GT 4.0: GSI-OpenSSH

GT 4.0: GSI-OpenSSH

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.

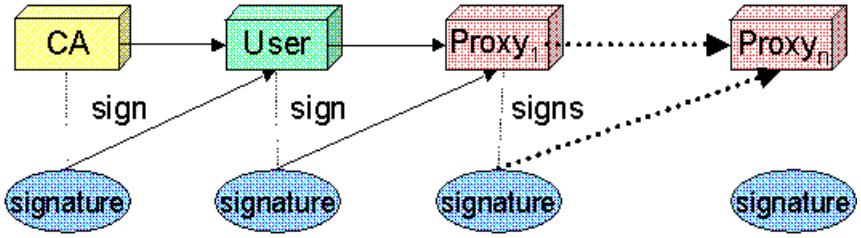
We have also prototyped the use of cryptographic smartcards in conjunction with GSI. This allows users to store their private key on a smartcard rather than in a file system, making it still more difficult for others to gain access to the key.

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: GSI-OpenSSH

1. Component Overview

GSI-OpenSSH is a modified version of OpenSSH that adds support for X.509 *proxy certificate* authentication and delegation, providing a single sign-on remote login and file transfer service. GSI-OpenSSH can be used to login to remote systems and transfer files between systems without entering a password, relying instead on a valid *proxy credential* for authentication. GSI-OpenSSH forwards proxy credentials to the remote system on login, so commands requiring proxy credentials (including GSI-OpenSSH commands) can be used on the remote system without the need to manually create a new proxy credential on that system.

2. Feature Summary

Features new in GT 4.0

- This is the first Globus Toolkit release that includes GSI-enabled OpenSSH.

Other Supported Features

- The **gsissh** command provides a secure remote login service with forwarding of X.509 *proxy credentials*.
- The **gsiscp** and **gsisftp** commands provide a secure file transfer service authenticated with X.509 proxy credentials, mimicking the **rcp/scp** and **ftp/sftp** commands.
- All standard OpenSSH features are supported, excluding Kerberos authentication. Kerberos authentication is *not* compatible with GSI-enabled OpenSSH.
- The GSI-OpenSSH server can replace the standard system SSH server in typical environments.
- If no username is given on the command-line, GSI-OpenSSH automatically determines the username that corresponds to the X.509 proxy *certificate subject* in the server's `grid-mapfile`.

Deprecated Features

- None

3. Bug Fixes

This is the first release of the Globus Toolkit that includes GSI-enabled OpenSSH.

4. Known Problems

None.

5. Technology Dependencies

GSI-enabled OpenSSH depends on the following GT components:

- Pre-WS Authentication and Authorization

GSI-enabled OpenSSH depends on the following 3rd party software:

- OpenSSH[1]

6. Tested Platforms

Tested Platforms for GSI-OpenSSH

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

7. Backward Compatibility Summary

Protocol changes since GT 3.2

- GSI-enabled OpenSSH was not included in GT 3.2.

API changes since GT 3.2

- GSI-enabled OpenSSH 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[2] for more information about this component.

[1] http://www.openssh.org/
[2] index.html

Chapter 3. GT 4.0.1 Incremental Release Notes: GSI-OpenSSH

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 GSI-OpenSSH 4.0 Release Notes[1].

2. Changes Summary

No changes have occurred for GSI-OpenSSH.

3. Bug Fixes

No bugs were fixed for GSI-OpenSSH.

4. Known Problems

No problems are known to exist for GSI-OpenSSH at the time of the 4.0.1 release.

5. For More Information

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

[1] http://www.globus.org/toolkit/docs/4.0/security/openssh/Util_OpenSSH_Release_Notes.html
[2] index.html

Chapter 4. GT 4.0.2 Incremental Release Notes: GSI-OpenSSH

1. Introduction

These release notes are for the incremental release 4.0.2. 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 4.0.2 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 GSI-OpenSSH 4.0 Release Notes[1].

2. Changes Summary

The following changes have occurred for GSI-OpenSSH:

- Updated GPT package from 3.5 to 3.7.
- Moved to OpenSSH 4.2p1.
- Incorporated openssh-4.2p1-gsskex-20050926-2.patch from Simon Wilkinson adding support for gssapi-keyex authentication method (http://www.sxw.org.uk/computing/patches/openssh.html).
- Incorporated High Performance Networking (HPN) patch (openssh-4.2p1-hpn11-none.diff) from Pittsburgh Supercomputing Center (http://www.psc.edu/networking/projects/hpn-ssh/) for improved network throughput over long and high bandwidth links.
- Enabled PAM password authentication by default, unless PAM library and headers are unusable.
- Added server-side GSIAllowLimitedProxy option to enable acceptance of limited proxy credentials.

3. Bug Fixes

The following bugs were fixed for GSI-OpenSSH:

- Bug 300:[2] Only delegate credentials if GSI authentication is used.

4. Known Problems

No problems are known to exist for GSI-OpenSSH at the time of the 4.0.2 release.

5. For More Information

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

[1] http://www.globus.org/toolkit/docs/4.0/security/openssh/Util_OpenSSH_Release_Notes.html
[2] http://bugzilla.ncsa.uiuc.edu/show_bug.cgi?id=300
[3] index.html

Chapter 5. GT 4.0.3 Incremental Release Notes: GSI-OpenSSH

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 GSI-OpenSSH 4.0 Release Notes[1].

2. Changes Summary

No changes have occurred for GSI-OpenSSH since GT 4.0.2.

3. Bug Fixes

No bugs have been fixed for GSI-OpenSSH since GT 4.0.2.

4. Known Problems

No problems are known to exist for GSI-OpenSSH at the time of the 4.0.3 release.

5. For More Information

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

[1] http://www.globus.org/toolkit/docs/4.0/security/openssh/Util_OpenSSH_Release_Notes.html
[2] index.html

Chapter 6. GT 4.0.4 Incremental Release Notes: GSI-OpenSSH

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 GSI-OpenSSH 4.0 Release Notes[1].

2. Changes Summary

- Updated GPT package from 3.7 to 3.9.
- Upgraded to OpenSSH 4.5p1.
- Upgraded to HPN12v14[2] patch.

3. Bug Fixes

- Bug 348:[3] Added sshd_config GssapiCredentialsPath option.
- Bug 4592:[4] Added support for Globus Authorization Callouts using service name "ssh".

4. Known Problems

No problems are known to exist for GSI-OpenSSH at the time of the 4.0.4 release.

5. For More Information

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

[1] http://www.globus.org/toolkit/docs/4.0/security/openssh/Util_OpenSSH_Release_Notes.html
[2] http://www.psc.edu/networking/projects/hpn-ssh/
[3] http://bugzilla.ncsa.uiuc.edu/show_bug.cgi?id=348
[4] http://bugzilla.globus.org/bugzilla/show_bug.cgi?id=4592
[5] index.html

Chapter 7. GT 4.0 GSI-OpenSSH: System Administrator's Guide

1. Introduction

This guide contains advanced configuration information for system administrators working with GSI-OpenSSH. 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!

This guide is meant solely to cover the GSI aspects of GSI-OpenSSH, and is not meant to be a full manual for OpenSSH itself. Please refer to the OpenSSH Home Page[2] for general documentation for OpenSSH.

2. Building and Installing

GSI-OpenSSH 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[3]. No extra installation steps are required for this component.

2.1. Optional Build-Time Configuration

You can optionally pass build-time configure options to the GSI-OpenSSH package using the --with-gsiopensshargs option when running configure for your GT 4.0 installation. For example:

```
./configure --prefix=$HOME/globus
      --with-gsiopensshargs="--with-pam"
```

No options are typically needed for client-only installations, but options are often needed for full server functionality. The following table lists suggested options for different platforms.

Table 7.1. GSI-OpenSSH build arguments

Platform	Configuration
Linux	--with-pam --with-md5-passwords --with-tcp-wrappers
Solaris	--with-pam --with-md5-passwords --with-tcp-wrappers
Irix	--with-tcp-wrappers
AIX	--with-tcp-wrappers

Note: If you enable PAM support with the --with-pam configuration option, be sure to also set "UsePAM yes" in $GLOBUS_LOCATION/etc/ssh/sshd_config after installation.

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

If you have an already configured and installed system-wide SSHD and you would like your build of GSI-OpenSSH to behave similarly, investigate the configure options available in GSI-OpenSSH and select those options that would add the functionality that your current SSHD possesses. Be aware that since GSI-OpenSSH is based on OpenSSH, the standard set of functionality is turned on by default.

Please do not attempt to override the following options:

```
--prefix
--sysconfdir
--with-globus
--with-globus-flavor
--with-ssl-dir
```

2.2. Building and Installing only GSI-OpenSSH

If you wish to install GSI-OpenSSH without installing the rest of the Globus Toolkit, follow the instructions in the GT 4.0 System Administrator's Guide[4] with the following changes. First, you do not need Ant, a JDK, or a JDBC database to build only GSI-OpenSSH. Second, instead of running "make", run:

```
globus$ make gsi-openssh
```

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

3. Configuring

The GSI-enabled OpenSSH software is installed with a default set of configuration files, described below. You may want to modify the `ssh_config` file before using the clients and the `sshd_config` file before using the server.

If the GSI-enabled OpenSSH install script finds existing SSH key pairs, it will create symbolic links to them rather than generating new key pairs. The SSH key pairs are not required for GSI authentication. However, if you wish to support other SSH authentication methods, make sure the sshd (running as root) can read the key pair files (i.e., beware of NFS mounts with root_squash). If running multiple sshds on a system, we recommend configuring them so they all use the same key pairs (i.e., use symbolic links) to avoid client-side confusion.

- `$GLOBUS_LOCATION/etc/ssh/moduli`

  moduli is a crypto parameter for generating keys.

- `$GLOBUS_LOCATION/etc/ssh/ssh_config`

  ssh_config contains options that are read by ssh, scp, and sftp at run-time. The installed version is the default provided by OpenSSH, with X11Forwarding enabled. You may need to customize this file for compatibility with your system SSH installation (i.e., compare it with /etc/ssh/ssh_config).

- `$GLOBUS_LOCATION/etc/ssh/ssh_host_key[.pub]`

  Your system's RSA public-/private-key pair for SSH protocol 1 communications.

[4] http://www.globus.org/toolkit/docs/4.0/admin/docbook/

- `$GLOBUS_LOCATION/etc/ssh/ssh_host_dsa[.pub]`

  Your system's DSA public-/private-key pair for SSH protocol 2 communications.

- `$GLOBUS_LOCATION/etc/ssh/ssh_host_rsa[.pub]`

  Your system's RSA public-/private-key pair for SSH protocol 2 communications.

- `$GLOBUS_LOCATION/etc/ssh/ssh_prng_cmds`

  ssh_prng_cmds contains paths to a number of files that ssh-keygen may need to use if your system does not have a built-in entropy pool (like /dev/random).

- `$GLOBUS_LOCATION/etc/ssh/sshd_config`

  sshd_config contains options that are read by sshd when it starts up. The installed version is the default provided by OpenSSH, with X11Forwarding enabled. You may need to customize this file for compatibility with your system SSH installation (i.e., compare it with /etc/ssh/sshd_config). For example, to enable PAM authentication, you will need to set "UsePAM yes" in this file.

4. Deploying

1. To install the GSI-Enabled OpenSSH Server on most systems, you must be a privileged user, such as root.

   ```
   sh$ /bin/su - root
   ```

   Note: If your system functions like this and you attempt to run these commands as a user other than root, these commands should fail.

2. (optional) Start a copy of your system's currently running SSH server on an alternate port by running, eg.

   ```
   sh# /usr/sbin/sshd -p 2000 &
   ```

   You may then choose to log in to this server and continue the rest of these steps from that shell. We recommend doing this since some sshd shutdown scripts do particularly nasty things like killing *all* of the running SSH servers on a system, not just the parent server that may be listening on port 22. Roughly translated, this step is about guaranteeing that an alternate method of access is available should the main SSH server be shutdown and your connection via that server be terminated.

3. Locate your server's startup/shutdown script directory. On some systems this directory may be located at /etc/rc.d/init.d, but since this location is not constant across operating systems, for the purposes of this document we will refer to this directory as INITDIR. Consult your operating system's documentation for your system's location.

4. Run the following command.

   ```
   sh# mv $INITDIR/sshd $INITDIR/sshd.bak
   ```

5. Either copy or link the new sshd script to your system's startup/shutdown script directory.

```
sh# cp $GLOBUS_LOCATION/sbin/SXXsshd $INITDIR/sshd
```

6. Shutdown the currently running main SSH server.

```
sh# $INITDIR/sshd.bak stop
```

7. Provided you still have a connection to the machine, start the new SSH server.

```
sh# $INITDIR/sshd start
```

8. Test the new server by connecting to the standard SSH port (22) and authenticating via multiple methods. Especially test that GSI authentication works correctly.

9. If you are performing a new install, or if the old server was not configured to be started at run-time and shutdown automatically at system halt or reboot, either use a system utility such as RedHat's chkconfig to configure the system for the correct run-levels, or manually link up the correct run-levels.

```
sh# /sbin/chkconfig sshd reset
```

   The recommended run-levels are listed in a set of comments within the SXXsshd startup script. For example, on standard Unix systems we recommend running the GSI-Enabled OpenSSH server in run-levels two, three, four, and five.

10. Finally, if, as a precautionary measure, you started a SSH server on an alternate port in order to complete the install process, you can now safely stop all instances of that server.

5. Testing

1. Edit the file $GLOBUS_LOCATION/sbin/SXXsshd so that the GSI-Enabled OpenSSH server starts up on an alternate port.

2. Run the command

```
sh# $GLOBUS_LOCATION/sbin/SXXsshd start
```

   and verify that the server is running by checking that it both shows up in a process listing and creates a file named $GLOBUS_LOCATION/var/sshd.pid.

3. From a remote machine attempt to connect to the local server on the modified test port using the standard SSH authentication methods plus authenticating via your GSI credentials. This may require you to authorize these users via an appropriate entry in the grid-mapfile.

4. Stop the SSH server by running the command

```
sh# $GLOBUS_LOCATION/sbin/SXXsshd stop
```

and reverse any changes you made that altered the port on which the server resided upon startup. After this step, running SXXsshd start should start the server on the default port (22).

6. Security Considerations

GSI-OpenSSH is a modified version of OpenSSH[5] and includes full OpenSSH functionality. For more information on OpenSSH security, see the OpenSSH Security[6] page.

7. Troubleshooting

GSI authentication is very sensitive to clock skew. You must run a system clock synchronization service of some type on your system to prevent authentication problems caused by incorrect system clocks. We recommend NTP[7]. Please refer to your operating system documentation or the NTP Home Page[8] for installation instructions. Please also ensure your system timezone is set correctly.

[5] http://www.openssh.org/
[6] http://www.openssh.org/security.html
[7] http://www.ntp.org/
[8] http://www.ntp.org/

Chapter 8. GT 4.0 GSI-OpenSSH: User's Guide

1. Introduction

This is a guide for using the GSI-enabled OpenSSH client. It assumes that you (or your system administrator) have already installed the GSI OpenSSH and that you have also acquired a *user certificate* from an appropriate *Certificate Authority*.

2. Usage scenarios

2.1. Creating a proxy

First, set the GLOBUS_LOCATION environment variable to the location of your GSI-enabled OpenSSH installation. It may already be set for you by your system administrator.

Then, create a *proxy credential* for GSI authentication by running the **grid-proxy-init** program. This is your single sign-on to the Grid. By default, **grid-proxy-init** will create a proxy credential good for 12 hours.

To create a proxy credential with a different lifetime, use the **-hours** option.

For example:

```
% grid-proxy-init -hours 8
```

2.2. Deleting a proxy

To delete a proxy that was previously create with grid-proxy-init, run:

```
% grid-proxy-destroy
```

2.3. Getting authorized to connect to a site

Before you can connect to a site, the site needs to know the identity in your certificate so that it can map that identity to your local account. At a minimum, the site will need to know your subject name from your certificate. You can get your subject name by running **grid-cert-info** with the **-subject** argument. For example:

```
% grid-cert-info -subject
```

Email your subject name to the administrator of the system you wish to connect to so that they can add your entry to the appropriate authorization files.

Once you have your proxy credential, all you should have to do is run gsissh, providing it with the hostname of the host you want to connect to. For example:

```
% gsissh myhost.somedomain.edu
```

You should then find yourself automatically logged into your account on the remote system. If something goes wrong, please see Section 5, “Troubleshooting” for assistance.

3. Command line tools

Please see the GSI-OpenSSH Command Reference.

4. Graphical user interfaces

GSI-enabled OpenSSH does not provide a GUI.

5. Troubleshooting

Some common errors are listed below. If you need additional assistance, please run gsissh with the '-vvv' argument (specifying verbose output) and send the output to your system administrator for assistance.

5.1. GSS-API error Failure acquiring GSSAPI credentials: GSS_S_CREDENTIALS_EXPIRED

This means that your *proxy certificate* has expired. You need to run **grid-proxy-init** to acquire a new proxy certificate, then run gsissh again.

5.2. The gsissh command prompts you for a pass phrase when you run it

This could mean that you don't have a proxy certificate; try running **grid-proxy-init** and then running gsissh again. It could also mean that the GSI authentication is failing for some reason and gsissh is falling back to a different authentication mechanism. Reasons that it might fail include:

- The host you are connecting to does not have a GSI-enabled OpenSSH server.
- You are not authorized to use GSI authentication to the host. Contact the administrator.

Chapter 9. GT 4.0 GSI-OpenSSH: Developer's Guide

1. Introduction

This document provides information for GSI-OpenSSH developers.

The changes to OpenSSH[1] to add GSI support are limited, because we build on the existing GSSAPI support in OpenSSH for Kerberos. In addition to adding support for the GSI GSSAPI mechanism, GSI-OpenSSH includes support for GSSAPI key exchange, as specified in draft-ietf-secsh-gsskeyex-08.txt[2], whereas OpenSSH only supports GSSAPI authentication. Visit the GSI OpenSSH Patch Page[3] for the patch containing the differences between OpenSSH and GSI-OpenSSH.

2. Before you begin

2.1. Feature summary

Features new in GT 4.0

- This is the first Globus Toolkit release that includes GSI-enabled OpenSSH.

Other Supported Features

- The **gsissh** command provides a secure remote login service with forwarding of X.509 *proxy credentials*.
- The **gsiscp** and **gsisftp** commands provide a secure file transfer service authenticated with X.509 proxy credentials, mimicking the **rcp/scp** and **ftp/sftp** commands.
- All standard OpenSSH features are supported, excluding Kerberos authentication. Kerberos authentication is *not* compatible with GSI-enabled OpenSSH.
- The GSI-OpenSSH server can replace the standard system SSH server in typical environments.
- If no username is given on the command-line, GSI-OpenSSH automatically determines the username that corresponds to the X.509 proxy *certificate subject* in the server's `grid-mapfile`.

Deprecated Features

- None

2.2. Tested platforms

Tested Platforms for GSI-OpenSSH

- Mac OS X 10.3

[1] http://www.openssh.org/
[2] http://www.watersprings.org/pub/id/draft-ietf-secsh-gsskeyex-08.txt
[3] http://grid.ncsa.uiuc.edu/ssh/installpatch.html

- i686 GNU/Linux
- ia64 GNU/Linux

2.3. Backward compatibility summary

Protocol changes since GT 3.2

- GSI-enabled OpenSSH was not included in GT 3.2.

API changes since GT 3.2

- GSI-enabled OpenSSH 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

GSI-enabled OpenSSH depends on the following GT components:

- Pre-WS Authentication and Authorization

GSI-enabled OpenSSH depends on the following 3rd party software:

- OpenSSH[4]

2.5. Security considerations

GSI-OpenSSH is a modified version of OpenSSH[5] and includes full OpenSSH functionality. For more information on OpenSSH security, see the OpenSSH Security[6] page.

3. Architecture and design overview

For information about the SSH protocol, including the latest draft of the SSH GSSAPI protocol specification, see the current documents of the IETF Secure Shell (secsh) Working Group[7]. For information on GSSAPI, see RFC 2743[8] and RFC 2744[9].

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 Chapter 11, *GT 4.0 Component Guide to Public Interfaces: GSI-OpenSSH*.

[4] http://www.openssh.org/
[5] http://www.openssh.org/
[6] http://www.openssh.org/security.html
[7] http://www.ietf.org/html.charters/secsh-charter.html
[8] http://www.ietf.org/rfc/rfc2743.txt
[9] http://www.ietf.org/rfc/rfc2744.txt

5. Usage scenarios

The GSI-OpenSSH interface is through command-line tools only.

6. Tutorials

There are no tutorials available at this time

7. Debugging

Pass the '-vvv' flag to the GSI-OpenSSH clients when debugging to increase the verbosity of the output. For example:

```
$ gsissh -vvv <remote host>
```

Likewise, pass the following flags to the server when debugging:

```
$ sshd -ddd -o 'UsePrivilegeSeparation no' -r
```

You can add the '-p <port number>' option to run the sshd on an alternate port for debugging without affecting your system sshd. Then, give the same '-p <port number>' option to gsissh to test the sshd.

The presence of a debugging flag also runs the server without detaching it from the console. The server will only handle one connection in this mode.

8. Troubleshooting

8.1. No proxy found

Failing to run grid-proxy-init to create a user proxy with which to connect will result in the client notifying you that no local credentials exist. Any attempt to authenticate using GSI will fail in this case.

```
debug1: Local version string SSH-2.0-OpenSSH_3.2.3p1
debug1: Problem with local credentials
debug1: no proxy credentials: run grid-proxy-init or wgpi first
        Function:proxy_pw_cb
debug1: SSH2_MSG_KEXINIT sent
debug1: SSH2_MSG_KEXINIT received
```

Fix: Verify that your GSI proxy has been properly initialized via 'grid-proxy-info'. If you need to initialize the proxy, use the command 'grid-proxy-init'.

8.2. Host key permissions failure

The host key that the SSH server is using for GSI authentication must only be readable by the user which owns it. Any other permissions will cause the following debugging output to be generated.

```
7959: debug1: Local version string SSH-1.99-OpenSSH_3.4p1
```

```
7959: debug1: list_hostkey_types: ssh-rsa,ssh-dss
7959: debug1: Problem with local credentials
7959: debug1: bad file system permissions on private key
              key must only be readable by the user
              File=/etc/grid-security/hostkey.pem
              Function:proxy_init_cred
7959: debug1: SSH2_MSG_KEXINIT sent
7959: debug1: SSH2_MSG_KEXINIT received
```

Fix: Make sure that the host key's UNIX permissions are mode 400 (that is, it should only have mode readable for the user that owns the file, and no other mode bits should be set).

8.3. Unable to map implicit username

If the server was passed an "implicit username" (i.e. requested to map the incoming connection to a username based on some contextual clues such as the certificate's subject), and no entry exists in the grid-mapfile for the incoming connection's *certificate subject*, the server should output a clue that states it is unable to set the username against which to authenticate.

```
7978: debug2: input_userauth_request: try method none
7978: Failed none for cphillip from 141.142.21.10 port 1240 ssh2
7978: debug1: gssapi received empty username
7978: debug1: failed to set username from gssapi context
7978: Failed external-keyx for cphillip from 141.142.21.10 port 1240 ssh2
7978: debug1: gssapi received empty username
7978: debug1: failed to set username from gssapi context
7978: Failed gssapi for cphillip from 141.142.21.10 port 1240 ssh2
7978: debug1: userauth-request for user cphillip service ssh-connection method publickey
7978: debug1: attempt 0 failures 3
```

Fix: Add an entry for the user to the grid-mapfile.

8.4. Entry in grid-mapfile refers to non-existent username

If the subject name given in the system's grid-mapfile points to a non-existent user, the server will give an internal error which is best caught when it is running in debugging mode.

```
8046: debug2: input_userauth_request: setting up authctxt for cphillip
8046: debug2: input_userauth_request: try method none
8046: Failed none for cphillip from 141.142.21.10 port 1259 ssh2
8046: debug1: gssapi received empty username
8046: debug1: gssapi successfully set username to cphillip2
8046: debug1: userauth-request for user cphillip2 service ssh-connection method external-k
8046: debug1: attempt 0 failures 1
8046: input_userauth_request: illegal user cphillip2
8046: debug2: input_userauth_request: try method external-keyx
8046: GSI user /C=US/O=National Computational Science Alliance/CN=Chase Phillips is author
8046: INTERNAL ERROR: authenticated invalid user cphillip2
8046: debug1: Calling cleanup 0x806bb38(0x0)
```

Fix: Add a new account to the system matching the username pointed at by the user's subject in the grid-mapfile.

8.5. Client proxy uninitialized or non-GSI agent

Should the user attempt to connect without first creating a *proxy certificate*, or if the user is connecting via a SSH client that does not support GSI authentication, the server will note that no GSSAPI data was sent to it. Verify that the client is able to connect through another GSI service (such as the gatekeeper) to make sure that the user's proxy has been created correctly.

```
9597: debug2: input_userauth_request: setting up authctxt for cphillip
9597: debug2: input_userauth_request: try method none
9597: Failed none for cphillip from 141.142.21.10 port 2554 ssh2
9597: debug1: gssapi received empty username
9597: debug1: No suitable client data
9597: debug1: failed to set username from gssapi context
9597: Failed external-keyx for cphillip from 141.142.21.10 port 2554 ssh2
9597: debug1: gssapi received empty username
9597: debug1: No suitable client data
9597: debug1: failed to set username from gssapi context
9597: Failed gssapi for cphillip from 141.142.21.10 port 2554 ssh2
9597: debug1: userauth-request for user cphillip service ssh-connection method publickey
9597: debug1: attempt 0 failures 3
9597: debug2: input_userauth_request: try method publickey
```

Fix: Verify that you are using a GSI-enabled SSH client and that your GSI proxy has been properly initialized via 'grid-proxy-info'. If you need to initialize this proxy, use the command 'grid-proxy-init'.

9. Related Documentation

Please see the GSI-OpenSSH Home Page[10] at NCSA for more information.

[10] http://grid.ncsa.uiuc.edu/ssh/

Chapter 10. GT 4.0 Component Fact Sheet: Utilities - GSI-OpenSSH

1. Brief overview

GSI-OpenSSH is a modified version of OpenSSH that adds support for X.509 *proxy certificate* authentication and delegation, providing a single sign-on remote login and file transfer service. GSI-OpenSSH can be used to login to remote systems and transfer files between systems without entering a password, relying instead on a valid *proxy credential* for authentication. GSI-OpenSSH forwards proxy credentials to the remote system on login, so commands requiring proxy credentials (including GSI-OpenSSH commands) can be used on the remote system without the need to manually create a new proxy credential on that system.

2. Summary of features

Features new in GT 4.0

- This is the first Globus Toolkit release that includes GSI-enabled OpenSSH.

Other Supported Features

- The **gsissh** command provides a secure remote login service with forwarding of X.509 *proxy credentials*.
- The **gsiscp** and **gsisftp** commands provide a secure file transfer service authenticated with X.509 proxy credentials, mimicking the **rcp/scp** and **ftp/sftp** commands.
- All standard OpenSSH features are supported, excluding Kerberos authentication. Kerberos authentication is *not* compatible with GSI-enabled OpenSSH.
- The GSI-OpenSSH server can replace the standard system SSH server in typical environments.
- If no username is given on the command-line, GSI-OpenSSH automatically determines the username that corresponds to the X.509 proxy *certificate subject* in the server's `grid-mapfile`.

Deprecated Features

- None

3. Usability summary

Usability improvements for GSI-OpenSSH:

- This is the first Globus Toolkit release to include GSI-OpenSSH.

4. Backward compatibility summary

Protocol changes since GT 3.2

- GSI-enabled OpenSSH was not included in GT 3.2.

API changes since GT 3.2

- GSI-enabled OpenSSH 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

GSI-enabled OpenSSH depends on the following GT components:

- Pre-WS Authentication and Authorization

GSI-enabled OpenSSH depends on the following 3rd party software:

- OpenSSH[1]

6. Tested platforms

Tested Platforms for GSI-OpenSSH

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

7. Associated standards

Associated standards for GSI-OpenSSH:

- The latest draft of the SSH GSSAPI protocol specification is available from the IETF Secure Shell (secsh) Working Group[2]
- RFC 2743[3] GSSAPI
- RFC 2744[4] GSSAPI: C-bindings

8. For More Information

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

[1] http://www.openssh.org/
[2] http://www.ietf.org/html.charters/secsh-charter.html
[3] http://www.ietf.org/rfc/rfc2743.txt
[4] http://www.ietf.org/rfc/rfc2744.txt
[5] index.html

Chapter 11. GT 4.0 Component Guide to Public Interfaces: GSI-OpenSSH

1. Semantics and syntax of APIs

GSI-enabled OpenSSH does not provide an API.

2. Semantics and syntax of the WSDL

GSI-enabled OpenSSH does not have a WSDL interface.

3. Command line tools

Please see the GSI-OpenSSH Command Reference.

4. Overview of Graphical User Interface

GSI-enabled OpenSSH does not provide a GUI.

5. Semantics and syntax of domain-specific interface

GSI-enabled OpenSSH does not provide any domain-specific interfaces.

6. Configuration interface

The GSI-enabled OpenSSH software is installed with a default set of configuration files, described below. You may want to modify the `ssh_config` file before using the clients and the `sshd_config` file before using the server.

If the GSI-enabled OpenSSH install script finds existing SSH key pairs, it will create symbolic links to them rather than generating new key pairs. The SSH key pairs are not required for GSI authentication. However, if you wish to support other SSH authentication methods, make sure the sshd (running as root) can read the key pair files (i.e., beware of NFS mounts with root_squash). If running multiple sshds on a system, we recommend configuring them so they all use the same key pairs (i.e., use symbolic links) to avoid client-side confusion.

- `$GLOBUS_LOCATION/etc/ssh/moduli`

  moduli is a crypto parameter for generating keys.

- `$GLOBUS_LOCATION/etc/ssh/ssh_config`

  ssh_config contains options that are read by ssh, scp, and sftp at run-time. The installed version is the default provided by OpenSSH, with X11Forwarding enabled. You may need to customize this file for compatibility with your system SSH installation (i.e., compare it with /etc/ssh/ssh_config).

- `$GLOBUS_LOCATION/etc/ssh/ssh_host_key[.pub]`

  Your system's RSA public-/private-key pair for SSH protocol 1 communications.

- `$GLOBUS_LOCATION/etc/ssh/ssh_host_dsa[.pub]`

  Your system's DSA public-/private-key pair for SSH protocol 2 communications.

- `$GLOBUS_LOCATION/etc/ssh/ssh_host_rsa[.pub]`

  Your system's RSA public-/private-key pair for SSH protocol 2 communications.

- `$GLOBUS_LOCATION/etc/ssh/ssh_prng_cmds`

  ssh_prng_cmds contains paths to a number of files that ssh-keygen may need to use if your system does not have a built-in entropy pool (like /dev/random).

- `$GLOBUS_LOCATION/etc/ssh/sshd_config`

  sshd_config contains options that are read by sshd when it starts up. The installed version is the default provided by OpenSSH, with X11Forwarding enabled. You may need to customize this file for compatibility with your system SSH installation (i.e., compare it with /etc/ssh/sshd_config). For example, to enable PAM authentication, you will need to set "UsePAM yes" in this file.

7. Environment variable interface

The GSI-enabled OpenSSHD needs to be able to find certain files and directories in order to properly function.

The items that OpenSSHD needs to be able to locate, their default location and the environment variable to override the default location are:

- *Host key*

  Default location: /etc/grid-security/hostkey.pem

  Override with X509_USER_KEY environment variable

- *Host certificate*

  Default location: /etc/grid-security/hostcert.pem

  Override with X509_USER_CERT environment variable

- *Grid map file*

  Default location: /etc/grid-security/grid-mapfile

  Override with GRIDMAP environment variable

- *Certificate directory*

  Default location: /etc/grid-security/certificates

  Override with X509_CERT_DIR environment variable

Chapter 12. GT 4.0 GSI-OpenSSH: Quality Profile

1. Test coverage reports

Not yet available.

2. Code analysis reports

Not yet available.

3. Outstanding bugs

None.

4. Bug Fixes

This is the first release of the Globus Toolkit that includes GSI-enabled OpenSSH.

5. Performance reports

None.

Chapter 13. GT 4.0 Migrating Guide for GSI-OpenSSH

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

1. Migrating from GT2

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

2. Migrating from GT3

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

GT 4.0 GSI-OpenSSH Command Line Reference

The gsissh(1), gsiscp(1), and gsisftp(1) commands provide the same interfaces as the standard OpenSSH *ssh*, *scp*, and *sftp* commands, respectively, with the added ability to perform X.509 *proxy credential* authentication and delegation.

Name

gsissh -- Secure remote login

`gsissh`

Tool description

Use the *gsissh* command to securely login to a remote machine.

Command syntax

gsissh [-l login_name] hostname | user@hostname [command]

Name

gsiscp -- Secure remote file copy

`gsiscp`

Tool description

Use the *gsiscp* command to securely copy files to or from a remote machine.

Command syntax

gsiscp [-P port] [[user@]host1:]file1 [...] [[user@]host2:]destfile

Name

gsisftp -- Secure file transfer

`gsisftp`

Tool description

The *gsisftp* command provides an interactive interface for transferring files to and from remote machines.

Command syntax

gsisftp [[user@]host[:dir[/]]]

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