxy server out of inetd or xinetd, you need to do the following as root:

- Add the entries in $GLOBUS_LOCATION/share/myproxy/etc.services.modifications to the /etc/services or /etc/inet/services file.
- Add the entries in $GLOBUS_LOCATION/share/myproxy/etc.inetd.conf.modifications to /etc/inetd.conf or /etc/inet/inetd.conf, or copy $GLOBUS_LOCATION/share/myproxy/etc.xinetd.myproxy to /etc/xinetd.d/myproxy. You'll need to modify the paths in the file according to your installation.
- Reactivate the inetd (or xinetd). This is typically accomplished by sending the SIGHUP signal to the daemon. Refer to the inetd or xinetd man page for your system.

5. Testing

To verify your myproxy-server installation and configuration, you can run the myproxy-server directly from your shell. If using a *host certificate*, you will need to run the myproxy-server as root. First, make sure your Globus environment is setup in your shell. Set the GLOBUS_LOCATION environment variable to the location of your MyProxy installation. Then, depending on your shell, run one of the following commands.

For csh shells:

source $GLOBUS_LOCATION/etc/globus-user-env.csh

For sh shells:

. $GLOBUS_LOCATION/etc/globus-user-env.sh

Then, run **$GLOBUS_LOCATION/sbin/myproxy-server -d**. The `-d` argument runs the myproxy-server in debug mode. It will write debugging messages to the terminal and exit after servicing a single request. You'll need to start it once for each test request. In another shell, you can run the MyProxy client programs to test the server.

If run without the `-d` argument, the myproxy-server program will start up and background itself. It accepts connections on TCP port 7512, forking off a separate child to handle each incoming connection. It logs information via the syslog service under the daemon facility.

6. Security Considerations

You should choose a well-protected host to run the myproxy-server on. Consult with security-aware personnel at your site. You want a host that is secured to the level of a Kerberos KDC, that has limited user access, runs limited services, and is well monitored and maintained in terms of security patches.

For a typical myproxy-server installation, the host on which the myproxy-server is running must have /etc/grid-security created and a *host certificate* installed. In this case, the myproxy-server will run as root so it can access the host certificate and key.

7. Troubleshooting

Please refer to the MyProxy user manual[4].

[4] ../../security/myproxy/user-index.html#s-myproxy-user-troubleshooting

Chapter 8. GT 4.0 MyProxy: User's Guide

1. Introduction

The GridFTP User's Guide provides general end user-oriented information.

2. Usage scenarios

2.1. Storing a credential in the MyProxy repository

Rather than storing your X.509 credentials (certificate and *private key*) on each machine you use, you can store them in a MyProxy repository and retrieve a *proxy credential* from the MyProxy repository when needed.

To store a credential in the MyProxy repository, run the **myproxy-init** command on a computer where your Grid credentials are located. For example:

```
$ myproxy-init -a -s myproxy.ncsa.uiuc.edu
Your identity: /C=US/O=National Computational Science Alliance/CN=Jim Basney
Enter GRID pass phrase for this identity:
Creating proxy ........................................ Done
Your proxy is valid until Fri Sep 13 13:52:56 2002
Enter MyProxy Pass Phrase:
Verifying password - Enter MyProxy Pass Phrase:
A proxy valid for 168 hours (7.0 days) for user jbasney now exists on myproxy.ncsa
```

The **myproxy-init** command prompts first for the pass phrase of your private key (similar to **grid-proxy-init**) and then prompts twice for a new pass phrase to use to secure the credentials on the MyProxy server. By default, the credential is stored under your Unix username (jbasney in the example above) for 7 days and can be used to retrieve credentials with 12 hour lifetimes. Section 3, “Command-line tools” below lists all the available options for the myproxy-init command.

2.2. Retrieving a credential from the MyProxy repository

Once you've stored a credential in the MyProxy repository, you can retrieve a proxy credential whenever you need one with the **myproxy-logon** command. For example:

```
$ myproxy-logon -s myproxy.ncsa.uiuc.edu
Enter MyProxy Pass Phrase:
A proxy has been received for user jbasney in /tmp/x509up_u500
```

The **myproxy-logon** command prompts for the pass phrase you set previously with **myproxy-init**, retrieves a proxy credential for you, and stores it in the correct default location for use with other Globus Toolkit programs. The MyProxy Command Reference lists all the available options for the **myproxy-logon** command.

3. Command-line tools

Please see the MyProxy Command Reference.

4. Graphical user interfaces

MyProxy does not have a GUI.

5. Troubleshooting

When troubleshooting a MyProxy problem, it is important to consult the myproxy-server logs. If you don't have access to the myproxy-server logs, please contact your myproxy-server administrator for help. The myproxy-server logs to the system logger (syslog) LOG_DAEMON facility. Alternatively, run

myproxy-server -d

from a terminal. In that mode, the myproxy-server will write debugging messages to the terminal and exit after servicing a single request.

The most common cause of MyProxy authentication problems is incorrect system clocks. GSI authentication is very sensitive to clock skew. Make sure your system clock is accurate (for example, by running NTP[1]) and your timezone is set correctly.

To debug GSI authentication problems, run

grid-proxy-init -debug -verify

from the terminal where you run the MyProxy clients, and run

grid-proxy-init -debug -verify -cert /etc/grid-security/hostcert.pem -key /etc/grid-security/hostkey.pem

as root on the myproxy-server machine (assuming you run the myproxy-server as root).

The following common problems are documented below:

5.1. MyProxy server name does not match expected name.

This error appears as a mutual authentication failure or a server authentication failure, and the error message should list two names: the expected name of the MyProxy server and the actual authenticated name. By default, the MyProxy clients expect the MyProxy server to be running with a *host certificate* that matches the target hostname. This error can occur when running the MyProxy server under a non-host certificate or if the server is running on a machine with multiple hostnames. The MyProxy clients authenticate the identity of the MyProxy server to avoid sending passphrases and credentials to rogue servers.

If the expected name contains an IP address, your system is unable to do a reverse lookup on that address to get the canonical hostname of the server, indicating either a problem with that machine's DNS record or a problem with the resolver on your system.

If the server name shown in the error message is acceptable, set the MYPROXY_SERVER_DN environment variable to that name to resolve the problem.

[1] http://www.ntp.org/

5.2. Error in bind(): Address already in use

This error indicates that the myproxy-server port (default: 7512) is in use by another process, probably another myproxy-server instance. You can not run multiple instances of the myproxy-server on the same network port. If you want to run multiple instances of the myproxy-server on a machine, you can specify different ports with the -p option, and then give the same -p option to the MyProxy commands to tell them to use the myproxy-server on that port.

5.3. grid-proxy-init failed

This error indicates that the grid-proxy-init command failed when myproxy-init attempted to run it, which implies a problem with the underlying Globus installation. Run

grid-proxy-init -debug -verify

for more information.

5.4. User not authorized

An error from the myproxy-server saying you are "not authorized" to complete an operation typically indicates that the myproxy-server.config file settings are restricting your access to the myproxy-server. It is possible that the myproxy-server is running with the default myproxy-server.config file, which does not authorize any operations. See Section 3, “Configuring ” for more information.

Chapter 9. GT 4.0 MyProxy: Developer's Guide

1. Introduction

We recommend using the CoG Kits[1] when developing with MyProxy.

2. Before you begin

2.1. Feature summary

Features new in release 4.0

- This is the first Globus Toolkit release that includes MyProxy.

Other Supported Features

- Users can store and retrieve multiple X.509 *proxy credentials* using **myproxy-init** and **myproxy-logon**.
- Users can store and retrieve multiple X.509 end-entity credentials using **myproxy-store** and **myproxy-retrieve**.
- Administrators can load the repository with X.509 end-entity credentials on the users' behalf using **myproxy-admin-load-credential**.
- Administrators can use the **myproxy-admin-adduser** command to create *user credentials* and load them into the MyProxy repository.
- Users and administrators can set access control policies on the credentials in the repository.
- If allowed by policy, job managers (such as Condor-G) can renew credentials before they expire.
- The MyProxy server enforces local site passphrase policies using a configurable external call-out.

Deprecated Features

- None

2.2. Tested platforms

Tested Platforms for MyProxy

- Mac OS X 10.3
- i686 GNU/Linux
- ia64 GNU/Linux

[1] http://www.cogkit.org/

2.3. Backward compatibility summary

Protocol changes since GT 3.2

- MyProxy was not included in GT 3.2.

API changes since GT 3.2

- MyProxy was not included in GT 3.2.

Exception changes since GT 3.2

- Not applicable

Schema changes since GT 3.2

- Not applicable

2.4. Technology dependencies

MyProxy depends on the following GT component:

- Pre-WS Authentication and Authorization

MyProxy depends on the following 3rd party software:

- None

2.5. Security considerations

You should choose a well-protected host to run the myproxy-server on. Consult with security-aware personnel at your site. You want a host that is secured to the level of a Kerberos KDC, that has limited user access, runs limited services, and is well monitored and maintained in terms of security patches.

For a typical myproxy-server installation, the host on which the myproxy-server is running must have /etc/grid-security created and a *host certificate* installed. In this case, the myproxy-server will run as root so it can access the host certificate and key.

3. Architecture and design overview

The MyProxy system architecture and design is described in the following two publications:

- J. Basney, M. Humphrey, and V. Welch. The MyProxy Online Credential Repository[2]. Software: Practice and Experience, 2005.
- J. Novotny, S. Tuecke, and V. Welch. An Online Credential Repository for the Grid: MyProxy[3]. Proceedings of the Tenth International Symposium on High Performance Distributed Computing (HPDC-10), IEEE Press, August 2001.

[2] http://www.ncsa.uiuc.edu/~jbasney/myproxy-spe.pdf
[3] http://www.globus.org/alliance/publications/papers/myproxy.pdf

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[4].

5. Usage scenarios

MyProxy provides a solution for delegating credentials to Grid portals to allow the portal to authenticate to Grid services on the user's behalf. A Grid portal is a web server that provides an interface to Grid services, allowing users to submit compute jobs, transfer files, and query Grid information services from a standard web browser. For example:

- The Open Grid Computing Environment[5] (OGCE) collaboratory provides software for building grid computing portals using MyProxy.
- The GridSphere[6] credential manager portlet supports MyProxy.
- The Grid Portal Toolkit[7] interfaces with MyProxy using the Perl CoG Kit[8].
- The Extreme! Computing Lab[9]'s Proxy Manager Xportlet[10] interfaces with MyProxy.

6. Tutorials

There are no tutorials available at this time.

7. Debugging

Please refer to Section 5, “Testing” and Section 5, “Troubleshooting” for debugging information.

8. Troubleshooting

Please refer to Section 5, “Troubleshooting”.

9. Related Documentation

For additional information about MyProxy, see the MyProxy Project Home Page[11] at NCSA.

[4] Cred_Mgmt_MyProxy_Public_Interfaces.html
[5] http://www.ogce.org/
[6] http://www.gridsphere.org/
[7] https://gridport.npaci.edu/
[8] https://gridport.npaci.edu/cog/
[9] http://www.extreme.indiana.edu/
[10] http://www.extreme.indiana.edu/xportlets/project/index.shtml
[11] http://myproxy.ncsa.uiuc.edu/

Chapter 10. GT 4.0 Component Fact Sheet: Credential Management - MyProxy

1. Brief overview

MyProxy is an online credential repository. You can store X.509 *proxy credentials* in the MyProxy repository, protected by a passphrase, for later retrieval over the network. This eliminates the need for manually copying *private key* and certificate files between machines. MyProxy can also be used for authentication to grid portals and credential renewal with job managers.

2. Summary of features

Features new in release 4.0

- This is the first Globus Toolkit release that includes MyProxy.

Other Supported Features

- Users can store and retrieve multiple X.509 *proxy credentials* using **myproxy-init** and **myproxy-logon**.
- Users can store and retrieve multiple X.509 end-entity credentials using **myproxy-store** and **myproxy-retrieve**.
- Administrators can load the repository with X.509 end-entity credentials on the users' behalf using **myproxy-admin-load-credential**.
- Administrators can use the **myproxy-admin-adduser** command to create *user credentials* and load them into the MyProxy repository.
- Users and administrators can set access control policies on the credentials in the repository.
- If allowed by policy, job managers (such as Condor-G) can renew credentials before they expire.
- The MyProxy server enforces local site passphrase policies using a configurable external call-out.

Deprecated Features

- None

3. Usability summary

Usability improvements for MyProxy:

- This is the first Globus Toolkit release to include MyProxy.

4. Backward compatibility summary

Protocol changes since GT 3.2

- MyProxy was not included in GT 3.2.

API changes since GT 3.2

- MyProxy was not included in GT 3.2.

Exception changes since GT 3.2

- Not applicable

Schema changes since GT 3.2

- Not applicable

5. Technology dependencies

MyProxy depends on the following GT component:

- Pre-WS Authentication and Authorization

MyProxy depends on the following 3rd party software:

- None

6. Tested platforms

Tested Platforms for MyProxy

- Mac OS X 10.3
- i686 GNU/Linux
- ia64 GNU/Linux

7. Associated standards

Associated standards for MyProxy:

- RFC 3820[1] Proxy Certificates
- RFC 2246[2] TLS

8. For More Information

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

[1] http://www.faqs.org/rfcs/rfc3820.html
[2] http://www.faqs.org/rfcs/rfc2246.html
[3] index.html

Chapter 11. GT 4.0 Component Guide to Public Interfaces: MyProxy

1. Semantics and syntax of APIs

A Java API[1] is available.

2. Semantics and syntax of the WSDL

MyProxy does not have a WSDL interface.

3. Command-line tools

Please see the MyProxy Command Reference.

4. Graphical User Interface

MyProxy does not have a GUI.

5. Semantics and syntax of domain-specific interface

MyProxy does not provide any domain-specific interfaces.

6. Configuration interface

A typical MyProxy configuration has one dedicated myproxy-server for the site, with MyProxy clients installed on all systems where other Globus Toolkit client software is installed.

No additional configuration is required to use MyProxy clients after they are installed, although you may want to set the MYPROXY_SERVER environment variable to the hostname of your myproxy-server in the default user environment on your systems.

To configure the myproxy-server you must modify the myproxy-server.config template provided at `$GLOBUS_LOCATION/share/myproxy/myproxy-server.config` and copy it to `/etc/myproxy-server.config` (if you have root access) or `$GLOBUS_LOCATION//etc/myproxy-server.config` (if you don't have root access). *If you skip this step, your myproxy-server will not start.* To enable all myproxy-server features uncomment the provided sample policy at the top of the myproxy-server.config config file, as follows:

```
#
# Complete Sample Policy
#
# The following lines define a sample policy that enables all
```

[1] http://www.globus.org/cog/distribution/1.2/api/org/globus/myproxy/package-summary.html

```
# myproxy-server features.  See below for more examples.
accepted_credentials      "*"
authorized_retrievers     "*"
default_retrievers        "*"
authorized_renewers       "*"
default_renewers          "none"
authorized_key_retrievers  "*"
default_key_retrievers    "none"
trusted_retrievers        "*"
default_trusted_retrievers "none"
```

Please see below for additional documentation on the myproxy-server.config options.

The `myproxy-server.config` file sets the policy for the **myproxy-server(8)**, specifying what credentials may be stored in the server's repository and who is authorized to retrieve credentials. By default, the **myproxy-server(8)** looks for this file in `/etc/myproxy-server.config` and if it is not found there, it looks in `$GLOBUS_LOCATION/etc/myproxy-server.config`. The **myproxy-server -c** option can be used to specify an alternative location. The file installed by default does not allow any requests.

The file also supports a **passphrase_policy_program** command for specifying an external program for evaluating the quality of users' passphrases. A sample program is installed in **$GLOBUS_LOCATION/share/myproxy/myproxy-passphrase-policy** but is not enabled by default.

Lines in the configuration file use limited regular expressions for matching the distinguished names (DNs) of classes of users. The limited regular expressions support the shell-style characters '*' and '?', where '*' matches any number of characters and '?' matches any single character.

The DN limited regexes should be delimited with double quotes ("DN regex").

The configuration file has the following types of lines:

Table 11.1. myproxy-server.config lines

accepted_credentials "DNregex"	Each of these lines allows any clients whose DNs match the given limited regex to connect to the myproxy-server and store credentials with it for future retrieval. Any number of these lines may appear. For backwards compatibility, these lines can also start with `allowed_clients` instead of `accepted_credentials`.
authorized_retrievers "DN regex"	Each of these lines allows the server administrator to set server-wide policies for authorized retrievers. If the client DN does not match the given limited regex, the client is not allowed to retrieve the credentials previously stored by a client. In addition to the server-wide policy, MyProxy also provides support for per-credential policy. The user can specify the regex DN of the allowed retrievers of the credential when uploading the credential (using **myproxy-init(1)**). The retrieval client DN must also match the user specified regex. In order to retrieve credentials the client also needs to know the name and pass phrase provided by the client when the credentials were stored. Any number of these lines may appear. For backwards compatibility, these lines can also start with `allowed_services` instead of `authorized_retrievers`.
default_retrievers "DN regex"	Each of these lines allows the server administrator to set server-wide default policies. The regex specifies the clients who can access the credentials. The default retriever policy is enforced if a per-credential policy is not specified on upload (using **myproxy-init(1)**). In other words, the client can override this policy for a credential on upload. The per-credential policy is enforced in addition to the server-wide policy specified by the `authorized_retrievers` line (which clients can not override). Any number of these lines may be present. For backwards compatibility, if no `default_retrievers` line is specified, the default policy is "*", which allows any client to pass the per-credential policy check. (The client must still pass the `authorized_retrievers` check).
authorized_renewers "DN regex"	Each of these lines allows the server administrator to set server-wide policies for authorized renewers. If the client DN does not match the given limited regex the client is not allowed to renew the credentials previously stored by a client. In addition to the server-wide policy, MyProxy also provides support for per-credential policy. The user can specify the regex DN of the allowed renewers of the credential on upload (using **myproxy-init(1)**). The renewal client DN must match both this regex and the user specified regex. In this case, the client must also already have a credential with a DN matching the DN of the credentials to be retrieved, to be used in a second authorization step (see the `-a` option for **myproxy-logon(1)**).
default_renewers "DN regex"	Each of these lines allows the server administrator to set server-wide default renewer policies. The regex specifies the clients who can renew the credentials. The default renewer policy is enforced if a per-credential policy is not specified on upload (using **myproxy-init(1)**). This is enforced in addition to the server-wide policy specified by the `authorized_renewers` line. Any number of these lines may appear. For backwards compatibility, if no `default_renewers` line is specified, the default policy is "*", which allows any client to pass the per-credential policy check. (The client must still pass the `authorized_renewers` check).
passphrase_policy_program full-path-to-script	This line specifies a program to run whenever a passphrase is set or changed for implementing a local password policy. The program is passed the new passphrase via stdin and is passed the following arguments: username, distinguished name, credential name (if any), per-credential retriever policy (if any), and per-credential renewal policy (if any). If the passphrase is acceptable, the program should exit with status 0. Otherwise, it should exit with non-zero status, causing the operation in progress (credential load, passphrase change) to fail with the error message provided by the program's stdout. Note: You must specify the full path to the external program. `$GLOBUS_LOCATION` can't be used in the `myproxy-server.config` file.

max_proxy_lifetime hours	This line specifies a server-wide maximum lifetime for retrieved proxy credentials. By default, no server-wide maximum is enforced. However, if this option is specified, the server will limit the lifetime of any retrieved proxy credentials to the value given.

7. Environment variable interface

Table 11.2. Environment variables

MYPROXY_SERVER	Specifies the hostname where the **myproxy-server** is running. This environment variable can be used in place of the `-s` option.
MYPROXY_SERVER_PORT	Specifies the port where the **myproxy-server** is running. This environment variable can be used in place of the `-p` option.
MYPROXY_SERVER_DN	Specifies the distinguished name (DN) of the **myproxy-server**. All MyProxy client programs authenticate the server's identity. By default, MyProxy servers run with host credentials, so the MyProxy client programs expect the server to have a distinguished name of the form "host/<fqhn>" or "myproxy/<fqhn>" (where <fqhn> is the fully-qualified hostname of the server). If the server is running with some other DN, you can set this environment variable to tell the MyProxy clients to accept the alternative DN.
X509_USER_CERT	Specifies a non-standard location for the certificate from which the *proxy credential* is created by **myproxy-init**. It also specifies an alternative location for the server's certificate. By default, the server uses `/etc/grid-security/hostcert.pem` when running as root or `~/.globus/usercert.pem` when running as non-root.
X509_USER_KEY	Specifies a non-standard location for the *private key* from which the proxy credential is created by **myproxy-init**. It also specifies an alternative location for the server's private key. By default the server uses `/etc/grid-security/hostkey.pem` when running as root or `~/.globus/userkey.pem` when running as non-root.
X509_USER_PROXY	Specifies an alternative location for the server's certificate and private key (in the same file). Use when running the server with a proxy credential. Note that the proxy will need to be periodically renewed before expiration to allow the **myproxy-server** to keep functioning. When the **myproxy-server** runs with a non-host credential, clients must have the `MYPROXY_SERVER_DN` environment variable set to the distinguished name of the certificate being used by the server.
GLOBUS_LOCATION	Specifies the root of the MyProxy installation, used to find the default location of the `myproxy-server.config` file and the credential storage directory.
LD_LIBRARY_PATH	The MyProxy server is typically linked dynamically with Globus security libraries, which must be present in the dynamic linker's search path. This typically requires `$GLOBUS_LOCATION/lib` to be included in the list in the `LD_LIBRARY_PATH` environment variable, which is set by the `$GLOBUS_LOCATION/libexec/globus-script-initializer` script, which should be called from any **myproxy-server** startup script. Alternatively, to set `LD_LIBRARY_PATH` appropriately for the Globus libraries in an interactive shell, source `$GLOBUS_LOCATION/etc/globus-user-env.sh` (for sh shells) or `$GLOBUS_LOCATION/etc/globus-user.env.csh` (for csh shells).
GT_PROXY_MODE	Set to "old" to use the "legacy globus proxy" format. By default, MyProxy uses the RFC 3820 compliant proxy (also known as "proxy draft compliant") format. If `GT_PROXY_MODE` is set to "old", then myproxy-init will store a legacy proxy and myproxy-logon will retrieve a legacy proxy (if possible). Note that if the repository contains a proxy certificate, rather than an end-entity certificate, the retrieved proxy will be of the same type as the stored proxy, regardless of the setting of this environment variable.

Chapter 12. GT 4.0 MyProxy: Quality Profile

1. Test coverage reports

Not yet available.

2. Code analysis reports

Not yet available.

3. Outstanding bugs

- Bug 2709[1]: myproxy internationalization

4. Bug Fixes

This is the first release of the Globus Toolkit that includes MyProxy.

5. Performance reports

- MyProxy Scalability Information[2]

[1] http://bugzilla.globus.org/globus/show_bug.cgi?id=2709
[2] http://myproxy.ncsa.uiuc.edu/scalability.html

Chapter 13. GT 4.0 Migrating Guide for MyProxy

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

1. Migrating from GT2

No special procedures are required for MyProxy installations migrating from GT2 to GT4. MyProxy is backward compatible.

2. Migrating from GT3

No special procedures are required for MyProxy installations migrating from GT3 to GT4. MyProxy is backward compatible.

GT 4.0 MyProxy Command Reference

Name

myproxy-init -- Store a *proxy credential* for later retrieval

```
myproxy-init
```

Tool description

The **myproxy-init** command uploads a credential to a **myproxy-server** for later retrieval. In the default mode, the command first prompts for the user's Grid pass phrase (if needed), which is used to create a proxy credential. The command then prompts for a MyProxy pass phrase, which will be required to later retrieve the credential. The MyProxy pass phrase must be entered a second time for confirmation. A credential with a lifetime of one week (by default) is then delegated to the **myproxy-server** and stored with the given MyProxy pass phrase. Proxy credentials with a default lifetime of 12 hours can then be retrieved by **myproxy-logon** using the MyProxy passphrase. The default behavior can be overridden by options specified below.

The **myproxy-init** command can also upload a credential to a **myproxy-server** to support credential renewal. Renewal allows a trusted service (for example, a batch job scheduler) to obtain a new credential for a user before the existing credential it has for that user expires. The **-R** argument to **myproxy-init** configures the credential for renewal by the specified service. Renewal requires two authentications. The renewing service must authenticate with its own credentials, matching the distinguished name specified by the **-R** argument, and must also authenticate with an existing credential that matches the distinguished name of the stored credential to retrieve a new credential.

A credential may be used either for retrieval or renewal, but not both. If both are desired, upload a different credential for each use with a different name, using the **-k** option.

The hostname where the **myproxy-server** is running must be specified by either defining the **MYPROXY_SERVER** environment variable or the **-s** option.

By default, **myproxy-init** will create a proxy credential from the user's end-entity credentials at ~/.globus/usercert.pem and ~/.globus/userkey.pem to delegate to the **myproxy-server**. To specify an alternate location for the source certificate and key to delegate, use the X509_USER_CERT and X509_USER_KEY environment variables. To use a proxy credential as the source of the delegation, set both environment variables to the location of the proxy credential. To delegate a "legacy globus proxy", set the GT_PROXY_MODE environment variable to "old".

Command syntax

```
myproxy-init [ options ]
```

Command options

Table 4. myproxy-init options

-h, --help	Displays command usage text and exits.
-u, --usage	Displays command usage text and exits.
-v, --verbose	Enables verbose debugging output to the terminal.
-V, --version	Displays version information and exits.
-s *hostname*, --pshost *hostname*	Specifies the hostname of the myproxy-server. This option is required if the **MYPROXY_SERVER** environment variable is not defined. If specified, this option overrides the **MYPROXY_SERVER** environment variable.
-p *port*, --psport *port*	Specifies the TCP port number of the **myproxy-server**. Default: 7512.
-P, --pidfile *path*	Specifies a file to write the pid to.
-l, --username	Specifies the MyProxy account under which the credential should be stored. by default, the command uses the value of the **LOGNAME** environment variable. Use this option to specify a different account username on the MyProxy server. The MyProxy username need not correspond to a real Unix username.
-c *hours*, --cred_lifetime *hours*	Specifies the lifetime of the credential stored on the **myproxy-server** in hours. Specify *0* for the maximum possible lifetime, i.e., the lifetime of the original credential. Default: 1 week (168 hours).
-t *hours*, --proxy_lifetime *hours*	Specifies the maximum lifetime of credentials retrieved from the **myproxy-server** using the stored credential. Default: 12 hours.
-d, --dn_as_username	Use the *certificate subject* (DN) as the default username, instead of the **LOGNAME** environment variable.
-a, --allow_anonymous_retrievers	Allow credentials to be retrieved with just pass phrase authentication. By default, only entities with credentials that match the **myproxy-server.config** default retriever policy may retrieve credentials. This option allows entities without existing credentials to retrieve a credential using pass phrase authentication by including "anonymous" in the set of allowed retrievers. The **myproxy-server.config** server-wide policy must also allow "anonymous" clients for this option to have an effect.
-A, --allow_anonymous_renewers	Allow credentials to be renewed by any client. Any client with a valid credential with a subject name that matches the stored credential may retrieve a new credential from the MyProxy repository if this option is given. Since this effectively defeats the purpose of proxy credential lifetimes, it is not recommended. It is included only for the sake of completeness.
-r *dn*, --retrievable_by *dn*	Allow the specified entity to retrieve credentials. By default, the argument will be matched against the common name (CN) of the client (for example: "Jim Basney"). Specify **-x** before this option to match against the full distinguished name (DN) (for example: "/C=US/O=National Computational Science Alliance/CN=Jim Basney") instead.
-R *dn*, --renewable_by *dn*	Allow the specified entity to renew credentials. By default, the argument will be matched against the common name (CN) of the client (for example: "condorg/modi4.ncsa.uiuc.edu"). Specify **-x** before this option to match against the full distinguished name (DN) (for example: "/C=US/O=National Computational Science Alliance/CN=condorg/modi4.ncsa.uiuc.edu") instead. This option implies **-n** since passphrase authentication is not used for credential renewal.
-x, --regex_dn_match	Specifies that the DN used by options **-r** and **-R** will be matched as a regular expression.

-X, --match_cn_only	Specifies that the DN used by options **-r** and **-R** will be matched against the Common Name (CN) of the subject.
-k *name*, --credname *name*	Specifies the credential name.
-K *description*	blank
--creddesc *description*	Specifies credential description.
-S, --stdin_pass	by default, the command prompts for a passphrase and reads the passphrase from the active tty. When running the command non-interactively, there may be no associated tty. Specifying this option tells the command to read passphrases from standard input without prompts or confirmation.

Name

myproxy-info -- Display information about credentials

`myproxy-info`

Tool description

The **myproxy-info** command displays information about a user's credentials stored on a **myproxy-server**. The user must have a valid proxy credential as generated by **grid-proxy-init** or retrieved by **myproxy-logon** when running this command.

Command syntax

`myproxy-info [ options ]`

Command options

Table 5. myproxy-info options

-h, --help	Displays command usage text and exits.
-u, --usage	Displays command usage text and exits.
-v, --verbose	Enables verbose debugging output to the terminal.
-V, --version	Displays version information and exits.
-s *hostname*, --pshost *hostname*	Specifies the hostname of the myproxy-server. This option is required if the **MYPROXY_SERVER** environment variable is not defined. If specified, this option overrides the **MYPROXY_SERVER** environment variable.
-p *port*, --psport *port*	Specifies the TCP port number of the **myproxy-server**. Default: 7512.
-l *name*, --username *name*	Specifies the MyProxy account to query. By default, the command uses the value of the **LOGNAME** environment variable. Use this option to specify a different account username on the MyProxy server. The MyProxy username need not correspond to a real Unix username.
-d, --dn_as_username	Use the certificate subject (DN) as the default username, instead of the **LOGNAME** environment variable.

Name

myproxy-logon -- Retrieve a credential

`myproxy-logon`

Tool description

The **myproxy-logon** command retrieves a credential from the **myproxy-server** that was previously stored using **myproxy-init**. In the default mode, the command prompts for the MyProxy pass phrase associated with the credential to be retrieved and stores the retrieved credential in the standard location (*/tmp/x509up_u<uid>*).

If the repository contains an end-entity certificate, this command will retrieve an RFC 3820 compliant proxy (also known as "proxy draft compliant impersonation proxy") by default. Set the the GT_PROXY_MODE environment variable to "old" to retrieve a "legacy globus proxy" instead. If the repository contains a *proxy certificate*, the retrieved proxy will always be of the same type as the stored proxy.

The **myproxy-logon** is also available under the name **myproxy-get-delegation** for backward compatibility.

Command syntax

`myproxy-logon [ options ]`

Command options

Table 6. myproxy-logon options

-h, --help	Displays command usage text and exits.
-u, --usage	Displays command usage text and exits.
-v, --verbose	Enables verbose debugging output to the terminal.
-V, --version	Displays version information and exits.
-s *hostname*, --pshost *hostname*	Specifies the hostname of the myproxy-server. This option is required if the **MYPROXY_SERVER** environment variable is not defined. If specified, this option overrides the **MYPROXY_SERVER** environment variable.
-p *port*, --psport *port*	Specifies the TCP port number of the **myproxy-server**. Default: 7512.
-l, --username	Specifies the MyProxy account under which the credential to retrieve is stored. By default, the command uses the value of the **LOGNAME** environment variable. Use this option to specify a different account username on the MyProxy server. The MyProxy username need not correspond to a real Unix username.
-d, --dn_as_username	Use the certificate subject (DN) as the default username, instead of the **LOGNAME** environment variable. When used with the **-a** option, the certificate subject of the authorization credential is used. Otherwise, the certificate subject of the default credential is used.
-t *hours*, --proxy_lifetime *hours*	Specifies the lifetime of credentials retrieved from the **myproxy-server** using the stored credential. The resulting lifetime is the shorter of the requested lifetime and the lifetime specified when the credential was stored using **myproxy-init**. Default: 12 hours.
-o *file*, --out *file*	Specifies where the retrieved proxy credential should be stored. If this option is not specified, the proxy credential will be stored in the default location (**/tmp/x509up_u<uid>**).
-a *file*, --authorization *file*	Specifies a credential to be used for authorizing the request instead of a passphrase. When renewing a credential, use this option to specify the existing, valid credential that you want to renew. Renewing a credential generally requires two certificate-based authentications. The client authenticates with its identity, using the credential in the standard location or specified by **X509_USER_PROXY** or **X509_USER_CERT** and **X509_USER_KEY** in addition to authenticating with the existing credential, in the location specified by this option, that it wants to renew.
-k *name*, --credname *name*	Specifies the name of the credential that is to be retrieved or renewed.
-S, --stdin_pass	by default, the command prompts for a passphrase and reads the passphrase from the active tty. When running the command non-interactively, there may be no associated tty. Specifying this option tells the command to read passphrases from standard input without prompts or confirmation.

Name

myproxy-store -- Store end-entity credential for later retrieval

`myproxy-store`

Tool description

The **myproxy-store** command uploads a credential to a **myproxy-server(8)** for later retrieval. Unlike **myproxy-init(1)**, this command transfers the private key over the network (over a private channel). In the default mode, the command will take the credentials found in ~/.globus/usercert.pem and ~/.globus/userkey.pem and store them in the **myproxy-server(8)** repository. Proxy credentials with a default lifetime of 12 hours can then be retrieved by **myproxy-logon(1)** using the credential passphrase. The default behavior can be overridden by options specified below.

The hostname where the **myproxy-server(8)** is running must be specified by either defining the **MYPROXY_SERVER** environment variable or the **-s** option.

Command syntax

`myproxy-store [ options ]`

Command options

Table 7. myproxy-store options

-h, --help	Displays command usage text and exits.
-u, --usage	Displays command usage text and exits.
-v, --verbose	Enables verbose debugging output to the terminal.
-V, --version	Displays version information and exits.
-s *hostname*, --pshost *hostname*	Specifies the hostname of the myproxy-server. This option is required if the **MYPROXY_SERVER** environment variable is not defined. If specified, this option overrides the **MYPROXY_SERVER** environment variable.
-p *port*, --psport *port*	Specifies the TCP port number of the **myproxy-server(8)**. Default: 7512.
-l, --username	Specifies the MyProxy account under which the credential should be stored. by default, the command uses the value of the **LOGNAME** environment variable. Use this option to specify a different account username on the MyProxy server. The MyProxy username need not correspond to a real Unix username.
-c *filename*, --certfile *filename*	Specifies the filename of the source certificate. This is a required parameter.
-y *filename*, --keyfile *filename*	Specifies the filename of the source private key. This is a required parameter.
-t *hours*, --proxy_lifetime *hours*	Specifies the maximum lifetime of credentials retrieved from the **myproxy-server(8)** using the stored credential. Default: 12 hours
-d, --dn_as_username	Use the certificate subject (DN) as the default username, instead of the **LOGNAME** environment variable.
-a, --allow_anonymous_retrievers	Allow credentials to be retrieved with just pass phrase authentication. by default, only entities with credentials that match the **myproxy-server.config(5)** default retriever policy may retrieve credentials. This option allows entities without existing credentials to retrieve a credential using pass phrase authentication by including "anonymous" in the set of allowed retrievers. The **myproxy-server.config(5)** server-wide policy must also allow "anonymous" clients for this option to have an effect.
-A, --allow_anonymous_renewers	Allow credentials to be renewed by any client. Any client with a valid credential with a subject name that matches the stored credential may retrieve a new credential from the MyProxy repository if this option is given. Since this effectively defeats the purpose of proxy credential lifetimes, it is not recommended. It is included only for sake of completeness.
-r *dn*, --retrievable_by *dn*	Allow the specified entity to retrieve credentials. by default, the argument will be matched against the common name (CN) of the client (for example: "Jim Basney"). Specify **-x** before this option to match against the full distinguished name (DN) (for example: "/C=US/O=National Computational Science Alliance/CN=Jim Basney") instead.
-E *dn*, --retrieve_key *dn*	Allow the specified entity to retrieve end-entity credentials. by default, the argument will be matched against the common name (CN) of the client (for example: "Jim Basney"). Specify **-x** before this option to match against the full distinguished name (DN) (for example: "/C=US/O=National Computational Science Alliance/CN=Jim Basney") instead.

-R *dn*, --renewable_by *dn*	Allow the specified entity to renew credentials. by default, the argument will be matched against the common name (CN) of the client (for example: "condorg/modi4.ncsa.uiuc.edu"). Specify **-x** before this option to match against the full distinguished name (DN) (for example: "/C=US/O=National Computational Science Alliance/CN=condorg/modi4.ncsa.uiuc.edu") instead. This option implies **-n** since passphrase authentication is not used for credential renewal.
-x, --regex_dn_match	Specifies that the DN used by options **-r** and **-R** will be matched as a regular expression.
-X, --match_cn_only	Specifies that the DN used by options **-r** and **-R** will be matched against the Common Name (CN) of the subject.
-k *name*, --credname *name*	Specifies the credential name.
-K *description*, --creddesc *description*	Specifies credential description.

Name

myproxy-retrieve -- Retrieve an end-entity credential

`myproxy-retrieve`

Tool description

The **myproxy-retrieve** command retrieves a credential directly from the **myproxy-server(8)** that was previously stored using **myproxy-init(1)** or **myproxy-store(1)**. Unlike **myproxy-logon(1)**, this command transfers the *private key* in the repository over the network (over a private channel). To obtain a proxy credential, we recommend using **myproxy-logon(1)** instead.

In the default mode, the command prompts for the pass phrase associated with the credential to be retrieved and stores the retrieved credential in the standard location (~/.globus/usercert.pem and ~/.globus/userkey.pem). You could then run **grid-proxy-init** to create a proxy credential from the retrieved credentials.

Command syntax

```
myproxy-retrieve [ options ]
```

Command options

Table 8. myproxy-retrieve options

-h, --help	Displays command usage text and exits.
-u, --usage	Displays command usage text and exits.
-v, --verbose	Enables verbose debugging output to the terminal.
-V, --version	Displays version information and exits.
-s *hostname*, --pshost *hostname*	Specifies the hostname of the myproxy-server. This option is required if the **MYPROXY_SERVER** environment variable is not defined. If specified, this option overrides the **MYPROXY_SERVER** environment variable.
-p *port*, --psport *port*	Specifies the TCP port number of the **myproxy-server(8)**. Default: 7512.
-l, --username	Specifies the MyProxy account under which the credential to retrieve is stored. by default, the command uses the value of the **LOGNAME** environment variable. Use this option to specify a different account username on the MyProxy server. The MyProxy username need not correspond to a real Unix username.
-d, --dn_as_username	Use the certificate subject (DN) as the default username, instead of the **LOGNAME** environment variable. When used with the **-a** option, the certificate subject of the authorization credential is used. Otherwise, the certificate subject of the default credential is used.
-t *hours*, --proxy_lifetime *hours*	Specifies the lifetime of credentials retrieved from the **myproxy-server(8)** using the stored credential. The resulting lifetime is the shorter of the requested lifetime and the lifetime specified when the credential was stored using **myproxy-init(1)**. Default: 12 hours.
-c *filename*, --certfile *filename*	Specifies the filename of where the certificate will be stored.
-y *filename*, --keyfile *filename*	Specifies the filename of where the private key will be stored.
-a *file*, --authorization *file*	Specifies a credential to be used for authorizing the request instead of a passphrase. When renewing a credential, use this option to specify the existing, valid credential that you want to renew. Renewing a credential generally requires two certificate-based authentications. The client authenticates with its identity, using the credential in the standard location or specified by **X509_USER_PROXY** or **X509_USER_CERT** and **X509_USER_KEY** in addition to authenticating with the existing credential, in the location specified by this option, that it wants to renew.
-k *name*, --credname *name*	Specifies the name of the credential that is to be retrieved or renewed.
-S, --stdin_pass	By default, the command prompts for a passphrase and reads the passphrase from the active tty. When running the command non- interactively, there may be no associated tty. Specifying this option tells the command to read passphrases from standard input without prompts or confirmation.

Name

myproxy-destroy -- Remove a credential from the repository

`myproxy-destroy`

Tool description

The **myproxy-destroy** command removes a credential from the **myproxy-server** that was previously stored using **myproxy-init**. The user must have a valid proxy credential as generated by **grid-proxy-init** or retrieved by **myproxy-logon** when running this command.

Command syntax

```
myproxy-destroy [ options ]
```

Command options

Table 9. myproxy-destroy options

-h, --help	Displays command usage text and exits.
-u, --usage	Displays command usage text and exits.
-v, --verbose	Enables verbose debugging output to the terminal.
-V, --version	Displays version information and exits.
-s *hostname*, --pshost *hostname*	Specifies the hostname of the myproxy-server. This option is required if the **MYPROXY_SERVER** environment variable is not defined. If specified, this option overrides the **MYPROXY_SERVER** environment variable.
-p *port*, --psport *port*	Specifies the TCP port number of the **myproxy-server**. Default: 7512.
-l, --username	Specifies the MyProxy account under which the credential to destroy is stored. By default, the command uses the value of the **LOGNAME** environment variable. Use this option to specify a different account username on the MyProxy server. The MyProxy username need not correspond to a real Unix username.
-d, --dn_as_username	Use the certificate subject (DN) as the default username, instead of the **LOGNAME** environment variable.
-k *name*, --credname *name*	Specifies name of the credential to be destroyed.

Name

myproxy-change-pass-phrase -- Change a credential's passphrase

`myproxy-change-pass-phrase`

Tool description

The **myproxy-change-pass-phrase** command changes the passphrase under which a credential is protected in the MyProxy repository. The command first prompts for the current passphrase for the credential, then prompts twice for the new passphrase. Only the credential owner can change a credential's passphrase. The user must have a valid proxy credential as generated by **grid-proxy-init** or retrieved by **myproxy-logon** when running this command.

Command syntax

`myproxy-change-pass-phrase [ options ]`

Command options

Table 10. myproxy-change-pass-phrase options

-h, --help	Displays command usage text and exits.
-u, --usage	Displays command usage text and exits.
-v, --verbose	Enables verbose debugging output to the terminal.
-V, --version	Displays version information and exits.
-s *hostname*, --pshost *host-name*	Specifies the hostname of the myproxy-server. This option is required if the **MYPROXY_SERVER** environment variable is not defined. If specified, this option overrides the **MYPROXY_SERVER** environment variable.
-p *port*, --psport *port*	Specifies the TCP port number of the **myproxy-server**. Default: 7512.
-l, --username	Specifies the MyProxy account under which the credential should be stored. by default, the command uses the value of the **LOGNAME** environment variable. Use this option to specify a different account username on the MyProxy server. The MyProxy username need not correspond to a real Unix username.
-d, --dn_as_username	Use the certificate subject (DN) as the default username, instead of the **LOGNAME** environment variable.
-k *name*, --credname *name*	Specifies the credential name.
-S, --stdin_pass	by default, the command prompts for a passphrase and reads the passphrase from the active tty. When running the command noninteractively, there may be no associated tty. Specifying this option tells the command to read pass-phrases from standard input without prompts or confirmation.

Name

myproxy-admin-adduser -- Add a new *user credential*

`myproxy-admin-adduser`

Tool description

The **myproxy-admin-adduser** command creates a new credential for a user and loads it into the MyProxy repository. It is a **perl** script that runs **grid-cert-request** (a standard Globus Toolkit program) and **grid-ca-sign** (from the Globus Simple CA package) to create the credential and then runs **myproxy-admin-load-credential** to load the credential into the MyProxy repository. The command prompts for the common name to be included in the new certificate (if the **-c** argument is not specified), the Globus Simple CA key password for signing the certificate, the MyProxy username (if the **-l** or **-d** arguments are not specified), and the MyProxy passphrase for the credential. Most of the command-line options for this command are passed directly to the **myproxy-admin-load-credential** command. The Globus Simple CA must be configured before using this command.

Command syntax

```
myproxy-admin-adduser [ options ]
```

Command options

Table 11. myproxy-admin-adduser options

-h	Displays command usage text and exits.
-u	Displays command usage text and exits.
-c *cn*	Specifies the Common Name for the new credential (for example: "Jim Basney").
-s *dir*	Specifies the location of the credential storage directory. The directory must be accessible only by the user running the **myproxy-server** process for security reasons. Default: /var/myproxy or $GLOBUS_LOCATION/var/myproxy.
-l *username*	Specifies the MyProxy account under which the credential should be stored.
-t *hours*	Specifies the maximum lifetime of credentials retrieved from the **myproxy-server** using the stored credential. Default: 12 hours.
-n	Disables passphrase authentication for the stored credential. If specified, the command will not prompt for a passphrase, the credential will not be encrypted by a passphrase in the repository, and the credential will not be retrievable using passphrase authentication with **myproxy-logon**. This option is used for storing renewable credentials and is implied by **-R**.
-d	Use the certificate subject (DN) as the username.
-a	Allow credentials to be retrieved with just pass phrase authentication. by default, only entities with credentials that match the **myproxy-server.config** default retriever policy may retrieve credentials. This option allows entities without existing credentials to retrieve a credential using pass phrase authentication by including "anonymous" in the set of allowed retrievers. The **myproxy-server.config** server-wide policy must also allow "anonymous" clients for this option to have an effect.
-A	Allow credentials to be renewed by any client. Any client with a valid credential with a subject name that matches the stored credential may retrieve a new credential from the MyProxy repository if this option is given. Since this effectively defeats the purpose of proxy credential lifetimes, it is not recommended. It is included only for sake of completeness.
-r *dn*	Allow the specified entity to retrieve credentials. By default, the argument will be matched against the common name (CN) of the client (for example: "Jim Basney"). Specify **-x** before this option to match against the full distinguished name (DN) (for example: "/C=US/O=National Computational Science Alliance/CN=Jim Basney") instead.
-R *dn*	Allow the specified entity to renew credentials. By default, the argument will be matched against the common name (CN) of the client (for example: "condorg/modi4.ncsa.uiuc.edu"). Specify **-x** before this option to match against the full distinguished name (DN) (for example: "/C=US/O=National Computational Science Alliance/CN=condorg/modi4.ncsa.uiuc.edu") instead. This option implies **-n** since passphrase authentication is not used for credential renewal.
-x	Specifies that the DN used by options **-r** and **-R** will be matched as a regular expression.
-X	Specifies that the DN used by options **-r** and **-R** will be matched against the Common Name (CN) of the subject.
-k *name*	Specifies the credential name.
-K *description*	Specifies credential description.

Name

myproxy-admin-change-pass -- Change credential passphrase

```
myproxy-admin-change-pass
```

Tool description

The **myproxy-admin-change-pass** command changes the passphrase used to encrypt a credential in the MyProxy repository. The command first prompts for the current passphrase for the credential, then prompts twice for the new passphrase. If an empty passphrase is given, the credential will not be encrypted. It accesses the repository directly and must be run on the machine where the **myproxy-server** is installed from the account that owns the repository.

Command syntax

```
myproxy-admin-change-pass [ options ]
```

Command options

Table 12. myproxy-admin-change-pass options

-h	Displays command usage text and exits.
-u	Displays command usage text and exits.
-s *dir*	Specifies the location of the credential storage directory. The directory must be accessible only by the user running the **myproxy-server** process for security reasons. Default: /var/myproxy or $GLOBUS_LOCATION/var/myproxy.
-l *username*	Specifies the MyProxy account under which the credential should be stored.
-k *name*	Specifies the credential name.
-S, --stdin_pass	by default, the command prompts for a passphrase and reads the passphrase from the active tty. When running the command non-interactively, there may be no associated tty. Specifying this option tells the command to read passphrases from standard input without prompts or confirmation.
-v, --verbose	Enables verbose debugging output to the terminal.
-V, --version	Displays version information and exits.

Name

myproxy-admin-query -- Query repository contents

`myproxy-admin-query`

Tool description

The **myproxy-admin-query** command displays information about the credentials stored in the MyProxy repository. It can also be used to remove credentials from the repository. It accesses the repository directly and must be run on the machine where the **myproxy-server** is installed from the account that owns the repository.

Command syntax

```
myproxy-admin-query [ options ]
```

Command options

Table 13. myproxy-admin-query options

-h, --help	Displays command usage text and exits.
-u, --usage	Displays command usage text and exits.
-v, --verbose	Enables verbose debugging output to the terminal.
-V, --version	Displays version information and exits.
-l *name*, --username *name*	Returns information on credentials for a single username. By default, the command returns information on all credentials for all usernames.
-k *name*, --credname *name*	Returns information on the credentials with the specified name.
-e *hours*, --expiring_in *hours*	Returns information on credentials with remaining lifetime less than the specified number of hours. For example, **-e** *0* will return all expired credentials.
-t *hours*, --time_left *hours*	Returns information on credentials with remaining lifetime greater than the specified number of hours.
-s *dir*, --storage *dir*	Specifies the location of the credential storage directory. The directory must be accessible only by the user running the **myproxy-server** process for security reasons. Default: /var/myproxy or $GLOBUS_LOCATION/var/myproxy.
-r, --remove	Remove the credentials matching the query from the repository. For example, *myproxy-admin-query -e 0 -r* will remove all expired credentials from the repository.
-L *'msg'*, --lock *'msg'*	Places the credentials matching the query under an administrative lock and specifies a message to be returned on access attempts. Be sure to put the message in quotes so it is captured as one argument to the command.
-U, --unlock	Removes any administrative locks for the credentials matching the query.

Name

myproxy-admin-load-credential -- Directly load repository

```
myproxy-admin-load-credential
```

Tool description

The **myproxy-admin-load-credential** command stores a credential directly in the local MyProxy repository. It must be run from the account that owns the repository. Many of the options are similar to **myproxy-init**. However, unlike **myproxy-init**, **myproxy-admin-load-credential** does not create a proxy from the source credential but instead directly loads a copy of the source credential into the repository. The pass phrase of the source credential is unchanged. Use **myproxy-admin-change-pass** to change the pass phrase after the credential is stored if desired. Proxy credentials with a default lifetime of 12 hours can then be retrieved by **myproxy-logon** using the MyProxy passphrase. The command's behavior is controlled by the following options.

Command syntax

```
myproxy-admin-load-credential [ options ]
```

Command options

Table 14. myproxy-admin-load-credential options

-h, --help	Displays command usage text and exits.
-u, --usage	Displays command usage text and exits.
-v, --verbose	Enables verbose debugging output to the terminal.
-V, --version	Displays version information and exits.
-s *dir*, --storage *dir*	Specifies the location of the credential storage directory. The directory must be accessible only by the user running the **myproxy-server** process for security reasons. Default: /var/myproxy or $GLOBUS_LOCATION/var/myproxy
-c *filename*, --certfile *filename*	Specifies the filename of the source certificate. This is a required parameter.
-y *filename*, --keyfile *filename*	Specifies the filename of the source private key. This is a required parameter.
-l *username*, --username *username*	Specifies the MyProxy account under which the credential should be stored. by default, the command uses the value of the **LOGNAME** environment variable. Use this option to specify a different account username on the MyProxy server. The MyProxy username need not correspond to a real Unix username.
-t *hours*, --proxy_lifetime *hours*	Specifies the maximum lifetime of credentials retrieved from the **myproxy-server** using the stored credential. Default: 12 hours.
-d, --dn_as_username	Use the certificate subject (DN) as the username.
-a, --allow_anonymous_retrievers	Allow credentials to be retrieved with just pass phrase authentication. by default, only entities with credentials that match the **myproxy-server.config** default retriever policy may retrieve credentials. This option allows entities without existing credentials to retrieve a credential using pass phrase authentication by including "anonymous" in the set of allowed retrievers. The **myproxy-server.config** server-wide policy must also allow "anonymous" clients for this option to have an effect.
-A, --allow_anonymous_renewers	Allow credentials to be renewed by any client. Any client with a valid credential with a subject name that matches the stored credential may retrieve a new credential from the MyProxy repository if this option is given. Since this effectively defeats the purpose of proxy credential lifetimes, it is not recommended. It is included only for sake of completeness.
-r *dn*, --retrievable_by *dn*	Allow the specified entity to retrieve credentials. By default, the argument will be matched against the common name (CN) of the client (for example: "Jim Basney"). Specify **-x** before this option to match against the full distinguished name (DN) (for example: "/C=US/O=National Computational Science Alliance/CN=Jim Basney") instead.
-R *dn*, --renewable_by *dn*	Allow the specified entity to renew credentials. By default, the argument will be matched against the common name (CN) of the client (for example: "condorg/modi4.ncsa.uiuc.edu"). Specify **-x** before this option to match against the full distinguished name (DN) (for example: "/C=US/O=National Computational Science Alliance/CN=condorg/modi4.ncsa.uiuc.edu") instead.
-x, --regex_dn_match	Specifies that the DN used by options **-r** and **-R** will be matched as a regular expression.
-X, --match_cn_only	Specifies that the DN used by options **-r** and **-R** will be matched against the Common Name (CN) of the subject.
-k *name*, --credname *name*	Specifies the credential name.

-K *description*, --creddesc *description*	Specifies credential description.

Name

myproxy-server -- Store credentials in an online repository

`myproxy-server`

Tool description

The **myproxy-server** is a server that runs on a trusted, secure host and manages a database of security credentials for use from remote sites. The **myproxy-init** program stores credentials with associated policies that specify credential lifetimes and who is authorized to retrieve credentials. The **myproxy-server.config** file sets server-wide policies that are used in conjunction with the policies set by **myproxy-init** to control who is authorized to store and retrieve credentials.

Command syntax

`myproxy-server [ options ]`

Command options

Table 15. myproxy-server options

-h, --help	Displays command usage text and exits.
-u, --usage	Displays command usage text and exits.
-v, --verbose	Enables verbose debugging output to the terminal.
-V, --version	Displays version information and exits.
-d, --debug	Run the server in debug mode. In this mode, the server will run in the foreground, will accept one connection, write log messages to the terminal while processing the incoming request, and exit after completing one request.
-p *port*, --port *port*	Specifies the TCP port number that the **myproxy-server** should listen on. Default: 7512.
-c *file*, --config *file*	Specifies the location of the **myproxy-server** configuration file. Default: /etc/myproxy-server.config or $GLOBUS_LOCATION/etc/myproxy-server.config.
-s *dir*, --storage *dir*	Specifies the location of the credential storage directory. The directory must be accessible only by the user running the **myproxy-server** process for security reasons. Default: /var/myproxy or $GLOBUS_LOCATION/var/myproxy.

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 `/etc/grid-security/certificates/<hash>.0`, where <hash> 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 `/etc/grid-security/certificates/<hash>.signing_policy`, where <hash> 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 `/etc/grid-security/certificates/<hash>.r0`, where <hash> is the hash code of the CA identity.
certificate subject	A identifier for the certificate owner, e.g. "/DC=org/DC=doegrids/OU=People/CN=John Doe 123456". 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 `/etc/grid-security/gsi-gaa.conf`.
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 `/etc/grid-security/grid-mapfile`. For more information see the Gridmap file[1].

[1] 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[2].

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[3].

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[4]

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.

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[5].

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

[2] http://www.globus.org/toolkit/docs/4.0/security/prewsaa/Pre_WS_AA_Public_Interfaces.html#prewsaa-env-gridsecurity
[3] http://www.globus.org/toolkit/docs/4.0/security/prewsaa/Pre_WS_AA_Public_Interfaces.html#prewsaa-env-credentials
[4] http://www.globus.org/toolkit/docs/4.0/security/prewsaa/Pre_WS_AA_Public_Interfaces.html#prewsaa-env-credentials
[5] 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 `/etc/grid-security/certificates`. For more information see Grid security directory[6].

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[7].

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

[6] http://www.globus.org/toolkit/docs/4.0/security/prewsaa/Pre_WS_AA_Public_Interfaces.html#prewsaa-env-gridsecurity
[7] http://www.globus.org/toolkit/docs/4.0/security/prewsaa/Pre_WS_AA_Public_Interfaces.html#prewsaa-env-credentials