GT 6 Quickstart

Introduction

This is a quickstart that shows a full installation of the Toolkit on two Fedora Linux machines, named `elephant` and `donkey`. It shows the installation of prereqs, installation of the toolkit, creation of *certificates*, and configuration of services. It is designed to supplement the main admin guide, Installing GT 6.0.

Scenarios are shown for running *GridFTP* and *GRAM5* services, and using GridFTP and GRAM clients.

Table of Contents

1. Typographical Conventions

Where there is a command to be typed, it will be preceded by one of the following prompts:

`elephant#` , `donkey#`	Run this command as the `root` super-user, on the `elephant` or `donkey` hosts respectively. You might have to use a command like **su**(8) or **sudo**(8) to start a root shell before executing the command.
`myproxy@elephant%`	Run this command as the `myproxy` user, on the `elephant` host. This user is created automatically when the myproxy-server package is installed.
`quser@elephant%` , `quser@donkey%`	Run this command as the normal user account you are intending to interact with your Globus sevices, on the `elephant` or `donkey` hosts. In this document, we use the `quser` accout for this, but if you have another user, you can use it for that purpose.

Commands themselves will be typeset as **run-this-command -with-arguments**, and responses to the commands like this `Some Response Text`. If there is some portion of a command which should be replaced by value, such as a version number, it will be typeset like this: *`REPLACEME`*.

Finally, in some cases you will be prompted for a passphrase. When that occurs, the entry of the passphrase will be indicated by **`******`**, even though nothing will be printed to the screen.

2. Prerequisites

We distribute the Globus Toolkit 6 as a set of RPM and Debian packages for Linux systems, as an installable package for Mac OS X, as a .zip file for Windows and Cygwin, as well as a source installer which can be used on other operating systems. In this quickstart, we will be installing RPM packages. Thus, it is a prerequisite for following this quickstart that you are running a distribution for which we provide RPMs. If you are running a supported Debian or Ubuntu system, the process is very similar, but you'll need to use the **apt-get** or similar tools to install the packages. For the source installer, there is more work involved, and you'll need to consult the full installation guide.

First, we will to set up our system to use the Globus package repository. This repository contains the Globus software packages, signed by our build manager. We provide RPM and Debian packages that contain a source configuration file and the public key which can be used to verify the packages. If your distribution has Globus 6.0 packages within its repository, you can skip to the next section.

The globus toolkit package repo RPM can be downloaded from the repo RPM package on globus.org[1].

To install binary RPMs, download the globus-toolkit-repo package from the link above and install it with the command:

```
elephant# rpm -hUv globus-toolkit-repo-6-1.noarch.rpm
```

The globus toolkit package repo Debian file can be downloaded from the repo Debian package on globus.org[2].

To install Debian or Ubuntu package, download the globus-toolkit-repo package from the link above and install it with the command:

```
elephant# dpkg -i globus-toolkit-repo_6-2_all.deb
```

Once you've installed the Globus repository package, you can use your operating system's packaging tools: **yum** or **apt-get**, to install the Globus components.

Important

For operating systems based on RHEL (such as Red Hat Enterprise Linux, CentOS, and Scientific Linux), the compatible EPEL repository must be enabled before installing myproxy. For OS versions 5.x, install the EPEL 5 package[3], and for OS version 6.x, use 6 package[4].

For information about installing these, see the EPEL FAQ[5].

This step is not needed for Fedora, Debian, or Ubuntu systems.

3. Setting up the first machine (GridFTP, GRAM, and MyProxy services)

3.1. Installing the Toolkit

Install packages:

```
elephant# yum install globus-gridftp globus-gram5 globus-gsi myproxy \
    myproxy-server myproxy-admin
```

This will install the GridFTP, GRAM, and *MyProxy* services, as well as set up a basic *SimpleCA* so that you can issue security credentials for users to run the Globus services.

Note

For Debian and Ubuntu systems, use **apt-get** or **aptitude** or another package manager to install the same packages as in the **yum** command above.

[1] http://www.globus.org/ftppub/gt6/installers/repo/globus-toolkit-repo-6-1.noarch.rpm
[2] http://www.globus.org/ftppub/gt6/installers/repo/globus-toolkit-repo_6-2_all.deb

3.2. Setting up security on your first machine

The Globus Toolkit uses X.509 certificates and *proxy certificates* to authenticate and authorize grid users. For this quickstart, we use the Globus SimpleCA tools to manage our own *Certificate Authority*, so that we don't need to rely on any external entitty to authorize our grid users.

Note

In many deployment scenarios, certificates for both services and users are obtained through one or more third party CAs. In such scenarios, it is unnecessary to use SimpleCA or MyProxy to issue certificates. Since this quickstart is intended to describe a simple, standalone deployment scenario, we describe how to use these tools to issue your own certificates.

When the globus-simple-ca package is installed, it will automatically create a new Certificate Authority and deploy its public certificate into the globus trusted certificate directory. It will also create a host certificate and key, so that the Globus services will be able to run.

We'll also need to copy the host certificate and key into place so that the myproxy service can use it as well.

```
elephant# install -o myproxy -m 644 \
    /etc/grid-security/hostcert.pem \
    /etc/grid-security/myproxy/hostcert.pem
elephant# install -o myproxy -m 600 \
    /etc/grid-security/hostkey.pem \
    /etc/grid-security/myproxy/hostkey.pem
```

3.3. Creating a MyProxy Server

We are going to create a MyProxy server on elephant, following the instructions at http://grid.ncsa.illinois.edu/myproxy/fromscratch.html#server. This will be used to store our user's certificates. In order to enable myproxy to use the SimpleCA, modify the `/etc/myproxy-server.config` file, by uncommenting every line in the section `Complete Sample Policy #1` such that section looks like this myproxy configuration[6]:

```
#
# Complete Sample Policy #1 - Credential Repository
#
# The following lines define a sample policy that enables all
# myproxy-server credential repository 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"
cert_dir /etc/grid-security/certificates
```

We'll next add the myproxy user to the simpleca group so that the myproxy server can create certificates.

```
elephant# usermod -a -G simpleca myproxy
```

[6] myproxy-server.config

Start the myproxy server:

```
elephant# service myproxy-server start
Starting myproxy-server (via systemctl):                   [  OK  ]
```

Note

For Debian and Ubuntu systems, use the **invoke-rc.d** command in place of **service**.

Check that it is running:

```
elephant# service myproxy-server status
myproxy-server.service - LSB: Startup the MyProxy server daemon
   Loaded: loaded (/etc/rc.d/init.d/myproxy-server)
   Active: active (running) since Fri, 02 Nov 2012 09:07:51 -0400; 1min 20s ago
  Process: 1205 ExecStart=/etc/rc.d/init.d/myproxy-server start (code=exited, status=0/SUC
   CGroup: name=systemd:/system/myproxy-server.service
    # 1214 /usr/sbin/myproxy-server -s /var/lib/myproxy

Nov 02 09:07:51 elephant.globus.org runuser[1210]: pam_unix(runuser:session):...
Nov 02 09:07:51 elephant.globus.org myproxy-server[1212]: myproxy-server v5.9...
Nov 02 09:07:51 elephant.globus.org myproxy-server[1212]: reading configurati...
Nov 02 09:07:51 elephant.globus.org myproxy-server[1212]: usage_stats: initia...
Nov 02 09:07:51 elephant.globus.org myproxy-server[1212]: Socket bound to 0.0...
Nov 02 09:07:51 elephant.globus.org myproxy-server[1212]: Starting myproxy-se...
Nov 02 09:07:51 elephant.globus.org runuser[1210]: pam_unix(runuser:session):...
Nov 02 09:07:51 elephant.globus.org myproxy-server[1205]: Starting myproxy-se...
```

The important thing to see in the above is that the process is in the `active (running)` state.

Note

For other Linux distributions which are not using systemd, the output will be different. You should still see some information indicating the service is running.

As a final sanity check, we'll make sure the myproxy TCP port `7512` is in use via the netstat command:

```
elephant# netstat -an | grep 7512
tcp        0      0 0.0.0.0:7512            0.0.0.0:*               LISTEN
```

3.3.1. User Credentials

We'll need to specify a full name and a login name for the user we'll create credentials for. We'll be using the `QuickStart User` as the user's name and `quser` as user's account name. You can use this as well if you first create a quser unix account. Otherwise, you can use another local user account. Run the **myproxy-admin-adduser** command as the `myproxy` user to create the credentials. You'll be prompted for a passphrase, which must be at least 6 characters long, to encrypt the *private key* for the user. You must communicate this passphrase to the user who will be accessing this credential. He can use the **myproxy-change-passphrase** command to change the passphrase.

The command to create the myproxy credential for the user is

```
elephant# su - -s /bin/sh myproxy
myproxy@elephant% PATH=$PATH:/usr/sbin
myproxy@elephant% myproxy-admin-adduser -c "QuickStart User" -l quser
```

```
Legacy library getopts.pl will be removed from the Perl core distribution in the next majo
Enter PEM pass phrase: ******
Verifying - Enter PEM pass phrase:******

The new signed certificate is at: /var/lib/globus/simple_ca/newcerts/02.pem

using storage directory /var/lib/myproxy
Credential stored successfully
Certificate subject is:
/O=Grid/OU=GlobusTest/OU=simpleCA-elephant.globus.org/OU=local/CN=QuickStart User
```

3.3.2. User Authorization

Finally, we'll create a *grid map file* entry for this credential, so that the holder of that credential can use it to access globus services. We'll use the **grid-mapfile-add-entry** program for this. We need to use the exact string from the output above as the parameter to the `-dn` command-line option, and the local account name of user to authorize as the parameter to the `-ln` command-line option.

```
elephant# grid-mapfile-add-entry -dn \
    "/O=Grid/OU=GlobusTest/OU=simpleCA-elephant.globus.org/OU=local/CN=QuickStart User" \
    -ln quser
Modifying /etc/grid-security/grid-mapfile ...
/etc/grid-security/grid-mapfile does not exist... Attempting to create /etc/grid-security/
New entry:
"/O=Grid/OU=GlobusTest/OU=simpleCA-elephant.globus.org/OU=local/CN=QuickStart User" quser
(1) entry added
```

3.4. Setting up GridFTP

Now that we have our host and user credentials in place, we can start a globus service. This set up comes from the GridFTP Admin Guide.

Start the GridFTP server:

```
elephant# service globus-gridftp-server start
Started GridFTP Server                                     [  OK  ]
```

Check that the GridFTP server is running and listening on the gridftp port:

```
elephant# service globus-gridftp-server status
GridFTP Server Running (pid=20087)
elephant# netstat -an | grep 2811
tcp        0      0 0.0.0.0:2811            0.0.0.0:*               LISTEN
```

Now the GridFTP server is waiting for a request, so we'll generate a proxy from the myproxy service by using **myproxy-logon** and then copy a file from the GridFTP server with the **globus-url-copy** command. We'll use the passphrase used to create the myproxy credential for `quser`.

```
quser@elephant% myproxy-logon -s elephant
Enter MyProxy pass phrase: ******
A credential has been received for user quser in /tmp/x509up_u1001
quser@elephant% globus-url-copy gsiftp://elephant.globus.org/etc/group \
    file:///tmp/quser.test.copy
quser@elephant% diff /tmp/quser.test.copy /etc/group
```

At this point, we've configured the myproxy and GridFTP services and verified that we can create a security credential and transfer a file. If you had trouble, check the security troubleshooting section in the Security Admin Guide. Now we can move on to setting up GRAM5 resource management.

3.5. Setting up GRAM5

Now that we have security and GridFTP set up, we can set up GRAM for resource management. There are several different Local Resource Managers (LRMs) that one could configure GRAM to use, but this guide will explain the simple case of setting up a "fork" jobmanager, without auditing. For details on all other configuration options, and for reference, you can see the GRAM5 Admin Guide. The GRAM service will use the same host credential as the GridFTP service, and is configured by default to use the fork manager, so all we need to do now is start the service.

Start the GRAM gatekeeper:

```
elephant# service globus-gatekeeper start
Started globus-gatekeeper                                  [  OK  ]
```

We can now verify that the service is running and listening on the GRAM5 port:

```
elephant# service globus-gatekeeper status
globus-gatekeeper is running (pid=20199)
elephant# netstat -an | grep 2119
tcp6       0      0 :::2119                 :::*                    LISTEN
```

The gatekeeper is set up to run, and is ready to authorize job submissions and pass them on to the fork job manager. We can now run a couple of test jobs:

```
quser@elephant% myproxy-logon -s elephant
Enter MyProxy pass phrase: ******
A credential has been received for user quser in /tmp/x509up_u1001.
quser@elephant% globus-job-run elephant /bin/hostname
elephant.globus.org
quser@elephant% globus-job-run elephant /usr/bin/whoami
quser
```

If you had trouble, check the security troubleshooting section in the Security Admin Guide. To learn more about using GRAM 5, take a look at the GRAM User's Guide.

4. Setting up your second machine

Alas, it's not much of a grid with just one machine. So let's start up on another machine and add it to this little test grid.

4.1. Setting up your second machine: Prereqs

See Prereqs.

4.2. Setting up your second machine: Installation

Install packages as before:

```
donkey# yum install globus-gridftp myproxy globus-gram5
```

4.3. Setting up your second machine: Security

Now let's get security set up on the second machine. We're going to trust the original simpleCA to this new machine; there's no need to create a new one. First, we'll bootstrap trust of the SimpleCA running on elephant:

```
donkey# myproxy-get-trustroots -b -s elephant
Bootstrapping MyProxy server root of trust.
New trusted MyProxy server: /O=Grid/OU=GlobusTest/OU=simpleCA-elephant.globus.org/CN=host/
New trusted CA (e3d1c34d.0): /O=Grid/OU=GlobusTest/OU=simpleCA-elephant.globus.org/CN=Glob
Trust roots have been installed in /etc/grid-security/certificates/.
```

This allows clients and services on `donkey` to trust certificates which are signed by the CA on `elephant` machine. If we weren't going to run any Globus services on `donkey`, then we could stop here. Users on `donkey` could acquire credentials using the **myproxy-logon** command and perform file transfers and execute jobs using the **globus-url-copy** and **globus-job-run** commands. However, we'll continue to configure the GridFTP and GRAM5 services on `donkey` as well.

We're going to create the host certificate for donkey, but we create it on elephant, so that we don't have to copy the certificate request between machines. The **myproxy-admin-addservice** command will prompt for a passphrase for this credential. We will use this passphrase to retrieve the credential on donkey.

```
myproxy@elephant% myproxy-admin-addservice -c "donkey.globus.org" -l donkey
Legacy library getopts.pl will be removed from the Perl core distribution in the next majo
Enter PEM pass phrase:******
Verifying - Enter PEM pass phrase:******

The new signed certificate is at: /var/lib/globus/simple_ca/newcerts/03.pem

using storage directory /var/lib/myproxy
Credential stored successfully
Certificate subject is:
/O=Grid/OU=GlobusTest/OU=simpleCA-elephant.globus.org/OU=local/CN=donkey.globus.org
```

Next we'll retrieve the credential on donkey as the root user.

```
donkey# myproxy-retrieve -s elephant -k donkey.globus.org -l donkey
Enter MyProxy pass phrase: ******
Credentials for quser have been stored in
/etc/grid-security/hostcert.pem and
/etc/grid-security/hostkey.pem.
```

At this point, we no longer need to have `donkey`'s host certificate on `elephant`'s myproxy server, so we'll delete it.

```
donkey# myproxy-destroy -s elephant -k donkey.globus.org -l donkey
MyProxy credential 'donkey.globus.org' for user donkey was successfully removed.
```

And as a final setup, we'll add quser's credential to the grid-mapfile on `donkey`, so that the `quser` account can access services there as well.

```
donkey# grid-mapfile-add-entry -dn \
    "/O=Grid/OU=GlobusTest/OU=simpleCA-elephant.globus.org/OU=local/CN=QuickStart User" \
    -ln quser
Modifying /etc/grid-security/grid-mapfile ...
/etc/grid-security/grid-mapfile does not exist... Attempting to create /etc/grid-security/
```

```
New entry:
"/O=Grid/OU=GlobusTest/OU=simpleCA-elephant.globus.org/OU=local/CN=QuickStart User" quser
(1) entry added
```

At this point, we have set up security on donkey to trust the CA on elephant. We have created a host certificate for donkey so that we can run Globus services on donkey, and we have enabled the quser account to use services on donkey. The last thing to do is to turn on the Globus services on donkey.

4.4. Setting up your second machine: GridFTP

GridFTP set up on the second machine is identical to the first. I'll just list the commands here; see Section 3.4, "Setting up GridFTP" for additional information.

```
donkey# service globus-gridftp-server start
Started GridFTP Server                                     [  OK  ]
```

Now we can test it.

First, we'll retrive a proxy credential from the myproxy server so that the user on donkey can interact with the Globus services. Here we'll use the same passphrase as we used to create the `quser` credential.

```
quser@donkey% myproxy-logon -s elephant
Enter MyProxy pass phrase: ******
A credential has been received for user quser in /tmp/x509up_u1001.
```

Next we'll transfer a file between the gridftp servers on donkey and elephant:

```
quser@donkey% globus-url-copy gsiftp://elephant.globus.org/etc/group \
   gsiftp://donkey.globus.org/tmp/from-elephant
```

That was a slightly more complicated test than we ran on elephant earlier. In this case, we did a third-party transfer between two GridFTP servers. It worked, so I have the local and remote security configured correctly.

If you run into problems, perhaps you have a firewall between the two machines? GridFTP needs to communicate on data ports, not just port 2811. The error for this condition looks like:

```
error: globus_ftp_client: the server responded with an error
500 500-Command failed. : callback failed.
500-globus_xio: Unable to connect to 140.221.8.19:42777
500-globus_xio: System error in connect: No route to host
500-globus_xio: A system call failed: No route to host
500 End.
```

You can set up a range of ports to be open on the firewall and configure GridFTP to use them. See the GridFTP admin firewall doc.

4.5. Setting up your second machine: GRAM5

Now we can submit a staging job. This job will copy the **/bin/echo** program from donkey to a file called `/tmp/my_echo`. Then it runs it with some arguments, and captures the stderr/stdout. Finally, it will clean up the `my_echo` file when execution is done.

```
quser@donkey% globus-job-run elephant \
    -x '(file_stage_in=(gsiftp://donkey.globus.org/bin/echo /tmp/echo)) \
        (file_clean_up=/tmp/echo)' /bin/ls -l /tmp/echo
```

```
-rw-r--r-- 1 quser quser 27120 Nov  2 09:56 /tmp/echo
```

This example staged in a file, had an executable act on that file, and cleaned up the file afterward.

You can get other examples of GRAM files from GRAM usage scenarios.

5. Conclusion

Hopefully this guide has been helpful in familiarizing you with some of the administration tasks and tools to use the Globus Toolkit. If you've reached this point successfully, you should have enough knowledge to enable additional hosts to use your grid by repeating the tasks in Section 4, “Setting up your second machine”. Also, by repeating the tasks in Section 3.3.1, “User Credentials” and Section 3.3.2, “User Authorization” you can enable additional users to access your compute and data resources.

Glossary

C

Certificate Authority (CA)	An entity that issues certificates.
certificate	A public key plus information about the certificate owner bound together by the digital signature of a CA. In the case of a CA certificate, the certificate is self signed, i.e. it was signed using its own private key.

G

Grid Resource Allocation and Management (GRAM)	This component is used to locate, submit, monitor, and cancel jobs on Grid computing resources.
GridFTP	A file transfer protocol based on FTP with extensions for security and parallel data transfers.
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 section here.

M

MyProxy	Myproxy manages X.509 credentials (certificates and private keys). MyProxy combines an online credential repository with an online certificate authority to allow users to securely obtain credentials.

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 this.
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 place. GSI uses proxy certificates for single sign on and delegation of rights to other entities. For more information about types of proxy certificates and their compatibility in different versions of GT, see http://dev.globus.org/wiki/Security/ProxyCertTypes.

S

SimpleCA	SimpleCA is a tool for creating and managing a CA. It provides a way to implement a X.509 trust root and sign certificates for users and hosts.

Installing GT 6.0

Installing GT 6.0

Introduction

This guide is the starting point for everyone who wants to install Globus Toolkit 6.0. It will take you through a basic installation that installs the following basic services: a security infrastructure (GSI), GridFTP, and Execution Services (GRAM5).

This guide is also available as a PDF[1]. However, each component includes online reference material, which this guide sometimes links to.

[1] installingGT.pdf

Table of Contents

List of Tables

Chapter 1. Before you begin

Before you start installing the Globus Toolkit 6.0, there are a few things you should consider. The toolkit contains several components, and you may only be interested in some of them.

The Globus Toolkit version 6.0 includes:

- GSI: security
- GridFTP: file transfer
- GRAM: job execution/resource management
- MyProxy: credential repository/certificate authority
- GSI-OpenSSH: GSI secure single sign-on remote shell

Important

These all run on Unix platforms only.

If you are new to the toolkit and want to experiment with the components, you may want to use a supported RedHat based or Debian based Linux system. With the new supported native packaging installs, they are the simplest platforms on which to install GT services.

For the purposes of this documentation, Globus is being installed on a machine called `elephant`.

1. Typographical Conventions

Where there is a command to be typed, it will be preceded by one of the following prompts:

`elephant#` , `donkey#`	Run this command as the `root` super-user, on the `elephant` or `donkey` hosts respectively. You might have to use a command like **su**(8) or **sudo**(8) to start a root shell before executing the command.
`myproxy@elephant%`	Run this command as the `myproxy` user, on the `elephant` host. This user is created automatically when the myproxy-server package is installed.
`quser@elephant%` , `quser@donkey%`	Run this command as the normal user account you are intending to interact with your Globus sevices, on the `elephant` or `donkey` hosts. In this document, we use the `quser` accout for this, but if you have another user, you can use it for that purpose.

Commands themselves will be typeset as **run-this-command -with-arguments**, and responses to the commands like this `Some Response Text`. If there is some portion of a command which should be replaced by value, such as a version number, it will be typeset like this: *`REPLACEME`*.

Finally, in some cases you will be prompted for a passphrase. When that occurs, the entry of the passphrase will be indicated by **`******`**, even though nothing will be printed to the screen.

Chapter 2. Installing GT 6.0

1. Installing Binary Packages

1.1. Prerequisites

We distribute the Globus Toolkit 6 as a set of RPM and Debian packages for Linux systems, as an installable package for Mac OS X, as a .zip file for Windows and Cygwin, as well as a source installer which can be used on other operating systems. In this quickstart, we will be installing RPM packages. Thus, it is a prerequisite for following this quickstart that you are running a distribution for which we provide RPMs. If you are running a supported Debian or Ubuntu system, the process is very similar, but you'll need to use the **apt-get** or similar tools to install the packages. For the source installer, there is more work involved, and you'll need to consult the full installation guide.

First, we will to set up our system to use the Globus package repository. This repository contains the Globus software packages, signed by our build manager. We provide RPM and Debian packages that contain a source configuration file and the public key which can be used to verify the packages. If your distribution has Globus 6.0 packages within its repository, you can skip to the next section.

The globus toolkit package repo RPM can be downloaded from the repo RPM package on globus.org[1].

To install binary RPMs, download the globus-toolkit-repo package from the link above and install it with the command:

```
elephant# rpm -hUv globus-toolkit-repo-6-1.noarch.rpm
```

The globus toolkit package repo Debian file can be downloaded from the repo Debian package on globus.org[2].

To install Debian or Ubuntu package, download the globus-toolkit-repo package from the link above and install it with the command:

```
elephant# dpkg -i globus-toolkit-repo_6-2_all.deb
```

Once you've installed the Globus repository package, you can use your operating system's packaging tools: **yum** or **apt-get**, to install the Globus components.

Important

For operating systems based on RHEL (such as Red Hat Enterprise Linux, CentOS, and Scientific Linux), the compatible EPEL repository must be enabled before installing myproxy. For OS versions 5.x, install the EPEL 5 package[3], and for OS version 6.x, use 6 package[4].

For information about installing these, see the EPEL FAQ[5].

This step is not needed for Fedora, Debian, or Ubuntu systems.

1.2. Installing the Toolkit on Linux

The components of the toolkit can be installed separately, or all at once. This section will show how to install various components, on both RPM based and Debian based Linux systems.

[1] http://www.globus.org/ftppub/gt6/installers/repo/globus-toolkit-repo-6-1.noarch.rpm
[2] http://www.globus.org/ftppub/gt6/installers/repo/globus-toolkit-repo_6-2_all.deb

For RPM-based systems, used the **yum** command to install the Globus components and their dependencies. For Debian-based systems, used the **apt-get** or **aptitude** commands.

For example, to install the GridFTP client tools, do the following for RPM-based systems:

```
elephant# yum install globus-data-management-client
```

Do the following for Debian-based systems:

```
elephant# apt-get install globus-data-management-client
```

1.2.1. Package Groups

The Globus Toolkit distribution includes several high-level package groups that can be used to install multiple packages to enable full client or server functionality of some Globus Toolkit component.

These packages are:

globus-gridftp	GridFTP client and server tools
globus-gram5	GRAM5 client and server tools
globus-gsi	Globus Security Infrastructure tools for managing certificates and proxies
globus-data-management-server	Server tools for deploying a GridFTP server.
globus-data-management-client	Client Tools for data management, including the GridFTP client programs and globus-url-copy
globus-data-management-sdk	Development headers and documentation for writing applications using the GridFTP APIs.
globus-resource-management-server	Server tools for deploying a GRAM5 resource manager
globus-resource-management-client	Client tools for resource management, including the globusrun tool, and the globus-job-* tools.
globus-resource-management-sdk	Development headers and documentation for writing applications using the GRAM5 APIs.

1.2.2. Updating a Globus Installation

In GT 6, there are three Globus Toolkit package repositories: *Stable*, *Testing*, and *Unstable*. The *Stable* repository is enabled by default, and is updated to to include fixes for major bugs and security issues. These can be easily installed via **yum** or **apt-get**. These updates will be published in the GT 6 updates rss feed[1]. Also, this means that when the next point release is made, collecting other minor bug fixes, the upgrade can be done via **yum** or **apt-get** without installing a new repository definition package.

In addition, users may enable the *Testing* or *Unstable* package repositories. These have different levels of documentation and testing done to them.

The *Testing* repository contains packages which have passed our automated test suite and are made available to people who are interested in the latest bug fixes. These packages will likely be migrated to the *Stable* repository

[1] http://www.globus.org/toolkit/rss/advisories/6.rss

once the package has been verified to fix a bug or issue and the documentation has been updated to include informtion about the issue.

The *Unstable* repository contains packages which have compiled successfully, but may not have completed all tests or are experimental in some way. Packages from the *Unstable* will potentially make it to the *Testing* repository once they seem to be functional.

1.3. Installing the Toolkit on Mac OS X

Download the Mac OS X Globus Toolkit Installation Package from the Globus Toolkit web site. Click on `globus_toolkit-6.0.pkg`, and follow the installation instructions. If you select the "Install for me only" option, your `$HOME/.profile` is modified to add the Globus Toolkit components to your path. If you are using a different shell, you may need to incorporate those changes into your shell initialization file. If you install for all users, the global path will be updated.

To uninstall the toolkit, run the **globus-uninstall** script which will remove the toolkit and revert the PATH changes.

1.4. Installing the Toolkit on Windows

There are four options when installing the Globus Toolkit on Windows: either using cygwin (32- and 64- bit builds) or MingW (32- and 64- bit builds).

The Cygwin installation requires the cygwin runtime (either 32-bit or 64-bit) to be installed: see cygwin.com[7] for details. To use the Globus Toolkit on cygwin, download the globus_toolkit-6.0-x86_64-pc-cygwin.zip or To use the Globus Toolkit on cygwin, download and unzip the globus_toolkit-6.0-i386-pc-cygwin.zip file and in the cygwin root directory. This will create files in /opt/globus

The mingw installtion does not require a special runtime, but some parts of the toolkit do not work with it: (LIST PENDING). To install the MingW packages, download the globus_toolkit-6.0-x86_64-w64-mingw32.zip or To use the Globus Toolkit on cygwin, download and unzip the globus_toolkit-6.0-i386-w64-mingw32.zip file. Add the unzipped directory's Globus\bin and Globus\sbin paths to your PATH environment to be able to use the Globus Toolkit.

2. Installation from Source Installer

Note

Installing using the Source Installer is only recommended on platforms for which native packages are not available. If you are installing onto a RedHat or Debian based Linux system, please see the section above.

Note

Make you sure you check out Platform Notes for specific installation information related to your platform.

2.1. Required software

To build the Globus Toolkit from the source installer, first download the source from download page[1], and be sure you have all of the following prerequisites installed.

This table shows specific package names (where available) for systems supported by GT 6.0:

[7] http://www.cygwin.com
[1] http://www.globus.org/toolkit/downloads/6.0

Prerequisite	Reason	RedHat-based Systems	Debian-based Systems	Solaris 11	Mac OS X
C Compiler	Most of the toolkit is written in C, using C99 and POSIX.1 features and libraries.	gcc	gcc	pkg:/developer/gcc-45 or Solaris Studio[9] 12.3	XCode[10]
GNU or BSD sed	Standard sed does not support long enough lines to process autoconf-generated scripts and Makefiles	sed	sed	pkg:/text/gnu-sed	(included in OS)
GNU Make	Standard make does not support long enough lines to process autoconf-generated makefiles	make	make	pkg:/developer/build/gnu-make	(included in XCode)
OpenSSL 0.9.8 or higher	GSI security uses OpenSSL's implementation of the SSL protocol and X.509 certificates.	openssl-devel	libssl-dev	pkg:/library/security/openssl	(included in base OS)
Perl 5.10 or higher	Parts of GRAM5 are written in Perl, as are many test scripts	perl	perl	pkg:/runtime/perl-512	(included in base OS)
pkg-config	Parts of GRAM5 are written in Perl	pkgconfig	pkg-config	pkg:/developer/gnome/gettext	Download and install from freedesktop.org source packages[11]

Note

In order to use the GNU versions of sed, tar, and make on Solaris, put `/usr/gnu/bin` at the head of your path. Also, to use all of the perl executables, add `/usr/perl5/bin` to your path.

2.2. Installing from Source Installer

1. Create a user named `globus`. This non-privileged user will be used to perform administrative tasks, deploying services, etc. Pick an installation directory, and make sure this account has read and write permissions in the installation directory.

Tip

You might need to create the target directory as `root`, then chown it to the `globus` user:

```
elephant# mkdir /usr/local/globus-6
elephant# chown globus:globus /usr/local/globus-6
```

Important

If for some reason you do *not* create a user named `globus`, be sure to run the installation as a *non-root* user. In that case, make sure to pick an install directory that your user account has write access to.

2. Download the required software noted in Section 2.1, "Required software".

3. The Globus Toolkit Source Installer sets the installation directory by default to `/usr/local/globus-6`, but you may replace `/usr/local/globus-6` with whatever directory you wish to install to, by setting the prefix when you configure.

   As the globus user, run:

   ```
   globus@elephant% ./configure --prefix=YOUR_PREFIX_DIRECTORY
   ```

   You can use command line arguments to ./configure for a more custom install.

   For a full list of options, see **./configure --help**.

4. The source installer will build all of the globus toolkit packages in the default make rule. The same package groups as the native packages may be used to build and install a subset of the toolkit.

   Run:

   ```
   globus@elephant% make PACKAGE-GROUPS
   ```

   Note that this command can take a while to complete. If you wish to have a log file of the build, use **tee**:

   ```
   globus@elephant% make 2>&1 | tee build.log
   ```

   The syntax above assumes a Bourne shell. If you are using another shell, redirect stderr to stdout and then pipe it to **tee**.

5. To test the toolkit, or particular packages within the toolkit, run:

   ```
   globus@elephant% make check
   ```

   or

   ```
   globus@elephant% make COMPONENT-check
   ```

   where *COMPONENT* is the name of the package to test. As an example, you could run

   ```
   globus@elephant% make globus_gssapi_gsi-check
   ```

   to run the GSSAPI test programs.

6. Finally, run:

   ```
   globus@elephant% make install
   ```

This completes your installation. Now you may move on to the configuration sections of the following chapters.

We recommend that you install any security advisories available for your installation, which are available from the Advisories page[1]. You may also be interested in subscribing to some mailing lists[13] for general discussion and security-related announcements.

2.3. Updating an Installation

The updates available in the native packages described above are also published as source packages on the updates page[1]. To install update packages, follow their download link, untar them, and then configure them with the same prefix as your original installation.

[1] http://www.globus.org/toolkit/advisories.html?version=6

Chapter 3. Basic Security Configuration

1. Obtain host credentials

You must have X.509 certificates to use the GT 6.0 software securely (referred to in this documentation as *host certificates*). For an overview of certificates for *GSI* (security) see GSI Configuration Information and GSI Environment Variables.

If you will need to be interoperable with other sites, you will need to obtain certs from a trusted Certificate Authority, such as those that are included in IGTF[1]. If you are simply testing the software on your own resources, SimpleCA offers an easy way to create your own certificates (see section below).

Host credentials must:

- consist of the following two files: `hostcert.pem` and `hostkey.pem`
- be in the appropriate directory for secure services: `/etc/grid-security/`
- match the hostname for a the machine. If the machine is going to be accessed remotely, the name on the certificate must match the network-visible hostname.

You have the following options:

1.1. Request a certificate from an existing CA

Your best option is to use an already existing CA. You may have access to one from the company you work for or an organization you are affiliated with. Some universities provide certificates for their members and affiliates. Contact your support organization for details about how to acquire a certificate. You may find your CA listed in the TERENA Repository[2].

If you already have a CA, you will need to follow their configuration directions. If they include a CA setup package, follow the CAs instruction on how to install the setup package. If they do not, you will need to create an `/etc/grid-security/certificates` directory and include the CA cert and signing policy in that directory. See Configuring a Trusted CA for more details.

This type of certificate is best for service deployment and Grid inter-operation.

1.2. SimpleCA

SimpleCA provides a wrapper around the OpenSSL CA functionality and is sufficient for simple Grid services. Alternatively, you can use OpenSSL's **CA.sh** command on its own. Instructions on how to use the SimpleCA can be found in Installing SimpleCA.

SimpleCA is suitable for testing or when a certificate authority is not available.

If you install the globus-simpleca native package, it will automatically create a CA and host certificate if you don't have one configured yet. Otherwise, you'll need to use **grid-ca-create** to create the CA and **grid-default-ca** to make that the default for requesting credentials.

[1] http://www.igtf.net
[2] http://www.tacar.org/

To create user credentials, you can run the command **grid-cert-request** as a user that you want to create a credential for. You can then run the **grid-ca-sign** command as the `simpleca` user to sign the certificate.

2. Add authorization

Installing Globus services on your resources doesn't automatically authorize users to use these services. Each user must have their own user certificate, and each user certificate must be mapped to a local account.

To add authorizations for users, you'll need to update the `grid-mapfile` database to include the mapping between the credentials and the local user accounts.

You'll need two pieces of information:

- the subject name of a user's certificate
- the local account name that the certificate holder can access.

To start with, if you have created a user certificate, you can run the **grid-cert-info** command to get the certificate's subject name, and **id -un** to get the account name:

```
globus@elephant% grid-cert-info -subject
/O=Grid/OU=GlobusTest/OU=simpleCA-elephant.globus.org/CN=Globus User
globus@elephant% id -un
globus
```

You may add the line by running the following command as root:

```
elephant# grid-mapfile-add-entry \
    -dn "/O=Grid/OU=GlobusTest/OU=simpleCA-elephant.globus.org/CN=Globus User" \
    -ln gtuser
Modifying /etc/grid-security/grid-mapfile ...
/etc/grid-security/grid-mapfile does not exist... Attempting to create /etc/grid-security/
New entry:
"/O=Grid/OU=GlobusTest/OU=simpleCA-elephant.globus.org/CN=Globus User" globus
(1) entry added
```

Important

The quotes around the subject name are *important*, because it contains spaces.

3. Verify Basic Security

Now that you have installed a trusted CA, acquired a hostcert and acquired a usercert, you may verify that your security setup is complete. As your user account, run the following command:

```
gtuser$ grid-proxy-init -verify -debug

User Cert File: /home/gtuser/.globus/usercert.pem
User Key File: /home/gtuser/.globus/userkey.pem

Trusted CA Cert Dir: /etc/grid-security/certificates

Output File: /tmp/x509up_u506
Your identity: /DC=org/DC=doegrids/OU=People/CN=GT User 332900
Enter GRID pass phrase for this identity:
```

```
Creating proxy ...++++++++++++
..................++++++++++++
 Done
Proxy Verify OK
Your proxy is valid until: Fri Jan 28 23:13:22 2005
```

There are a few things you can notice from this command. Your usercert and key are located in `$HOME/.globus/`. The proxy certificate is created in `/tmp/`. The "up" stands for "user proxy", and the *`_u506`* will be your UNIX userid. It also prints out your distinguished name (DN), and the proxy is valid for 12 hours.

If this command succeeds, your single node is correctly configured.

If you get an error, or if you want to see more diagnostic information about your certificates, run the following:

```
gtuser$ grid-cert-diagnostics
```

For more troubleshooting information, see the GSI troubleshooting guide

4. Firewall configuration

There are four possible firewall scenarios that might present themselves: restrictions on incoming and outgoing ports for both client and server scenarios.

This section divides sites into two categories: client sites, which have users that are acting as clients to Grid services, and server sites, which are running Grid services. Server sites also often act as client sites either because they also have users on site or jobs submitted by users to the site act as clients to other sites by retrieving data from other sites or spawning sub-jobs.

4.1. Client Site Firewall Requirements

This section describes the requirements placed on firewalls at sites containing Globus Toolkit clients. Note that often jobs submitted to sites running Globus services will act as clients (e.g. retrieving files needed by the job, spawning subjobs), so server sites will also have client site requirements.

4.1.1. Allowed Outgoing Ports

Clients need to be able to make outgoing connections freely from ephemeral ports on hosts at the client site to all ports at server sites.

4.1.2. Allowed Incoming Ports

As described in Section 3, “Job State Callbacks and Polling”, the Globus Toolkit GRAM service uses callbacks to communicate state changes to clients and, optionally, to stage files to/from the client. If connections are not allowed back to the Globus Toolkit clients, the following restrictions will be in effect:

- You cannot do a job submission request and redirect the output back to the client. This means the globus-job-run command won't work. globus-job-submit will work, but you cannot use globus-job-get-output. globusrun with the -o option also will not work.
- Staging to or from the client will also not work, which precludes the -s and -w options.
- The client cannot be notified of state changes in the job, e.g. completion.

To allow these callbacks, client sites should allow incoming connection in the ephemeral port range. Client sites wishing to restrict incoming connections in the ephemeral port range should select a port range for their

site. The size of this range should be approximately 10 ports per expected simultaneous user on a given host, though this may vary depending on the actual usage characteristics. Hosts on which clients run should have the GLOBUS_TCP_PORT_RANGE environment variable set for the users to reflect the site's chosen range.

4.1.3. Network Address Translation (NAT)

Clients behind NATs will be restricted as described in Section 4.1.2, "Allowed Incoming Ports" unless the firewall and site hosts are configured to allow incoming connections.

This configuration involves:

- Select a separate portion of the ephemeral port range for each host at the site on which clients will be running (e.g. 45000-45099 for host A, 45100-45199 for host B, etc.).
- Configure the NAT to direct incoming connections in the port range for each host back to the appropriate host (e.g., configure 45000-45099 on the NAT to forward to 45000-45099 on host A).
- Configure the Globus Toolkit clients on each site host to use the selected port range for the host using the techniques described in Section 2.1, "If client is behind a firewall".
- Configure Globus Toolkit clients to advertise the firewall as the hostname to use for callbacks from the server host. This is done using the GLOBUS_HOSTNAME environment variable. The client must also have the GLOBUS_HOSTNAME environment variable set to the hostname of the external side of the NAT firewall. This will cause the client software to advertise the firewall's hostname as the hostname to be used for callbacks causing connections from the server intended for it to go to the firewall (which redirects them to the client).

4.2. Server Site Firewall Requirements

This section describes firewall policy requirements at sites that host Grid services. Sites that host Grid services often host Grid clients, however the policy requirements described in this section are adequate for clients as well.

4.2.1. Allowed Incoming Ports

A server site should allow incoming connections to the well-known Grid Service Ports as well as ephemeral ports. These ports are 22/tcp (for gsi-enabled openssh), 2119/tcp (for GRAM) and 2811/tcp for GridFTP.

A server not allowing incoming connections in the ephemeral port range will have the following restrictions:

- If port 2119/tcp is open, GRAM will allow jobs to be submitted, but further management of the jobs will not be possible.
- While it will be possible to make GridFTP control connections if port 2811/tcp is open, it will not possible to actually get or put files.

Server sites wishing to restrict incoming connections in the ephemeral port range should select a range of port numbers. The size of this range should be approximately 20 ports per expected simultaneous user on a given host, though this may vary depending on the actual usage characteristics. While it will take some operational experience to determine just how big this range needs to be, it is suggested that any major server site open a port range of at least a few hundred ports. Grid Services should configured as described in Section to reflect the site's chosen range.

4.2.2. Allowed Outgoing Ports

Server sites should allow outgoing connections freely from ephemeral ports at the server site to ephemeral ports at client sites as well as to Grid Service Ports at other sites.

4.2.3. Network Address Translation (NAT)

Grid services are not supported to work behind NAT firewalls because the security mechanisms employed by Globus require knowledge of the actual IP address of the host that is being connected to.

We do note there have been some successes in running GT services behind NAT firewalls.

4.3. Summary of Globus Toolkit Traffic

Table 3.1. Summary of Globus Toolkit Traffic

Application	Network Ports	Comments
GRAM Gatekeeper(to start jobs)	To 2119/tcp on server from controllable ephemeral port on client	Connections back to client (controllable ephemeral port to controllable ephemeral port) required if executable or data staged from client or output from job sent back to client. Port 2119/tcp defined by IANA
GRAM Job-Manager	From controllable ephemeral port on client to controllable ephemeral port on server.	Port on server selected when original connection made by the client to the Gatekeeper and returned to the client in a URL. May result in connection back to client from ephemeral port on server to controllable ephemeral port on client.
GridFTP	From controllable ephemeral port on client to port 2811/tcp on server for control channel.	Port 2811/tcp defined by IANA.
GSI-Enabled SSH	From ephemeral port on client to port 22/tcp on server.	Same as standard SSH. Port 22/tcp defined by IANA.
MyProxy	From ephemeral port on client to port 7512/tcp on server.	Default. Can be modified by site.

4.4. Controlling The Ephemeral Port Range

Controllable ephemeral ports in the Globus Toolkit can be restricted to a given range. setting the environment variable GLOBUS_TCP_PORT_RANGE can restrict ephemeral ports. The value of this variable should be formatted as min,max (a comma separated pair). This will cause the GT libraries (specifically GlobusIO) to select port numbers for controllable ports in that specified range.

```
% GLOBUS_TCP_PORT_RANGE=40000,40010
% export GLOBUS_TCP_PORT_RANGE
% globus-gass-server
https://globicus.lbl.gov:40000
^C
%
```

This environment variable is respected by both clients and servers that are started from within the environment in which it is set. There are better ways, however, to configure a globus-job-manager or a GridFTP server to restrict its port range.

- globus-job-manager has an option, -globus-tcp-port-range PORT_RANGE that acts in the same manner as the environment variable. It can be specified on the command line or in the configuration file. See the job manager documentation for all of its options.
- See the GridFTP documentation for information about using GridFTP with firewalls.

Chapter 4. Basic Setup for GT 6.0

The Quickstart Guide walks you through setting up basic services on multiple machines.

Chapter 5. Platform Notes

1. Platform Notes

1.1. Mac OS X 10.8+ (Mountain Lion, Mavericks)

The GNU autotools and libtool is no longer distributed with OS X 10.8+. If you are building from git repository, you'll need to install the latest versions of those tools. If you are building from the source installer, these do not need to be installed.

- GNU Autoconf[1]
- GNU Automake[2]
- GNU Libtool[3]

Configure libtool with the configuration option **--program-prefix=g** to cause the libtool script to be named **glibtool** to avoid conflicts with the OS X libtool program which provides different functionality than GNU libtool. Install libtool (and the other tools) into the a common directory. If you do so, you'll need to set the `LIBTOOLIZE` environment variable to the path to the **glibtoolize** program. You'll need to include the autotools in your path to regenerate the configurable scripts and Makefile.in files for the toolkit.

The Globus Toolki build requires the **pkg-config** package to be installed. It is available from freedesktop.org[4]. Additionally, you'll need to set the environment variable `PKG_CONFIG_PATH` to `/usr/lib/pkgconfig` prior to running the configure script.

[4] http://pkgconfig.freedesktop.org/releases/

Chapter 6. Appendix

The Install Guide appendix can be found here.

Glossary

G

Grid Security Infrastructure (GSI)	GSI stands for Grid Security Infrastructure and is used to describe the original infrastructure of GT security, which is comprised of SSL, PKI and proxy certificates.

GT 6.0 Installation Appendix

GT 6.0 Installation Appendix

Table of Contents

List of Tables

List of Examples

Chapter 1. Advanced Installation for GT 6.0

This section introduces building from Git, and references the advanced installation sections of component documentation.

1. Advanced Installation

1.1. Building from Git

The Globus Toolkit is available for download from github. See https://github.com/globus/globus-toolkit/tree/globus_6_branch to branch or checkout the toolkit source code.

After checking out the toolkit, run **autoreconf -i** in the checkout root to generate configure scripts for building the toolkit components.

1.2. Building a specific package from source

If you need to build a specific package from the source installer, you can use the per-package make targets that exist in the source installer's Makefile. Instead of simply running "make" in the steps above, you can, for example, run "make globus_common" which will build the globus_common package and its dependencies, or "make globus_common-only" which will build exactly and only the globus_common package. Similar targets exist for each package.

1.3. Detailed installation instructions for these components

The following is a list of links to more detailed installation information available for the following components:

- GRAM5 Installation
- Building and installing GridFTP
- Building and installing MyProxy
- Optional Build-Time Configuration for GSI-OpenSSH

1.4. Building an update package without an installer

If you need to build an updated package that has been released without a source installer (for example, a security update to a package, or a new version of MyProxy,) you can use the familiar **configure; make; make install** sequence to rebuild that package.

Appendix A. Packaging details

1. The makefile

You do not have to build every subcomponent of this release. The makefile specifies subtargets for different functional subpieces.

Makefile targets

- gram: GRAM5
- gridftp: GridFTP

Note that all of these targets require the "install" target also. So, for instance, to build GridFTP alone, you would run:

```
$ ./configure --prefix=/path/to/install
$ make gridftp install
```

2. Linking with Globus Toolkit Libraries

Since GT 2.0, the toolkit has included a script called globus-makefile-header that can be used to assemble the cflags and link line information when linking a program with libraries included in the toolkit. This script would walk the package metadata dependency tree to ensure that all needed flags were included, without duplicates. This method worked, and continues to work in GT 6.0, but we consider it to be obsolete, as we have added support for using pkg-config[1]

Pkg-config is very similar in concept to globus-makefile-header, but it has gained widespread adoption across a range of unix platforms.

To get the cflags and link line information to link to the globus-ftp-client library, for example, you could

```
$pkg-config --cflags --libs globus-ftp-client
```

Each Globus Toolkit library has a pkg-config metadata file that is installed as part of its devel package.

For more information about pkg-config, please see the pkg-config homepage.[2]

[1] http://www.freedesktop.org/wiki/Software/pkg-config
[2] http://www.freedesktop.org/wiki/Software/pkg-config

Appendix B. Environment Variables in GT 6.0

1. Common Runtime Environmental Variables

1.1. Environmental variables for XIO

The vast majority of the environment variables that affect the Globus XIO framework are defined by the driver in use. The following are links to descriptions of the more common driver environment variables:

- TCP Driver Environment Variables[1]
- File Driver Environment Variables[2]
- GSI Driver Environment Variables[3]
- UDP Driver Environment Variables[4]

1.2. Environment variables for C Common Libraries

GLOBUS_HOSTNAME	Set this variable to the fully qualified name of the local machine's hostname.
GLOBUS_DOMAIN_NAME	Set this variable to the domain name to be used to qualify the local machine's hostname.
GLOBUS_ERROR_OUTPUT	Set this variable to 1 to cause Globus libraries to display error information to stderr.
GLOBUS_ERROR_VERBOSE	Set this variable to 1 to enable verbose error messages.
GLOBUS_I18N	Set this variable to 1 to attempt to use localized messages. (Currently not working)
GLOBUS_LOCATION	Set this variable to the path where the Globus Toolkit is installed, so that Globus tools can find libraries and data files. This is only needed if the Globus Toolkit was built with the source installer.
GLOBUS_THREAD_MODEL	Set to the name of a thread model to control the operation of the Globus event driver. Valid values are (depending on the platform) `none` for non-threaded operation (the default), `pthread` for POSIX threads, or `windows` for Windows threads.

2. Security Environmental Variables

2.1. Environmental Variables for GSI C

2.1.1. Credentials

Credentials are looked for in the following order:

1. service credential
2. host credential

3. proxy credential

4. user credential

`X509_USER_PROXY` specifies the path to the *proxy credential*. If `X509_USER_PROXY` is not set, the proxy credential is created (by **grid-proxy-init**) and searched for (by client programs) in an operating-system-dependent local temporary file.

`X509_USER_CERT` and `X509_USER_KEY` specify the path to the end entity (user, service, or host) certificate and corresponding *private key*. The paths to the certificate and key files are determined as follows:

For *service credentials*:

1. If `X509_USER_CERT` and `X509_USER_KEY` exist and contain a valid certificate and key, those files are used.

2. Otherwise, if the files `/etc/grid-security/`*`service/service`*`cert.pem` and `/etc/grid-security/`*`service/service`*`key.pem` exist and contain a valid certificate and key, those files are used.

3. Otherwise, if the files `$GLOBUS_LOCATION/etc/grid-security/`*`service/service`*`cert.pem` and `$GLOBUS_LOCATION/etc/grid-security/`*`service/service`*`key.pem` exist and contain a valid certificate and key, those files are used.

4. Otherwise, if the files *`service/service`*`cert.pem` and *`service/service`*`key.pem` in the user's `.globus` directory exist and contain a valid certificate and key, those files are used.

For *host credentials*:

1. If `X509_USER_CERT` and `X509_USER_KEY` exist and contain a valid certificate and key, those files are used.

2. Otherwise, if the files `/etc/grid-security/hostcert.pem` and `/etc/grid-security/hostkey.pem` exist and contain a valid certificate and key, those files are used.

3. Otherwise, if the files `$GLOBUS_LOCATION/etc/hostcert.pem` and `$GLOBUS_LOCATION/etc/hostkey.pem` exist and contain a valid certificate and key, those files are used.

4. Otherwise, if the files `hostcert.pem` and `hostkey.pem` in the user's `.globus` directory, exist and contain a valid certificate and key, those files are used.

For *user credentials*:

1. If `X509_USER_CERT` and `X509_USER_KEY` exist and contain a valid certificate and key, those files are used.

2. Otherwise, if the files `usercert.pem` and `userkey.pem` exist in the user's `.globus` directory, those files are used.

3. Otherwise, if a PKCS-12 file called `usercred.p12` exists in the user's `.globus` directory, the certificate and key are read from that file.

2.1.2. Gridmap file

`GRIDMAP` specifies the path to the *grid map file*, which is used to map distinguished names (found in certificates) to local names (such as login accounts). The location of the grid map file is determined as follows:

1. If the `GRIDMAP` environment variable is set, the grid map file location is the value of that environment variable.

2. Otherwise:

   - If the user is root (uid 0), then the grid map file is `/etc/grid-security/grid-mapfile`.

- Otherwise, the grid map file is `$HOME/.gridmap`.

2.1.3. Trusted CAs directory

`X509_CERT_DIR` is used to specify the path to the trusted certificates directory. This directory contains information about which CAs are trusted (including the *CA certificates* themselves) and, in some cases, configuration information used by **grid-cert-request** to formulate certificate requests. The location of the trusted certificates directory is determined as follows:

1. If the `X509_CERT_DIR` environment variable is set, the trusted certificates directory is the value of that environment variable.
2. Otherwise, if `$HOME/.globus/certificates` exists, that directory is the trusted certificates directory.
3. Otherwise, if `/etc/grid-security/certificates` exists, that directory is the trusted certificates directory.
4. Finally, if `$GLOBUS_LOCATION/share/certificates` exists, then it is the trusted certificates directory.

2.1.4. GSI authorization callout configuration file

`GSI_AUTHZ_CONF` is used to specify the path to the *GSI authorization callout configuration file*. This file is used to configure authorization callouts used by both the gridmap and the authorization API. The location of the GSI authorization callout configuration file is determined as follows:

1. If the `GSI_AUTHZ_CONF` environment variable is set, the authorization callout configuration file location is the value of this environment variable.
2. Otherwise, if `/etc/grid-security/gsi-authz.conf` exists, then this file is used.
3. Otherwise, if `$GLOBUS_LOCATION/etc/gsi-authz.conf` exists, then this file is used.
4. Finally, if `$HOME/.gsi-authz.conf` exists, then this file is used.

2.1.5. GAA (Generic Authorization and Access control) configuration file

`GSI_GAA_CONF` is used to specify the path to the GSI *GAA (Generic Authorization and Access control) configuration file*. This file is used to configure policy language specific plugins to the GAA-API. The location of the GSI GAA configuration file is determined as follows:

1. If the `GSI_GAA_CONF` environment variable is set, the GAA configuration file location is the value of this environment variable.
2. Otherwise, if `/etc/grid-security/gsi-gaa.conf` exists, then this file is used.
3. Otherwise, if `$GLOBUS_LOCATION/etc/gsi-gaa.conf` exists, then this file is used.
4. Finally, if `$HOME/.gsi-gaa.conf` exists, then this file is used.

2.1.6. Grid security directory

`GRID_SECURITY_DIR` specifies a path to a directory containing configuration files that specify default values to be placed in certificate requests. This environment variable is used only by the **grid-cert-request** and **grid-default-ca** commands.

The location of the *grid security directory* is determined as follows:

1. If the `GRID_SECURITY_DIR` environment variable is set, the grid security directory is the value of that environment variable.

2. If the configuration files exist in `/etc/grid-security`, the grid security directory is that directory.

3. if the configuration files exist in `$GLOBUS_LOCATION/etc`, the grid security directory is that directory.

2.1.7. Using TLS

`GLOBUS_GSSAPI_FORCE_TLS` specifies whether to use TLS by default when establishing a security context. The default behavior if this is not set is to use SSLv3.

2.1.8. Name Comparisons

`GLOBUS_GSSAPI_NAME_COMPATIBILITY` specifies what name matching algorithms are supported by GSSAPI for mutual authentication and `gss_compare_name`. This variable may be set to any of the following values:

`STRICT_GT2`	Strictly backward-compatible with GT 2.0 name matching. X.509 subjectAltName values are ignored. Names with hyphens are treated as wildcarded as described in the security considerations documentation. Name matching will rely on canonical host name associated with connection IP addresses.
`STRICT_RFC2818`	Support RFC 2818[5] server identity processing. Hyphen characters are treated as normal part of a host name. DNSName and IPAddress subjectAltName extensions are matched against the host and port passed to GSSAPI. If subjectAltName is present, X.509 SubjectName is ignored.
`HYBRID`	Support a hybrid of the two previous name matching algorithms, liberally matching both hyphen wildcards, canonical names associated with IP addresses, and subjectAltName extensions.

If this variable is not set, the `HYBRID` behavior is used.

2.2. Environmental variables for MyProxy

Please refer to the MyProxy Reference Manual[6] for documentation of MyProxy environment variable interfaces.

2.3. Environmental variables for GSI-OpenSSH

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*

[6] http://myproxy.ncsa.uiuc.edu/man/

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

3. Data Management Environmental Variables

3.1. Environment variables for GridFTP

The GridFTP *server* or *client* libraries do not read any environment variable directly, but the security and networking related variables described below may be useful.

- Non-WS (General) Authentication & Authorization Environment Variables.
- XIO Network Driver Environment Variables.

Appendix C. Installing SimpleCA

1. Create users

Make sure you have the following users on your machine:

- Your *user* account, which will be used to run the client programs.
- A `simpleca` account, which will be used to administer the Simple CA. This is created automatically if you install SimpleCA from RPM or Debian packages.
- A generic `globus` account, if you will be building from the source installer.

2. Install SimpleCA

SimpleCA can be installed in three ways, from a debian package, from an RPM package, and from the source installer. These installation methods are described in Installing GT 6.0

To install SimpleCA from binary packages, install the packages globus-simple-ca and globus-gsi-cert-utils-progs and their dependencies. On Debian based systems, use the command

```
elephant# apt-get install globus-simple-ca globus-gsi-cert-utils-progs
```

On RPM-based systems, use the command

```
elephant# yum install globus-simple-ca globus-gsi-cert-utils-progs
```

To install SimpleCA from the source installer, build the `globus_simple_ca` and `globus_gsi_cert_utils` installer targets with the command

```
globus@elephant% make globus_simple_ca globus_gsi_cert_utils
```

Afterward, run the command

```
globus@elephant% make install
```

3. Create SimpleCA Administrator Account

Create a user to adminster the SimpleCA. You can use the the `globus` user you used to build Globus, or another user that you create. For the purposes of this document, we'll assume a user named `simpleca`. Log in to that user, and run the **grid-ca-create** command. This will prompt for information needed to name the certificate, how to contact the CA administrator, lifetime of the CA certificate, and passphrase, and will then generate the new CA certificate and private key. Command-line options described in grid-ca-create can be used to avoid some of these prompts.

4. Invoking grid-ca-create

If you are creating a SimpleCA for testing purposes, you can use the `-noint` command-line option to **grid-ca-create** to use the default values for all prompts like this:

```
simpleca@elephant% grid-ca-create -noint
```

This will create a SimpleCA in the `simpleca`'s home directory with the passphrase `globus`. You can then move on to the Using a SimpleCA chapter of this document. For step-by-step details to create a customized SimpleCA, continue reading this chapter.

As the `simpleca` user, run the command **grid-ca-create**, and you'll see output like this:

```
simpleca@elephant% grid-ca-create
    C e r t i f i c a t e    A u t h o r i t y    S e t u p

This script will setup a Certificate Authority for signing Globus
users certificates.  It will also generate a simple CA package
that can be distributed to the users of the CA.

The CA information about the certificates it distributes will
be kept in:

/home/simpleca/.globus/simpleCA
```

This intro screen shows the path that the CA will be created into (in this example, `/home/simpleca/.globus/simpleCA`). The other commands needed by SimpleCA will automatically look in that path by default when invoked by the `simpleca` user.

5. Configure the subject name

The **grid-ca-create** program next prompts you for information about the name of CA you wish to create:

```
The unique subject name for this CA is:
cn=Globus Simple CA, ou=simpleCA-elephant.globus.org, ou=GlobusTest, o=Grid

Do you want to keep this as the CA subject (y/n) [y]:
```

To accept the default name, enter **y**. To choose a different name, type **n**, after which you will be prompted by

```
Enter a unique subject name for this CA:
```

The subject name is an X.509 distinguished name. Typical name component type abbreviations used in Grids are:

Table C.1. CA Name components

cn	Represents "common name". It identifies this particular certificate as the *CA Certificate* within the "GlobusTest/simpleCA-elephant.globus.org" domain, which in this case is Globus Simple CA.
ou	Represents "organizational unit". It identifies this CA from other CAs created by SimpleCA by other people. The second "ou" is specific to your hostname (in this case GlobusTest).
o	Represents "organization". It identifies the Grid.

6. Configure the CA's email

The next prompt looks like this:

```
Enter the email of the CA (this is the email where certificate
requests will be sent to be signed by the CA) [simpleca@elephant.globus.org]:
```

Enter the email address where you intend to receive certificate requests. It should be your real email address that you check, not the address of the globus user. When users request certificates with **grid-cert-request**, they will be instructed to send the request to this address.

7. Configure the expiration date

Then you'll see:

```
The CA certificate has an expiration date. Keep in mind that
once the CA certificate has expired, all the certificates
signed by that CA become invalid.  A CA should regenerate
the CA certificate and start re-issuing ca-setup packages
before the actual CA certificate expires.  This can be done
by re-running this setup script.  Enter the number of DAYS
the CA certificate should last before it expires.
[default: 5 years 1825 days]:
```

This is the number of days for which the CA certificate is valid. Once this time expires, the CA certificate will have to be recreated.

To accept the default, hit **enter**, or otherwise, enter a value in days.

8. Create a Passphrase to Encrypt the CA's Private Key

The next prompt will be for the passphrase for the CA's private key. It will be used to decrypt the CA's private key when signing certificates. It should be hard to guess, as its compromise might compromise all the certificates signed by the CA. You will be prompted twice for the passphrase, to verify that you typed it correctly. Enter the passphrase at these prompts.

```
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
```

9. SimpleCA Distribution Files

Finally **grid-ca-create** will create a tarball containing the public information about the CA, including its public certificate, signing policy, and supported X.509v3 extensions. This information is needed on machines that will be trusting the CA, and also on machines which will be used to request certificates from this CA.

Since we didn't run in this example as root, **grid-ca-create** will not be able to write the CA files to system paths, so it displays a warning message indicating that. We can use the tarball output here, or packages described below to install the CA support files on this and other machines.

The package output summary looks like this:

```
Insufficient permissions to install CA into the trusted certifiicate
directory (tried ${sysconfdir}/grid-security/certificates and
${datadir}/certificates)
```

```
Creating RPM source tarball... done
 globus_simple_ca_68ea3306
```

This information will be important for setting up other machines in your grid. The number *68ea3306* in the last line is known as your *CA hash*. It is an 8 digit hexadecimal string which is a hash of the subject name of the CA certificate.

The tarball contains Debian and RPM package metadata, so that it can be compiled to a binary package which can be easily installed on this and other systems on your Grid. It can also be packaged as a GPT setup package for compatibility with older versions of the Globus Toolkit.

10. Generating Binary CA Packages

The grid-ca-package command can be used to generate RPM, debian, or legacy GPT packages for a SimpleCA, or for any other CA which is installed on a host. These packages can make it easy to distribute the CA certificate and policy to other hosts with which you want to establish Grid trust relationships.

10.1. Generating RPM Packages

To generate an RPM package for the CA which we created, use the following command:

```
simpleca@elephant% grid-ca-package -r -cadir ~/.globus/simpleCA
Creating RPM source tarball... done
        globus_simple_ca_68ea3306.tar.gz
Creating RPM binary.../home/simpleca/globus-simple-ca-68ea3306
```

The resulting rpm package will be placed in the current directory. As root, you can install this via the **yum** or **rpm** tools. This package can then be installed on any RPM-based system.

10.2. Generating Debian Packages

To generate an Debian package for the CA which we created, use the following command:

```
simpleca@elephant% grid-ca-package -d -cadir ~/.globus/simpleCA
    Creating RPM source tarball... done
        globus_simple_ca_68ea3306.tar.gz
Creating debian binary...dpkg-buildpackage: export CFLAGS from dpkg-buildflags (origin: ve

...
Lots of dpkg-buildpackage output
```

The resulting debian package will be placed in the current directory. As root, you can install this via the **dpkg** tool.

10.3. Generating GPT Packages

The **grid-ca-package** command can also generate GPT packages in the form similar to previous versions of the Globus Toolkit. This is done with the `-g` and `-b` command-line options. See grid-ca-package for more details.

11. Examining a Certificate Request

To examine a certificate request, use the command **openssl req -text -in *REQNAME***, as shown in the following example.

Example C.1. Examine a Certificate Request

```
simpleca@elephant% openssl req -noout -text -in certreq.pem
Certificate Request:
    Data:
        Version: 0 (0x0)
        Subject: o=Grid, OU=GlobusTest, OU=simpleCA-elephant.globus.org, OU=local, CN=Joe
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (1024 bit)
                Modulus (1024 bit):
                    79:bd:a7:29:16:77:4c:e9:82:d3:73:a0:25:34:c7:
                    25:07:67:b3:2d:11:c1:e2:c9:b1:ec:41:20:a7:9a:
                    b7:2f:ee:d4:88:78:14:ff:d4:f2:f9:1b:d3:56:bc:
                    37:6f:f0:06:ea:b0:6f:70:12:a8:34:ac:8e:be:98:
                    00:b9:b8:ec:39:b5:6b:23:ad:1b:00:62:4b:cc:79:
                    97:cc:56:fb:54:7b:03:6d:a7:76:27:4e:ce:bd:94:
                    d0:eb:59:6b:25:c5:30:b0:47:15:bc:11:d5:7e:ff:
                    04:13:70:de:3b:8f:80:65:ae:63:82:61:38:f9:c6:
                    03:4a:92:b0:de:6f:bb:0a:bd
                Exponent: 65537 (0x10001)
        Attributes:
        Requested Extensions:
            Netscape Cert Type:
                SSL CA, S/MIME CA, Object Signing CA
    Signature Algorithm: sha1WithRSAEncryption
        85:70:a6:5d:de:be:61:45:83:48:43:8d:4b:4b:4a:79:79:98:
        0d:6c:d4:a9:96:26:41:a4:c2:94:10:92:ad:eb:ad:c5:3c:bf:
        d6:4e:84:0a:db:46:96:a9:52:5b:90:cc:6a:d1:57:73:27:98:
        9e:e2:8c:9a:7f:b4:ab:a8:28:2b:02:98:a2:d8:69:73:5e:12:
        ad:5b:de:0c:6e:60:e0:0f:2c:ad:8d:b9:59:3b:d3:49:19:52:
        e0:e1:8a:57:f2:c3:a6:4d:b9:2c:5c:58:ef:0e:59:84:55:8e:
        16:fc:f4:39:82:13:6f:28:a9:59:e3:c8:f1:4e:87:75:33:4f:
        ae:be
```

In this case, you see a certificate request with the subject distinguished name `o=Grid, OU=GlobusTest, OU=simpleCA-elephant.globus.org, OU=local, CN=Joe User`.

12. Signing a Certificate Request

If you are satisfied with the certificate request and are willing to sign it, use the **grid-ca-sign** command to do so. The command will store a copy of the newly signed certificate in the SimpleCA directory, so that you can keep track of what you've signed, and will also write it to the value of the `-out` parameter. Transmit this result file back to the user which requested the certificate.

Example C.2. Sign with grid-ca-sign

```
simpleca@elephant% grid-ca-sign -in certreq.pem -out cert.pem


To sign the request
please enter the password for the CA key:
```

```
The new signed certificate is at: /home/simpleca/.globus/simpleCA/newcerts/01.pem
```

Once you've signed the certificate, if it is a user certificate, you must communicate it back to the user, perhaps via email.

13. Revoking a Certificate

SimpleCA does not yet provide a convenient interface to revoke a signed certificate, but it can be done with the **openssl** command.

Example C.3. Revoke a certificate

```
simpleca@elephant% openssl ca -config ~/.globus/simpleCA/grid-ca-ssl.conf -revoke ~/.globu
Using configuration from /home/simpleca/.globus/simpleCA/grid-ca-ssl.conf
Enter pass phrase for /home/simpleca/.globus/simpleCA/private/cakey.pem:
Revoking Certificate 01.
Data Base Updated
```

Once a certificate is revoked, you can generate a Certificate Revocation List (CRL) for your CA, which will be a signed list of certificates which have been revoked. Sites which use your CA will need to keep the CRL up to date to be able to reject revoked certificates. This CRL can be generated with an **openssl** command. See ca(1) for details about how to control the CRL lifetime and other options.

Example C.4. Create CRL

```
simpleca@elephant% openssl ca -config ~/.globus/simpleCA/grid-ca-ssl.conf -gencrl > CAHASH
Using configuration from /home/simpleca/.globus/simpleCA/grid-ca-ssl.conf
Enter pass phrase for /home/simpleca/.globus/simpleCA/private/cakey.pem:
```

The output file *CAHASH*`.crl` (based on the hash of your CA subject name) should be distributed to sites which trust your CA, so that they can install it into the trusted certificate directory.

14. Renewing a CA

The **openssl** command can be used to renew a CA certificate. This will generate a new CA certificate with the same subject name and private key as before, but valid for a different time interval. This new certificate packaged and distributed as before using grid-ca-package.

Example C.5. Renew CA Certificate

```
simpleca@elephant% openssl req -key ~/.globus/simpleCA/private/cakey.pem -new -x509 -days
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Level 0 Organization [Grid]:
Level 0 Organizational Unit [GlobusTest]:
Level 1 Organizational Unit [simpleCA-elephant.globus.org]:
Name (E.g., John M. Smith) []:Globus Simple CA
```

Important

The Subject Name of the new certificate must match *exactly* the previous certificate name, or clients will not recognize it as the correct certificate.

15. Security considerations for SimpleCA

The operator of a CA must protect the private key of the CA. It should not be stored unencrypted or on a network filesystem.

Simple CA enforces the subject name policies in the simple CA's configuration files. If modified, the signing_policy file distributed to clients of the CA must also be modified.

Appendix D. Troubleshooting your installation

The following is a list of links that take you to information about troubleshooting your installation by component

- Common Runtime components
  - XIO
  - C Common Libraries
- Security components
  - GSI C
  - MyProxy
  - GSI-OpenSSH
- Data Management components
  - GridFTP
- Execution Management components
  - GRAM5

Appendix E. Detailed Configuration by Component

The following is a list of links that take you to information about detailed configuration for each component.

- Common Runtime components
  - XIO
- Security components
  - GSI C
  - MyProxy
  - GSI-OpenSSH
- Data Management components
  - GridFTP
- Execution Management components
  - GRAM5

Appendix F. Security Considerations in GT 6.0

1. Common Runtime

1.1. Security considerations for XIO

Globus XIO is a framework for creating network protocols. Several existing protocols, such as TCP, come built into the framework. XIO itself introduces no known security risks. However, all network applications expose systems to the risks inherent when outsiders can connect to them. Also included in the XIO distribution is the GSI driver, which provides a driver that allows for secure connections.

2. Security

2.1. Security considerations for GSI C

- During host authorization, the toolkit treats host names of the form "hostname-*ANYTHING*.edu" as equivalent to "hostname.edu". This means that if a service was set up to do host authorization and hence accept the certificate "hostname.edu", it would also accept certificates with DNs "hostname-*ANYTHING*.edu".

  The feature is in place to allow a multi-homed host following a "hostname-interface" naming convention, to have a single host certificate. For example, host "grid.test.edu" would also accept the likes of "grid-1.test.edu" or "grid-foo.test.edu".

  

  Note

  The string *ANYTHING* matches only the name of the host and not domain components. This means that "hostname.edu" will not match "hostname-foo.sub.edu", but will match "host-foo.edu".

  

  Note

  If a host was set up to accept "hostname-1.edu", it will not accept "hostname-*ANYTHING*.edu" but will accept "hostname.edu". That is, only one of the names being compared may contain the hyphen character in the host name.

  A bug[1] has been opened to see if this feature needs to be modified.

  In GT 6.0, it is possible to disable this behavior, by setting the enviornment variable `GLOBUS_GSSAPI_NAME_COMPATIBILITY` to `STRICT_RFC2818`.

2.2. MyProxy 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.

2.3. GSI-OpenSSH Security Considerations

GSI-OpenSSH is a modified version of OpenSSH[2] and includes full OpenSSH functionality. For more information on OpenSSH security, see the OpenSSH Security[3] page.

3. Data Management

3.1. Security Considerations

3.1.1. Ways to configure your server

There are various ways to configure your GridFTP server that provide varying levels of security. For more information, see System Administrator's Guide.

3.1.2. Firewall requirements

If the GridFTP *server* is behind a firewall:

1. Contact your network administrator to open up port 2811 (for GridFTP control channel connection) and a range of ports (for GridFTP data channel connections) for the incoming connections. If the firewall blocks the outgoing connections, open up a range of ports for outgoing connections as well.

2. Set the environment variable GLOBUS_TCP_PORT_RANGE:

   ```
   export GLOBUS_TCP_PORT_RANGE=min,max
   ```

   where `min`,`max` specify the port range that you have opened for the incoming connections on the firewall. This restricts the listening ports of the GridFTP server to this range. Recommended range is 1000 (e.g., 50000-51000) but it really depends on how much use you expect.

3. If you have a firewall blocking the outgoing connections and you have opened a range of (local) ports, set the environment variable GLOBUS_TCP_SOURCE_RANGE:

   ```
   export GLOBUS_TCP_SOURCE_RANGE=min,max
   ```

   where `min`,`max` specify the port range that you have opened for the outgoing connections on the firewall. This restricts the GridFTP server to bind to a local port in this range for outbound connections. Recommended range is twice the range used for GLOBUS_TCP_PORT_RANGE, because if parallel TCP streams are used for transfers, the listening port would remain the same for each connection but the connecting port would be different for each connection.

Note

If the server is behind NAT, the `--data-interface <real ip/hostname>` option needs to be used on the server.

If the GridFTP *client* is behind a firewall:

1. Contact your network administrator to open up a range of ports (for GridFTP data channel connections) for the incoming connections. If the firewall blocks the outgoing connections, open up a range of ports for outgoing connections as well.

[2] http://www.openssh.org/
[3] http://www.openssh.org/security.html

2. Set the environment variable GLOBUS_TCP_PORT_RANGE

   ```
   export GLOBUS_TCP_PORT_RANGE=min,max
   ```

   where min,max specify the port range that you have opened for the incoming connections on the firewall. This restricts the listening ports of the GridFTP client to this range. Recommended range is 1000 (e.g., 50000-51000) but it really depends on how much use you expect.

3. If you have a firewall blocking the outgoing connections and you have opened a range of ports, set the environment variable GLOBUS_TCP_SOURCE_RANGE:

   ```
   export GLOBUS_TCP_PORT_RANGE=min,max
   ```

   where min,max specify the port range that you have opened for the outgoing connections on the firewall. This restricts the GridFTP client to bind to a local port in this range for outbound connections. Recommended range is twice the range used for GLOBUS_TCP_PORT_RANGE, because if parallel TCP streams are used for transfers, the listening port would remain the same for each connection but the connecting port would be different for each connection.

Additional information on Globus Toolkit Firewall Requirements is available here[4].

4. Execution Management

4.1. Security Considerations

4.1.1. Gatekeeper Security Considerations

GRAM5 runs different parts of itself under different privilege levels. The **globus-gatekeeper** runs as root, and uses its root privilege to access the host's private key. It uses the *grid map file* to map *Grid Certificates* to local user ids and then uses the `setuid()` function to change to that user and execute the **globus-job-manager** program

4.1.2. Job Manager Security Considerations

The **globus-job-manager** program runs as a local non-root account. It receives a delegated limited *proxy certificate* from the GRAM5 client which it uses to access Grid storage resources via GridFTP and to authenticate job signals (such as client cancel requests), and send job state callbacks to registered clients. This proxy is generally short-lived, and is automatically removed by the job manager when the job completes.

The **globus-job-manager** program uses a publicly-writable directory for job state files. This directory has the *sticky* bit set, so users may not remove other users files. Each file is named by a UUID, so it should be unique.

4.1.3. Fork SEG Module Security Considerations

The Fork Scheduler Event Generator module uses a globally writable file for job state change events. This is not recommended for production use.

[4] http://www.globus.org/toolkit/security/firewalls/

Appendix G. Usage Statistics

The following components collect usage statistics as outlined here (along with information about how to opt-out): Usage Statistics in GT[1]

1. Data Management Usage Statistics

1.1. GridFTP-specific usage statistics

The following GridFTP-specific usage statistics are sent in a UDP packet at the end of each transfer, in addition to the standard header information described in the Usage Stats[1] section.

- Start time of the transfer
- End time of the transfer
- Version string of the server
- TCP buffer size used for the transfer
- Block size used for the transfer
- Total number of bytes transferred
- Number of parallel streams used for the transfer
- Number of stripes used for the transfer
- Type of transfer (STOR, RETR, LIST)
- FTP response code -- Success or failure of the transfer

Note

The client (globus-url-copy) does NOT send any data. It is the *servers* that send the usage statistics.

We have made a concerted effort to collect only data that is not too intrusive or private and yet still provides us with information that will help improve and gauge the usage of the GridFTP server. Nevertheless, if you wish to disable this feature for GridFTP only, use the `-disable-usage-stats` option of globus-gridftp-server. Note that you can disable transmission of usage statistics globally for all C components by setting "GLOBUS_USAGE_OPTOUT=1" in your environment.

Also, please see our policy statement[3] on the collection of usage statistics.

2. Execution Management Usage Statistics

2.1. GRAM5-specific usage statistics

The following usage statistics are sent by default in a UDP packet (in addition to the GRAM component code, packet version, timestamp, and source IP address) at the end of each job.

[1] ../../Usage_Stats.html
[1] /toolkit/docs/6/6.0/Usage_Stats.html
[3] /toolkit/docs/latest-stable/Usage_Stats.html

- Job Manager Session ID
- dryrun used
- RSL Host Count
- Timestamp when job hit `GLOBUS_GRAM_PROTOCOL_JOB_STATE_UNSUBMITTED`
- Timestamp when job hit `GLOBUS_GRAM_PROTOCOL_JOB_STATE_FILE_STAGE_IN`
- Timestamp when job hit `GLOBUS_GRAM_PROTOCOL_JOB_STATE_PENDING`
- Timestamp when job hit `GLOBUS_GRAM_PROTOCOL_JOB_STATE_ACTIVE`
- Timestamp when job hit `GLOBUS_GRAM_PROTOCOL_JOB_STATE_FAILED`
- Timestamp when job hit `GLOBUS_GRAM_PROTOCOL_JOB_STATE_FILE_STAGE_OUT`
- Timestamp when job hit `GLOBUS_GRAM_PROTOCOL_JOB_STATE_DONE`
- Job Failure Code
- Number of times status is called
- Number of times register is called
- Number of times signal is called
- Number of times refresh is called
- Number of files named in file_clean_up RSL
- Number of files being staged in (including executable, stdin) from http servers
- Number of files being staged in (including executable, stdin) from https servers
- Number of files being staged in (including executable, stdin) from ftp servers
- Number of files being staged in (including executable, stdin) from gsiftp servers
- Number of files being staged into the GASS cache from http servers
- Number of files being staged into the GASS cache from https servers
- Number of files being staged into the GASS cache from ftp servers
- Number of files being staged into the GASS cache from gsiftp servers
- Number of files being staged out (including stdout and stderr) to http servers
- Number of files being staged out (including stdout and stderr) to https servers
- Number of files being staged out (including stdout and stderr) to ftp servers
- Number of files being staged out (including stdout and stderr) to gsiftp servers
- Bitmask of used RSL attributes (values are 2^id from the gram5_rsl_attributes table)
- Number of times unregister is called

- Value of the `count` RSL attribute
- Comma-separated list of string names of other RSL attributes not in the set defined in `globus-gram-job-manager.rvf`
- Job type string
- Number of times the job was restarted
- Total number of state callbacks sent to all clients for this job

The following information can be sent as well in a job status packet but it is not sent unless explicitly enabled by the system administrator:

- Value of the executable RSL attribute
- Value of the arguments RSL attribute
- IP adddress and port of the client that submitted the job
- User DN of the client that submitted the job

In addition to job-related status, the job manager sends information periodically about its execution status. The following information is sent by default in a UDP packet (in addition to the GRAM component code, packet version, timestamp, and source IP address) at job manager start and every 1 hour during the job manager lifetime:

- Job Manager Start Time
- Job Manager Session ID
- Job Manager Status Time
- Job Manager Version
- LRM
- Poll used
- Audit used
- Number of restarted jobs
- Total number of jobs
- Total number of failed jobs
- Total number of canceled jobs
- Total number of completed jobs
- Total number of dry-run jobs
- Peak number of concurrently managed jobs
- Number of jobs currently being managed
- Number of jobs currently in the UNSUBMITTED state
- Number of jobs currently in the STAGE_IN state

- Number of jobs currently in the PENDING state
- Number of jobs currently in the ACTIVE state
- Number of jobs currently in the STAGE_OUT state
- Number of jobs currently in the FAILED state
- Number of jobs currently in the DONE state

Also, please see our policy statement[4] on the collection of usage statistics.

[4] /toolkit/docs/latest-stable/Usage_Stats.html

Glossary

C

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.
certificate	A public key plus information about the certificate owner bound together by the digital signature of a CA. In the case of a CA certificate, the certificate is self signed, i.e. it was signed using its own private key.
client	A process that sends commands and receives responses. Note that in GridFTP, the client may or may not take part in the actual movement of data.

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 section here.
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 this.
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[2] 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 GSI C Developer's Guide.
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 this.

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 place. GSI uses proxy certificates for single sign on and delegation of rights to other entities.

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

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

S

server — A process that receives commands and sends responses to those commands. Since it is a server or service, and it receives commands, it must be listening on a port somewhere to receive the commands. Both FTP and GridFTP have IANA registered ports. For FTP it is port 21, for GridFTP it is port 2811. This is normally handled via inetd or xinetd on Unix variants. However, it is also possible to implement a daemon that listens on the specified port. This is described more fully in in the Architecture section of the GridFTP Developer's Guide.

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

U

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