Globus Toolkit 1.1.3
System Administration Guide

December 2000

the globus project
www.globus.org

A joint project of Argonne National Laboratory and the University of Southern California's Information Sciences Institute, The Globus Project is developing the basic software infrastructure for computations that integrate geographically distributed computational and information resources.

Contents

Chapter 1 Introduction

This chapter provides an overview of the *Globus Toolkit 1.1.3 System Administration Guide*.

Audience

For purposes of this *Guide*, it is assumed that

- You and the users on your system(s) are familiar with Unix.
- You have some understanding of the concepts underlying computational grids.
- You are comfortable working with research software.

As you will learn in the next chapter, the Globus Toolkit is an evolving set of tools and services for building computational grids.

Organization

In general, the *Globus Toolkit 1.1.3 System Administration Guide* is a reference for a diverse set of materials relating to system administration. If you are new to system administration or administering the Globus Toolkit, we recommend that you review the document from beginning to end. If you are already familiar with the Globus Toolkit, you will probably want to skip around within the document to head straight for the information for which you are looking.

Chapter 2 is an overview of grids, the Globus Project, and the Globus Toolkit. Chapters 3 through 5 are devoted to describing preparations for and the actual installation of the Globus Toolkit. If you have already installed the Globus Toolkit and need a refresher for reinstalling it or installing it on a different system, you can go straight to Chapter 5, “Installing Globus.” Chapter 6 describes day-to-day management of the Globus Toolkit. Appendix A is a list of the Globus Toolkit command set and Appendix B is a list of the options to `globus-install`.

Related Documents

The *Globus Toolkit 1.1.3 System Administration Guide* is meant to be used in conjunction with other documents. Specific cross references to related materials from sources outside of the Globus Project are included in the text. Other Globus Toolkit documents can be found at www.globus.org/toolkit/documentation. They presently include:

Quick Start Guide
(http://www.globus.org/toolkit/documentation/QuickStart.pdf)
Release Notes
(http://www.globus.org/toolkit/download/release-113.html

Under development is:

Typographic Conventions

To facilitate your use of the *System Administration Guide*, the following typographic conventions have been used:

`Constant width`	is used to differentiate commands and examples from other text
< >	Angle brackets are used to indicate when generic values are to be replaced with user-supplied values. **NOTE:** The brackets are not to be entered and should not be confused with the Unix use of < and > to indicate redirection from stdin and to stdout.
Italics	are used within normal text for emphasis
%	represents a system prompt; do not enter
|	A vertical line between option parameters in a command line indicates multiple choice. One of the options separated by | must be entered.
[ ]	indicate parameters that sometimes may be omitted.
\	indicates that a command line continues. It is preceded by a space.

Technical Assistance

To obtain technical assistance, you can use the manual pages available with the Globus Toolkit. For brief explanations of some of the Globus Toolkit commands, enter

```
% <globus-command-name> -usage
% <globus-command-name> -help
```

For more detailed explanations, see

http://www.globus.org/v1.1/programs/<globus-command-name>.html

A complete listing of Globus Toolkit commands with descriptions is included in “Appendix A, Commands.” They are grouped by functionality.

If you need further assistance, please go to http://www.globus.org/about/contacts.html for details on how to communicate with the Globus Project. This web page includes a link to the Globus Toolkit problem report form and information on how to subscribe to mailing lists for the general Globus user community. You can also go to the Frequently Asked Questions (FAQ) section on the Globus Project web site, http://www.globus.org/about/faq.html.

Chapter 2 About Globus

The Globus Toolkit is designed to provide an integrated set of basic services for building computational grids. This chapter briefly defines grids, the Globus Project and the tools and services provided by the Globus Toolkit.

What Are Grids?

Grids are persistent environments that enable software applications to integrate instruments, displays, and computational and information resources that are managed by diverse organizations in widespread locations. *Grid applications* often involve large amounts of data and/or computing and are not easily handled by today's Internet and web infrastructures.

In this document, when we are referring specifically to the network of computers and other resources available to users of the Globus Toolkit, we use the term *Globus grid*. At the present time, there are other grids operating or being developed.

For more information on grid concepts, see *The Grid: Blueprint for a New Computing Infrastructure* (Morgan Kaufman, 1999, http://www.mkp.com/), edited by Ian Foster and Carl Kesselman, principal investigators of the Globus Project.

What Is the Globus Project?

The Globus Project is developing the fundamental technology that is needed to build computational grids. Research of the Globus Project targets technical challenges in, for example, communication, scheduling, security, information, data access, and fault detection. Development of the Globus Project focuses on the Globus Toolkit, an integrated set of tools and software that facilitate the creation of applications that can exploit the advanced capabilities of the grid and the creation of production or prototype grid infrastructures, using a combination of the Globus Toolkit and other technologies. Examples of applications work being done by groups around the world include:

- **Smart instruments**: Advanced scientific instruments, such as, electron microscopes, particle accelerators, and wind tunnels, coupled with remote supercomputers, users, and databases, to enable interactive rather than batch use, online comparisons with previous runs, and collaborative data analysis.
- **Computationally enhanced desktops**: Software packages, such as, chemical modeling and symbolic algebra, that transfer computationally intensive operations to more capable remote resources.
- **Collaborative engineering**: High-bandwidth access to shared virtual spaces that support interactive manipulation of shared data sets and steering of sophisticated simulations for collaborative design of complex systems.

- **Distributed computing**: Virtual supercomputers constructed from many individual supercomputers to solve problems too large for any single computer to accommodate.
- **Parameter studies**: Rapid, large-scale parametric studies, in which a single program is run many times in order to explore a multidimensional parameter space.

There are other software packages that provide similar services, but the Globus Toolkit differs from these in three significant ways:

- Its *bag of services* approach, which allows application software to use components of the Globus Toolkit without having to adopt the whole Globus Toolkit or a particular programming model or language.
- Its provision of specialized mechanisms that usually coexist with but also sometimes replace mechanisms provided by commodity computing.
- Its support for an information-based approach to meeting application performance requirements.

Using the basic services provided by the Globus Toolkit, researchers may build a range of higher-level capabilities. For example, the Globus Toolkit provides a complete implementation of the Message Passing Interface (MPI) that can run across heterogeneous collections of computers.

Due to its bag of services approach the Globus Toolkit can be used in different ways. The Globus Toolkit can be used at a site that wishes to participate in a computational grid to contribute resources to a grid's pool of resources. (Of course, the use of a site's grid resources is closely controlled by the site's access policies.) The Globus Toolkit can also be used to provide access to other grid resources without contributing any of a site's own resources, that is, assuming that the appropriate access policies have been negotiated with the owners of the other resources. The Globus Toolkit also provides other services, like single sign-on authentication, without the need for contributing computational resources.

The Globus Toolkit Components

The components of the Globus Toolkit can be used either independently or together to develop useful grid applications and programming tools.

Globus Resource Allocation Manager (GRAM)

The GRAM provides resource allocation and process creation, monitoring, and management services. GRAM implementations map requests expressed in a Resource Specification Language (RSL) into commands to local schedulers and computers. GRAM is one of the Globus Toolkit components that may be used independently. The user interface to GRAM is the *gatekeeper*.

Grid Security Infrastructure (GSI)

The GSI provides a single sign-on authentication service, with support for local control over access rights and mapping from global to local user identities. GSI support for hardware tokens increases credential security. GSI may be used independently and, in fact, has been integrated into numerous programs that are independent of the rest of the Globus Toolkit.

Metacomputing Directory Service (MDS)

The MDS is an integrated information service distributed across Globus-enabled resources that provides information about the state of the Globus grid infrastructure. The service is based on the Lightweight Directory Access Protocol (LDAP).

Global Access to Secondary Storage (GASS)

The GASS service implements a variety of automatic and programmer-managed data movement and data access strategies, enabling programs running at remote locations to read and write local data.

Globus Toolkit I/O

Globus Toolkit I/O provides an easy-to-use interface to TCP, UDP, and file I/O. It supports both synchronous and asynchronous interfaces, multithreading, and integrated GSI security.

Nexus

Nexus provides communication services for heterogeneous environments, supporting multimethod communication, multithreading, and single-sided operations.

Heartbeat Monitor (HBM)

The HBM allows you or your users to detect failure of the Globus Toolkit components or application processes.

For each component, a C-language (and in some cases, Java) application programming interface (API) is defined for use by Globus Team developers and those interested in developing Globus Toolkit applications.

Chapter 3 Planning Your Installation

This chapter provides details about what you need to consider prior to building, installing, and deploying the Globus Toolkit.

Tested Platforms

The Globus Team installs the Globus Toolkit on the following platforms prior to each software release and runs tests to ensure basic functionality. Bug reports are accepted from a wide range of hardware and software platforms, but problems with the platforms listed here are the easiest for the team to reproduce and test.

- AIX 4.2.1
- AIX 4.3.3 (IBM SP)
- IRIX 6.5
- Red Hat Linux 6.1
- Solaris 7
- UNICOS/mk 2.0.4.x (on Cray T3E)

Other platforms on which earlier releases of the Globus Toolkit have been installed include:

- AIX 4.3.1
- Digital Unix 4.0
- FreeBSD 2.x
- FreeBSD 3.x
- HPUX (V class machines) 10.20
- HPUX (V class machines) 11.x
- IRIX 6.4
- Linux (Intel-based) 2.2.8+ (requires GNU Libc 2 or greater for threads)
- Linux (Intel-based) 2.3.8+ (requires GNU Libc 2 or greater for threads)
- Linux (Intel-based) 2.4.x (requires GNU Libc 2 or greater for threads)
- Solaris 2.5
- Solaris 2.5.1
- Solaris 2.6
- Solaris 8
- UNICOS/mk 2.0.3.x (on Cray T3E)

Supported Scheduling Interfaces

The following scheduling interfaces are supported by the resource management architecture of the Globus Toolkit:

- Unix fork
- POE
- Condor
- Easy-LL (easymcs)
- NQE (on Cray T3E)
- Prun
- Loadleveler
- LSF
- PBS
- GLUnix
- Pexec

System Requirements

The hardware and software required for the Globus Toolkit are detailed in this section.

Hardware

In order to build, install, and run the Globus Toolkit on your system, please follow or consider these requirements or recommendations about your system's CPU, physical memory, and disk space.

CPU

The Globus software itself is not CPU intensive, but the computing power required to run the Globus Toolkit depends on what kind of host your system is used as.

If the host that will be running the Globus Toolkit will not be providing computational services for Globus Toolkit jobs then a moderately equipped system will suffice. In this configuration the purpose of the host is to provide a gateway to other resources that Globus Toolkit jobs can use. For example, for an array of multiprocessor systems or a cluster of workstations the gateway may act as a central entry point from which Globus Toolkit jobs will be distributed to other resources.

If the host is also to provide computing services for Globus Toolkit jobs, then the computing power should be enough to service the computational needs of the Globus Toolkit jobs targeted for the host.

Physical Memory

The Globus Toolkit itself is not memory intensive, therefore, the host on which it will run need only have a nominal amount of memory for the sake of the Globus Toolkit code.

Disk Space

Disk space requirements for building, installing, and deploying the Globus Toolkit can vary depending on the number of architectures and the number of development libraries that are built. Thus only approximate disk space requirements can be given.

Source Code. The approximate size of the compressed Globus Toolkit tar file is 10 megabytes (Mbyte). The approximate size of an uncompressed Globus tar file is 60 Mbyte.

Building the Globus Toolkit. The approximate size of the build space depends on how many communication package protocol or development libraries are being built. By default, the Globus Toolkit build process keeps the build space to a minimum by cleaning up the build space of unneeded files. If, however, the -`builddirs-persist` option is used then all files in the build directory remain intact. This option is used for debugging the build process and is usually not used for a normal build procedure. If the option is used, additional disk space will be needed to accommodate the build files.

Installing the Globus Toolkit. The Globus Toolkit installation procedure allows for a common location for Globus Toolkit code to be installed. This approach uses architecture-dependent subdirectories for differentiating between binaries from each architecture, thus, the amount of disk space required is dependent on the number of architectures for which the Globus Toolkit is built. For example, the approximate size of an install directory supporting three architectures and two types of communication packages per architecture is 150 Mbyte.

Deploying the Globus Toolkit. You should deploy the Globus Toolkit onto local disk space. During the deployment process, architecture-specific binary files are copied to a separate deployment location. Because some of these binaries are daemons typically run at machine bootup time, it is generally preferable that they not run from a mounted filesystem, that is, to ensure proper startup. The Globus Toolkit log files are also kept in one of the deploy subdirectories. Under normal conditions the log files do not grow very large, however, a rotation and/or archiving plan should still be considered. The approximate size of the deploy directory is 20 Mbyte.

Software

This section describes the software on which the Globus Toolkit depends and what additional software is recommended and why.

Required Software

In order to install and run the Globus Toolkit, you must install SSLeay and OpenLDAP.

SSLeay. In order to compile the Globus Toolkit, you will need to first download and install SSLeay. The Globus Security Infrastructure (GSI) is implemented in terms of a Generic Security Service Application Program Interface (GSSAPI) that is built on top of the SSLeay package. See “Install SSLeay 0.9.0” in Chapter 4 on page 14 for more information. OpenSSL, the follow-on to the SSLeay library, will be supported in a future release.

OpenLDAP. The Globus Toolkit uses modified libraries from the OpenLDAP distribution, though this will change in the future as the Globus Toolkit modifications are rolled back into OpenLDAP. Installing OpenLDAP is a straightforward process, but the Globus Toolkit patched distribution must be downloaded from the Globus Project FTP site. See “Install OpenLDAP” in Chapter 4 for more information.

Optional Software

We strongly recommend that you install a time synchronization mechanism, such as the Network Time Protocol, on all the machines running the Globus Toolkit. Time synchronization is important for authentication. When users attempt to authenticate they must present a proxy that has a timestamp and duration associated with it. If this proxy is presented to a host on which the time is not synchronized with the host on which the proxy was created, then users may not be able to authenticate (or login) with Globus Toolkit services. The timestamp of the proxy may be later than the current time on the host to which the proxy is being presented. In this case, the proxy will not be honored because its timestamp is later than the time on the local host.

Installation Considerations

There are several issues you should consider before beginning to install the Globus Toolkit.

Choosing a Host

Choosing the host on which the Globus Toolkit will be installed depends on how it will be used. If no resources will be contributed to the Globus grid resource pool, then the host on which the Globus Toolkit is installed is a matter of convenience. In that case, it is assumed that the Globus Toolkit will be used primarily for something like single sign-on authentication or to provide access to other resources in the Globus grid. Therefore, no special requirements are needed of the host on which the Globus Toolkit will be installed.

If the host on which the Globus Toolkit is installed is to be made available as part of the Globus grid resource pool, then you do need to consider issues such as computing power, disk space, and memory. However, running the Globus Toolkit on the same resource is acceptable because the Globus Toolkit is not CPU intensive and should not have an impact on any Globus grid jobs requesting use of this host.

Choosing Filesystems

There are three considerations to keep in mind when choosing the appropriate filesystems on your system on which to build, install, and deploy the Globus Toolkit:

- The Globus Toolkit is better installed on a *shared* filesystem.

  The Globus Toolkit is designed so that several architecture-dependent versions can be installed in a common location on your system. That way all the different hosts on which the Globus Toolkit is deployed can share information from the installation directory. Using a shared filesystem improves the performance and reliability of the Globus Toolkit by removing its dependency on files that might become inaccessible otherwise. (See the Chapter 3, “System Requirements” section on Disk Space to be sure you have enough space to accommodate building of the code in this common location.)

- *Each* host running a Globus Toolkit gatekeeper must be able to accommodate deployment of the Globus Toolkit on its local filesystem.

- The deployment location must have room for the log files.

  While these log files can be managed with a rotation and archival plan, there should be enough space to accommodate the space needed for log file output on a daily basis.

Contributing Resources

Before you decide to contribute your site's resources to the Globus grid resource pool, you should evaluate the impact of Globus Toolkit job requests on these resources.

Local Site Policies

Many sites have policies regarding the use of their resources. These policies will have an impact on the choice of resources to contribute and the extent to which they can be made available. You must consider limits on the amount of physical resource such as CPU, memory, and disk you have; limits on the number of jobs a user can submit or run; and the cost and level of service. You should consider how and if these policies can be enforced on Globus Toolkit jobs.

Level of Service

If you install the Globus Toolkit, GRAM provides a convenient way of submitting and monitoring jobs remotely. The level of service you provide for Globus Toolkit jobs is determined as it would be for any other application you are running.

Accounting

Resource usage accounting is presumed to be handled locally by each site. The Globus Toolkit does not change any existing local accounting mechanisms. Globus Toolkit jobs run under the user account as specified in the grid-mapfile. Therefore, a user is required to already have a conventional Unix account on the host to which the job is submitted.

Information Services

With Release 1.1.3, the method by which Globus Toolkit information services provide information about the state of the Globus grid infrastructure has changed substantially. You need to be aware of these changes when preparing to install the Globus Toolkit. In previous releases of the Globus Toolkit, the MDS is implemented using a *push model:* information about Globus resources is stored in a central directory server that is updated periodically by daemons running on Globus resources. (The push model is subsequently referred to as the *classic MDS model*.) The new *pull model* of the MDS allows for the definition and use of multiple and different types of information services using multiple technologies like directory servers or relational databases. (The pull model is subsequently referred to as the *distributed MDS model*.) The main difference between the classic and distributed MDS models is that under the classic model system information is only stored and made available via a centralized MDS server, the *organizational server*. Also, pre-1.1.3 releases did not include an implementation of an MDS server in the software distribution. System administrators had to either share in the use of a global MDS server or implement their own centralized MDS server, typically using commercial LDAP servers. For details on the classic MDS model, see the *Globus Toolkit 1.1.2 System Administration Manual*.

Release 1.1.3 of the Globus Toolkit includes two specialized types of information services: a Grid Resource Information Service (GRIS) and an optional Grid Index Information Service (GIIS).

GRIS

The GRIS is a distributed information service that can answer queries about a particular resource by directing the query to an information provider deployed as part of the Globus services on a grid resource. Examples of information provided by this service include host identity (e.g., OS and versions), as well as more dynamic information such as CPU availability. Each resource on which the Globus Toolkit is installed will run a GRIS. Therefore, each GRIS is responsible for providing information only about the resource on which it is running. By default, a GRIS is automatically configured and will be assigned to use the IANA-approved port 2135.

GIIS

The GIIS represents a centralized MDS server that provides information about all of your resources. By default, a GIIS is not configured. If you do not have a GIIS your system will still be fully functional, but you will not have a centralized server that contains information about all of your resources. Each resource will have its own LDAP information service that can be connected to remotely for obtaining system and status information. The GIIS can only be configured to run on a host on which the Globus Toolkit will be installed.

The default configuration includes a GRIS and the use of the Domain Name Service (DNS) component distinguished name for the resource on which Globus is being installed. The `globus-setup` command can be used to define the host and port information for the site's GIIS. The normal configuration for an organization would be to have a GRIS on each resource running Globus and one GIIS managed by the lead site for that organization.

If you will be deploying the Globus Toolkit gatekeeper on multiple resources, you may want to operate the optional GIIS. To do so, you must choose a host/port on which to run the service and provide this information during the installation of the software.

Security

The Globus Toolkit uses an authentication system known as gssapi_ssleay, the Generic Security Service API based on Eric A. Young's implementation of Secure Sockets Layer (SSL). This system uses the RSA encryption algorithm for its encryption, therefore employing both public and private keys.

X.509 Certification Process

The gssapi_ssleay authentication relies on an X.509 certification process. Globus Toolkit users place their X.509 certificates in their home directories, thus identifying themselves to the system.

User Certificates and Keys. The X.509 certificate includes information about the duration of the permissions, the RSA public key, and the signature of the Certificate Authority (CA). With the certificate is the user's private key. In the Globus Toolkit, the certificates can be created only by the CA, who reviews the X.509 certificate request submitted by the user and accepts or denies it according to an established policy. The request and keys are created using a certificate request generation script located on the user's system. The user's public key is inserted into the certificate request and the user's private key is saved in a separate file, which is encrypted using the user's pass-phrase (password). The request is emailed to the CA, who upon receiving the request, reviews it, and signs it electronically. This signed certificate is then emailed back to the user, who saves the file into his directory tree. The user must also save into his directory a copy of the trusted certificate of the CA. This copy of the CA's signature is used during job requests in order to verify the identity of the remote resource. The Globus Toolkit supports the use of multiple CAs, as are defined in the appropriate Globus configuration files.

Gatekeeper Certificates and Keys. The gatekeeper also must have a certificate and key. They are requested and created in a like manner by the system administrator using the Globus Toolkit certificate request generation script. The certificate and key for the gatekeeper, however, do not require a pass-phrase. Within the gatekeeper's home directory is a copy of the trusted Globus Toolkit CA's certificate, which is used to validate job requests from clients. When a job is submitted to the gatekeeper by the client, before any allocation of resources occurs, a process of mutual authentication ensures that the client has permission to execute jobs on the computer and that the gatekeeper is the correct resource.

Proxies. The gssapi_ssleay authentication requires the use of proxies, a convenient mechanism for reducing the number of times users must enter their pass-phrase. If many jobs are to be submitted over the course of a few hours, the tedium of reentering the pass-phrase can be avoided by creating a proxy. The proxy is a certificate and key, signed by the user, created by using the `grid-proxy-init` program that uses the user's name in the subject name of the proxy and a /CN=proxy appended to it. The number of hours for which it will be valid, as well as the number of bits in the key, can be set by the user. The intention is that these proxies be used for short- rather than long-term authentication. When proxies are used, the mutual authentication process differs slightly. The gatekeeper receives not only the certificate of the proxy, which is signed by the user or another proxy, but also the certificate of the user that signed the proxy's certificate.

Proxy files must be kept secure, within your system's local filesystem, rather than on the Network File System (NFS). They must allow only the user to have read-access to them, as they essentially allow job submission without pass-phrase protection, a feature that can potentially compromise the security of the system.

Encoding. The certificates and keys are stored in *PEM format*, a base-64 encoding that can be transmitted via email or edited with a text editor. The files in which the certificates and keys are included may also contain comments and other relevant information. The system regards everything between the ----BEGIN CERTIFICATE---- and ----END CERTIFICATE---- labels as the certificate, and everything between the ----BEGIN RSA PRIVATE KEY---- and ----END RSA PRIVATE KEY---- as the key.

Environment Variables

The Globus Toolkit locates the certificate and key files in the user's ~/.globus directory by default or by referring to environment variables. There are five basic environment variables that can be set:

Variable	Description
X509_CERT_DIR	Directory containing trusted certificates. The default is ~/.globus if this variable is not set.
X509_CERT_FILE	File in which one or more trusted certificates are stored (not normally used)
X509_USER_CERT	File in which the user's certificate can be found. The default is ~/.globus/userkey.pem if the variable is not set.

Variable	Description
X509_USER_KEY	File containing the key for the above certificate. The default is ~/.globus/userkey.pem if the variable is not set.
X509_USER_PROXY	File in which a proxy certificate, key, and additional certificates can be found, normally in /tmp/x509_u<uid>

If the certificate and key are located in the same file, there is no need for an X509_USER_KEY variable. The system will search the file referred to by X509_USER_CERT for the key automatically. Either X509_CERT_DIR or X509_CERT_FILE is required; both may be used if a user has many trusted certificates. X509_USER_PROXY is always followed, if it is defined, overriding X509_USER_CERT/X509_USER_KEY.

Chapter 4 Before Installing Globus

Now that your planning has been completed, you must perform the steps described in this chapter before you install the Globus Toolkit on your system.

Getting Started

To get started installing the Globus Toolkit, you must perform these tasks first. Except where noted below you should perform the rest of the installation steps as the user identified as globus.

- Create a separate user id, such as globus, under which Globus Toolkit daemons will run. While it is possible to run the Globus Toolkit under an existing user id, we strongly recommend that you not do so to facilitate tracking of Globus Toolkit daemon activities. The globus user id is only used to run the unprivileged Globus daemons. User jobs submitted to the Globus Toolkit run under the user's own local user id.
- Create a directory in which to install all Globus Toolkit software (e.g., /usr/local/globus). This directory should be writeable by the user globus created above.
- You might also want to create a separate group id, such as globus, to allow more than one user to perform Globus Toolkit system administration tasks.

Install Required Software

Both SSLeay and OpenLDAP must be installed *prior* to installing the Globus Toolkit.

Install SSLeay 0.9.0.

In order to compile the Globus Toolkit, you will need to first download and install SSLeay. For more information on SSLeay see:

ftp://ftp.psy.uq.oz.au/pub/Crypto/SSL

The GSSAPI implementation uses many of the SSLeay routines and does not require any specific modifications to the SSLeay code. If you uncover any SSLeay bugs, please report them to the Globus Team by going to

http://www.globus.org/about/contacts.html

and using the problem report form linked from there. You may also want to contact the SSLeay mailing list.

We are currently recommending the use of SSLeay 0.9.0b, which has improved error message support, better 64-bit support, and support for certificate revocation lists.

Applying a Patch

The Globus Toolkit is being deployed on some architectures for which SSLeay has not been tested previously and the SSLeay `Configure` script may need to be updated. The appropriate architecture-specific patch file:

> ssleay-0.9.0-irix-patch (ftp://ftp.globus.org/pub/globus/ssleay-0.9.0-irix-patch)
> ssleay-0.9.0-hpux11-patch (ftp://ftp.globus.org/pub/globus/ssleay-0.9.0-hpux11-patch)
> ssleay-0.9.0-t3e-patch (ftp://ftp.globus.org/pub/globus/ssleay-0.9.0-t3e-patch)

should be applied if you wish to run on IRIX, HPUX 11.01, or the Cray T3E. Note that these patches are also available in the Globus Toolkit source distribution.

To apply a patch to your SSLeay distribution:

```
% cd <SSLeay-src-dir>
% patch -p0 < <full-path-to-location-of-patch-file>
```

for example:

```
% patch -p0 < /usr/local/ssleay-0.9.0-<arch>-patch
```

Configuring SSLeay

The `Configure` script for SSLeay requires Perl. If you do not have Perl on the architecture on which you are compiling, you will need to install it. If that is not possible, configure SSLeay on another architecture. You should be able to run the `Configure` script with the appropriate options for your target architecture and then move the directories to a filesystem that is available from the machine on which you want to build.

The SSLeay `Configure` process changes a number of its source files. SSLeay must be compiled in its source directory. Symbolic links between a build directory and the SSLeay source directory will not work.

A web page containing the SSLeay FAQ written by Tim Hudson can be found at http://www.psy.uq.oz.au/~ftp/Crypto.

You *must* use the same compilers for SSLeay and Globus Toolkit builds. You can not mix 32-bit and 64-bit object files in the systems we support. Also, you can not mix a GCC-compiled SSLeay with a non-GCC-compiled Globus Toolkit.

Recommended SSLeay configuration options for various architectures are:

Architecture	**Options to SSLeay Configure**
AIX 4.x	aix-cc
FreeBSD	gcc
Digital Unix 4.0	alpha-cc
IRIX 6.x 32 bit	irix-cc-n32*
IRIX 6.x 64 bit	irix-cc-n64*

Architecture	Options to SSLeay Configure
HPUX 11.01 32bit	hpux11-32bit-cc*
HPUX 11.01 64bit	hpux11-64bit-cc*
Solaris (with SunPro Compiler)	solaris-sparc-cc
Solaris (gcc)	solaris-sparc-gcc
Linux	linux-elf
Cray T3E	cray-t3e*

* *Configuration options marked with an asterisk require a patch to be applied* before *configuring.*

Building SSLeay

To build SSLeay as described in the INSTALL file from the SSLeay-0.9.0b distribution, you should follow the steps given below. These steps should work in most cases. Examine the INSTALL file from the SSLeay distribution for more details. These steps presume Perl is available on the system. If Perl is not available, please refer to the SSLeay INSTALL file for instructions on how to build SSLeay.

Fix paths. If you have Perl and it is not in /usr/local/bin, you can run the following command to fix the paths in all SSLeay scripts.

NOTE: Do *not* put /new/path/perl, just /new/path.

```
% perl <SSLeay-src-dir>/util/perlpath.pl /new/path
```

For example, you can run

```
% perl <SSLeay-src-dir>/util/ssldir.pl /new/ssl/home
```

Setup machine-dependent parameters. To do this, run

```
% <SSLeay-src-dir>/Configure <system type>
```

where <system type> represents the system for which SSLeay is being built. For a list of supported system types, run the SSLeay `Configure` command without any arguments:

```
% <SSLeay-src-dir>/Configure
```

Clean out old files. Run the command

```
% make clean
```

For additional cleanup, if you have the `makedepend` command installed, you can run

```
% make depend
```

This step is not required, but it is useful when you are going to be developing applications.

For performance reasons, there are assembler versions of some of the SSLeay routines. These may be very specific to the CPU chip and compiler revision being used. If you are not using the specific chip in question, we would recommend not using these assembler routines, but rather the C versions. To do this, run:

```
% make BN_MULW=bn_mulw.o
```

To use the assembler versions, simply run

```
% make
```

Installing SSLeay

To install all the SSLeay code that was just built, run

```
% make install
```

Testing the SSLeay Installation

Once you have built SSLeay, you should run the SSLeay tests provided with it to verify that all is working. This can be done by running the command:

```
% make tests
```

It is okay if you see errors about certificates being expired. SSLeay is very sensitive to errors in compiler optimizations. If you have errors running the SSLeay tests, try removing any optimization flags (usually `-o` or `-O` followed by a number) from the SSLeay Makefile.

Install OpenLDAP

Starting with version 1.1.0b27, the Globus Toolkit uses libraries from the OpenLDAP distribution, in the same manner as it uses SSL libraries from SSLeay. However, in testing the Globus Team has encountered some weaknesses in the current release of OpenLDAP (1.2.7). Therefore, please download our patched distribution from the Globus Project FTP site:

ftp://ftp.globus.org/pub/globus/OpenLDAP-1.2.7-globus-latest.tar.gz

To begin installing OpenLDAP, define what compiler to use by setting the CC environment variable:

Architectures	**CC Environment Variables**
AIX 4.x	xlc
FreeBSD	gcc
IRIX 6.x, 32 bit	cc -n32
IRIX 6x, 64 bit	cc -64
HPUX 11.01, 32 bit	???
HPUX 11.01, 64 bit	???
Linux	gcc
Solaris 2.{5,6} (SunPro Compiler)	/opt/SUNWspro/bin/cc
Solaris 7 (SunPro Compiler)	/opt/SUNWspro/SC5.0/bin/cc

Architectures	CC Environment Variables
Cray T3E	/opt/ctl/bin/cc

For sh-type (or Bourne type) shells:

```
CC=<setting>;export CC
```

For csh-type shells:

```
setenv CC <setting>
```

Next, configure OpenLDAP *non-threaded*. We recommend that you use the following options:

```
configure --prefix=<ldap-install-path> --enable-slapd \
--enable-shell --disable-ldbm --without-threads
```

After you have configured:

```
% make depend
% make
% make install
```

Please be aware that several warning messages will be generated when OpenLDAP is compiled. These messages can be ignored.

Install Optional Software

Information about downloading and installing the Network Time Protocol (NTP) can be found at http://www.eecis.udel.edu/~ntp/

Create User ID for Globus Toolkit daemons

A special user account should be used under which the Globus Toolkit daemons will run. Doing so can facilitate tracking of Globus software and configuration changes, as well as producing accounting records associated with Globus Toolkit daemon activities. Identifying a special group for the account will facilitate execution of Globus administrative tasks among several different system administrators belonging to that group.

Download the Globus Toolkit Software

The Globus Toolkit software is available from the Globus Project FTP site. The latest 1.1.x release is available from:

ftp://ftp.globus.org/pub/globus/globus-latest.tar.gz

Update patches to the Globus Toolkit are available from:

ftp://ftp.globus.org/pub/globus/patches

Uncompressing and untarring the Globus source distribution will create the following directory structure under globus/:

VERSION	The version number of the Globus Toolkit release.
CONTRIBUTORS	A list of contributors to the Globus Project.
README	Pointers to on-line documentation and email lists
globus-install	The script you run to configure and install the Globus Toolkit software.
configure	The configuration script used internally by `globus-install`
configure.in Makefile.in aclocal.m4	Files used internally by `autoconf` and `configure`
ssleay-0.9.0-hpux11-patch ssleay-0.9.0-irix-patch ssleay-0.9.0-t3e-patch ssleay-0.9.0-cygwin-patch	Patches to compile the SSLeay distribution on various platforms
ResourceManagement/	The source code for the Globus Toolkit Resource Management Architecture. This includes the GRAM, DUROC (Dynamically-Updated Request Online Coallocator), and RSL implementations
Security/	The source code for the Globus Security Infrastructure (GSI). This includes an SSL-based implementation of the GSSAPI, and the GSS-Assist helper library
Communication/	The source code for the Globus Toolkit Communications libraries. These include the Nexus multi-method communication library and the DUCT communication bootstrapping library
Startup/	The source code for the Globus Toolkit daemon startup scripts
Configuration/	The macros used to handle portable compilation of the Globus Toolkit
Examples/	A directory containing example code for the Globus Toolkit APIs

FileAccess/	The source code for GASS
HealthAndStatus/	The source code for the HBM
InformationServices/	The source code for MDS
Miscellaneous/	The source code for various scripts distributed with the Globus Toolkit distribution and the globusrun job startup tool

Chapter 5 Installing Globus

Installation of the Globus Toolkit is best thought of as three phases:

- Phase I: Build (build configuration, compilation, and linking)
- Phase II: Setup (setting up conf and system files)
- Phase III: Deploy (deploying all programs/scripts to their locations)

We recommend that you perform all of the steps described in this chapter as the user globus to ensure that all permissions will be set correctly. If you did not create an account for the user globus, you should perform these steps from whatever account you run the Globus Toolkit daemons.

Also, the Globus Toolkit requires very robust versions of the standard Unix shell tools to complete the configuration of the software. If you are running on certain systems, we recommend installing the GNU versions of `sed` and `make`, and putting them in your path before the vendor-supplied versions. Otherwise, the Globus Toolkit may fail to compile. The systems we have noted problems on are:

- HP: HPUX 10.x and 11.01
- IBM: AIX 4.2 and earlier
- Cray: UNICOS on Cray T3E

Globus Toolkit Build

The Globus Toolkit installation process makes use of programs like `autoconf` and `configure`. This approach aids in the build process by providing host-specific software information to identify compilers and other tools to be used during compilation.

The first phase of the installation process is the configuration and compilation of the Globus Toolkit code. This phase is also the one in which the Globus Toolkit configuration parameters are specified.

To build the Globus Toolkit, you must do a build for each architecture type at your site on which the Globus Toolkit will be installed. The `globus-install` script will configure, compile, and install all software. The directory structure rooted at <globus-install-dir> has been designed for a heterogeneous environment, so that you can execute the command on several platforms using the same directory as the <globus-install-dir>.

The general form of the `globus-install` command is:

```
% <globus-src-dir>/globus-install [<globus-install options>] \
 [- <configure options>]
```

The usual syntax that you will need to use is

```
% <globus-src-dir>/globus-install -prefix=<globus-install-dir> \
-with-ssl-path=<ssl-install-path> \
-with-ldap-path=<ldap-install-path>
```

You will be prompted for these values if they are not provided:

`-prefix=`	Path to where Globus is to be installed
`-with-ssl-path=`	Path to where SSLeay was installed
`-with-ldap-path=`	Path to where OpenLDAP was installed

The SSLeay and OpenLDAP paths should point to the root of the installations, respectively. Defaults for these values are provided in “Appendix B”.

If the following conditions are met, you should add these extra options to `globus-install`:

Condition	**Additional Option**
Configured SSLeay with irix-cc-n64	`-enable-64bit`
Configured SSLeay with hpux11-64bit-cc	`-enable-64bit`
Globus requires a replacement make for your architecture	`-with-make=command`
Built SSLeay with GCC	`--with-TARGET-CC=gcc*`

* *The option marked with * is a configure option, so it needs to be passed to* `globus-install` *after all* `globus-install` *options and after a double-hyphen "--".*

The `globus-install` script will prompt for information it requires to complete. You can provide additional arguments to the command line to override this behavior.

All command line options to `globus-install` are documented on the web or can be seen by running

```
% <globus-src-dir>/globus-install -help
```

Available Options

The more commonly used options to `globus-install` are listed here. For a complete set of options to the `globus-install` command, see “Appendix B.”

Option	Description	Default (Notes)
`-enable-64bit`	build for 64 bit target on IRIX	n32 libs are built
`-with-make=make-prog`	use specific `make` program	the *MAKE* environment variable is used if set or make is searched for in the user’s path
`-with-umask=octal-umask`	use specific umask permission defaults	existing umask is used if it gives read access or the user is prompted to use or override the default 022
`-prompt`	verify before each major step	None
`-separate-logs`	separate error/output logs	None
`-dryrun`	don’t really build	None
`-help`	print this	None
`-with-threads`	only build threaded libs	
`-with-threads=package`	same as above, set thread package	
`-without-threads`	only build non-threaded libs	by default both non-threaded and threaded libs are built
`-with-domain-name=domain`	supply domain name for this host	useful for hosts in which automatic detection fails
`-without-standard`	skip standard protocol flavor	
`-with-mpi`	force MPI (override detection)	
`-with-mpl`	force MPL (override detection)	

Option	Description	Default (Notes)
`-with-inx`	force INX (override detection)	
`-without-mpi`	ignore MPI (override detection)	
`-without-mpl`	ignore MPL (override detection)	
`-without-inx`	ignore INX (override detection)	by default libs for all applicable message systems are built
`-disable-debug`	build performance (non-debug) libs	by default debug libs are built
`-development-tests`	extra files in development tree	
`-builddirs-persist`	don't clean up build space	These options are intended for the Globus Project team.

The options in the table below are commonly used when *not* building all the components. For example, when building different sets of development libraries the `-disable-services` and `-disable-tools` options should be used.

Option	Description	Default
`-disable-services`	Do not install system services, e.g., resource management, information services, and remote access	None
`-disable-tools`	Do not install user tools, e.g., job management tools, information management tools	None
`-disable-development`	Do not install development files, e.g., libraries that can be linked to user applications	None

Sample Build

Here is an example of how to build the Globus Toolkit on an SGI Origin2000 architecture running IRIX 6.5:

```
% ./globus-install -prefix=/afs/ncsa/packages/globus/globus-1.1.3-install \
    -with-ssl-path=/afs/ncsa/packages/globus/ssl/IRIX_6.5-n32 \
    -with-ldap-path=/afs/ncsa.uiuc.edu/packages/globus/openldap-1.2.7 \
      011400/IRIX_6.5-n32 \
    -with-umask=002 \
    -with-domain-name=ncsa.uiuc.edu \
    -enable-parallel-build \
    -enable-debug \
    -builddirs-persist
```

This example defines the following required arguments:

- Path to the Globus Toolkit installation directory
- Path to the OpenLDAP installation (32-bit mode)
- Path to the SSLeay installation (32-bit mode)

This example defines the following optional arguments:

- umask to apply to files being generated by the build procedure
- Domain name of the host on which the Globus Toolkit is being built
- Allow building using parallel makes
- Build code with debugging option turned on
- Do not remove files from the build directory after installation

The result of this build will be a complete Globus Toolkit built in 32-bit mode with all types of message passing development libraries that are found to be supported on it. In order to build additional development libraries in 64-bit mode, the following command could be executed after the initial build completed

```
% ./globus-install -prefix=/afs/ncsa/packages/globus/globus-1.1.3-install \
  -with-ssl-path=/afs/ncsa/packages/globus/ssl/IRIX_6.5-64 \
  -with-ldap-path=/afs/ncsa.uiuc.edu/packages/globus/openldap-1.2.7 \
    011400/IRIX6.5-64 \
  -with-umask=002 \
  -with-domain-name=ncsa.uiuc.edu \
  -enable-parallel-build \
  -enable-debug \
  -builddirs-persist \
  -disable-services \
  -disable-tools \
  -enable-64bit
```

This build contains the following differences from the first build:

- Different paths to the OpenLDAP and SSLeay directories. These paths point to the 64-bit mode installations of OpenLDAP and SSLeay instead of the 32-bit mode installations.
- The option to build in 64-bit mode
- The option to *not* build the tools and services modules. Only the development libraries are built.

The Globus Toolkit Directory Structure

Once the `globus-install` process has been completed, you will have the directory structure in <globus-install-dir>. There are four basic sets of directories: common directories, services directories, tools directories, and development directories.

The common directories contain the programs and data which are architecture-independent. The services directories contain architecture-specific programs and data that are useful to system administrators. The tools directories contain architecture-specific programs and data which are useful to Globus end-users and developers.

The development directories contain libraries and headers, useful to developers using the Globus Toolkit. Each architecture-specific directory consumes approximately 10 Mbyte of disk space. Multiple architecture-specific development directories may be built by default for any particular architecture. These provide support for different communications protocols and threading models.

The top-level directories of <globus-install-dir> are:

etc/	Globus Toolkit configuration files, including files to configure its gatekeepers, MDS, and GSI for your site
com/	Currently unused
share/	Globus Toolkit architecture-independent data files, including MDS Object specifications and GSI trusted certificates
var/	Globus Toolkit log files, including logs from the MDS and Heartbeat Monitor
tmp/	Globus Toolkit temporary files generated by the various configuration daemon scripts as they run
sbin/	Globus Toolkit system commands, including `globus-local-deploy` and `globus setup` (described below)

A services/<architecture> directory will be created for each architecture on which you run `globus-install`. Each of these directories will contain:

services/<architecture>/globus-software-summary	Globus Software summary. The configuration options used to create this build of the Globus Toolkit.
services/<architecture>/bin	Globus Toolkit services commands. These include the tools required by the system commands to update the MDS and generate security credentials.
services/<architecture>/libexec	Globus Toolkit services internal commands. These are utilities that are used by the commands located in the bin and sbin directories.

services/<architecture>/sbin	Globus Toolkit services system commands. These include the daemons and setup tools used by Globus administrators.

A tools/<architecture> directory, with Globus Toolkit architecture-dependent tools commands will be created for each architecture on which you run `globus-install`. Each of these directories will contain the following:

tools/<architecture>/globus-software-summary	Globus Software Summary. The configuration options used to create this build of the Globus Toolkit.
tools/<architecture>/bin	Globus Toolkit tools commands. These include the user-level tools that provide command-line interfaces to the Globus Toolkit.
tools/<architecture>/libexec	Globus Toolkit tools internal commands. These are commands that are used by the commands located in the bin and sbin directories.
tools/<architecture>/sbin	Globus Toolkit tools system commands. These include the commands to setup the tools in the bin directory.

A development/<architecture-flavor> directory, with Globus Toolkit architecture-dependent libraries, will be created for each architecture and each optional build type (such as, thread model, debug mode, and supported protocol modules). Each will contain the following:

development/<architecture-flavor>/globus-software-summary	Globus software summary. The configuration options used to create this build of the Globus Toolkit.
development/<architecture-flavor>/etc/	Globus development configuration files. These include a makefile_header header and a Makefile needed to configure MPICH-G.
development/<architecture-flavor>/include/	Globus architecture-dependent C headers
development/<architecture-flavor>/tools/	A link to the appropriate architecture/dependent tools directory.

development/<architecture-flavor>/lib/	Globus architecture-dependent C development libraries.

Globus Toolkit Setup

After the build process is done, you may need to set up site-specific parameters. The setup phase of the Globus Toolkit installation process is when site-specific information is defined by 1) modifying configuration files and 2) using the optional `globus-setup` command. This configuration information will be used by Globus Toolkit programs and scripts.

The globus-setup Command

You only need to run `globus-setup` once per install directory. The configuration information is shared for all architectures installed under that directory. The `globus-setup` script opens with instructions for setting up MDS and then takes you to further information about setting up GSI.

Using globus-setup for MDS

The `globus-setup` command supports setup of either the classic or distributed MDS models. It can be used to change the default MDS configuration.

NOTE: If you want to set up the distributed MDS server using only a GRIS and the default resource distinguished name, do *not* run `globus-setup` at all. This choice represents the default configuration for the Globus Toolkit version 1.1.3. Each resource will have its own LDAP information service to which you can connect remotely to obtain system and status information. No site-wide server will be used.

The following tasks can be accomplished using the `globus-setup` command:

Change the Distinguished Name of the Resource. The MDS requires unique names for Globus grid resources. By default, these names are constructed in part by using the resource's DNS components. For example, the resource denali.mcs.anl.gov gets a default name:

```
"hn=denali.mcs.anl.gov, dc=mcs, dc=anl, dc=gov, o=Grid"
```

A site is free to override the default using the `globus-setup` command. This option might be used to allow a site to maintain information about hosts in different DNS domains under a single Directory Information Tree (DIT) suffix.

Define a GIIS. The GIIS represents a centralized MDS server that provides information about all of your resources. By default, a GIIS is not configured. If you do not have a GIIS your system will still be fully functional, but you will not have a centralized server that contains information about all of your resources. Each resource will have its own LDAP information service that can be connected to remotely for obtaining system and status information. The GIIS can only be configured to run on a host on which the Globus Toolkit will be installed.

Select the Classic MDS Model. If you prefer to use the classic MDS model, in which each resource runs a `globus-gram-reporter` daemon and a `grid-info-build-site` daemon that pushes information into a centralized LDAP server, then run the `globus-setup` command with the `-classic` flag. The `globus setup` command needs to be run to specify the appropriate information about the site's LDAP server.

NOTE: When prompted by `globus-setup` to enter the hostname of your site's server, referred to as the MDS or organizational server in the script, you should enter the name of the host running the LDAP server. You will also need to supply the port number for the server.

Using globus-setup for GSI

The `globus-setup` command is also used to configure GSI site-specific parameters. These parameters determine how distinguished names for certificates will be constructed when a user executes the certificate request generation script from the same host on which the Globus Toolkit is installed. After completing the MDS setup portion of the `globus-setup` script, the second part of the script will open and the GSI menu will appear. The defaults that appear in this menu are obtained from the value of the organization's distinguished name as specified in the MDS portion of the script. To accept these defaults, quit the script. You will receive a message that setup is complete.

Managing the Configuration Files

Whether or not you have run `globus-setup`, you may still need to modify certain configuration files to change some default values. The two most common modifications to configuration files are to change the port on which the gatekeeper runs and to add additional services. The gatekeeper supports multiple services and one of the most common services is a job manager, which interfaces to local scheduling systems and, therefore, is indirectly responsible for the execution of a job on local resources. In the default configuration, the gatekeeper is configured to use port 2119/tcp. This default configuration also enables a single job manager service that creates new processes on the managed host using the Unix `fork` operation. On many platforms, however, user processes are created by resource management systems such as Loadleveler, LSF, or NQE. Enabling the Globus Toolkit to use one or more of these systems requires modification to the configuration files.

This section discusses how to make some of the more common modifications to the Globus Toolkit configuration files and points out differences when configuring information services for the classic and the distributed MDS models. If you are happy with the defaults or want to try things out quickly, you can skip this section and come back to it later.

Gatekeepers: globus-gatekeepers.conf

As stated earlier, the default configuration allocates port 2119/tcp for the gatekeeper. To change this value, you will need to edit the gatekeeper configuration file

```
<globus-install-dir>/etc/globus-gatekeepers.conf
```

prior to deploying the Globus Toolkit. This configuration file contains an entry for each host that is to have a gatekeeper deployed on it. Each entry contains three space-delimited fields:

```
<hostname> [inetd | daemon] <port#>
```

Obviously, <hostname> is the name of the host for this particular gatekeeper. The next field, the startup field, can contain either inetd or daemon. If the Globus Toolkit is to be used for multiple users, the gatekeeper requires root privilege to perform tasks on behalf of each user. Therefore, in this case, the startup field should be inetd. An individual user can make use of the Globus Toolkit by running the gatekeeper as a daemon on a nonprivileged port. In this case, the startup field should be daemon. The default gatekeeper configuration file looks something like this:

```
# This file lists all the gatekeepers associated with the current
site.
#
# IF YOU CHANGE THIS FILE YOU _MUST_ RERUN globus-local-deploy ON
# ALL RELEVANT MACHINES
#
# Format:
# hostname startup port
#
# hostname: A fully qualified DNS name on which the gatekeeper
runs.
# startup: Method by which the gatekeeper is started: inetd or
daemon.
# port: The port number on which the gatekeeper listens.
#
pitcairn.mcs.anl.gov inetd 2119
```

Gatekeeper Services: globus-services.conf

If the gatekeeper will manage additional services, they need to be defined. Even if you primarily use a scheduling system to execute jobs on your system, we strongly recommend that you keep a `fork` job manager running on your site. Typically, this is used to create processes on a *service node* of a parallel computer. For example, this service node may be used to run programs that stage data to the parallel computer prior to running a job using the scheduler.

The available job managers are:

Manager Type	**Manager Description**
fork	fork a job on the machine on which the gatekeeper is running
poe	start a job using IBM’s POE on the machine on which the gatekeeper is running
condor	start a job using the Condor high-throughput computing environment
easymcs	start a job on an IBM SP or cluster using the EASY scheduler
nqe	start a job using Cray’s NQE (network queueing environment) on a T3E
prun	start a job on the DAS using PRUN
loadleveler	start a job on an IBM SP using loadleveler
lsf	start a job on MPP or cluster using the Load Sharing Facility
glunix	start a job on a workstation cluster running GLUnix from UC Berkeley
pexec	start a job on an Intel Paragon using pexec
pbs	start a job on MPP or cluster using the Portable Batch System

The services configuration file

<globus-install_dir>/etc/globus-services.conf

contains an entry for each service running on a particular system. Each entry consists of the following fields:

<hostname> <service-name> <options> <uid> <command> [<argv0, argv1, …>]

A sample globus-services.conf configuration file is:

```
# This file lists all the services associated with the current
site.
#
# IF YOU CHANGE THIS FILE YOU _MUST_ RERUN globus-local-deploy ON
# ALL RELEVANT MACHINES
#
# Format:
# hostname service options uid executable argv0 ... argvN
#
# hostname: A fully qualified DNS name on which the gatekeeper
runs.
# service:
# options:
# uid:
# commandpath:
# {argv0, argv1 ...}
#
#! default jobmanager fork
pitcairn.mcs.anl.gov jobmanager stderr_log,local_cred - \
${libexecdir}/globus-jobmanager globus-jobmanager \
-conf ${sysconfdir}/globus-jobmanager.conf -type fork \
-rdn -jobmanager -machine-type cluster
```

To add a new job manager service, possibly to support an additional scheduling system, you would add an additional line containing the required information. (Each job manager must always have a per-machine uniquely defined distinguished name.) For example, for an LSF gatekeeper on the same host, you would add the line:

```
pitcairn.mcs.anl.gov jobmanager-lsf stderr_log,local_cred -\
${libexecdir}/globus-jobmanager globus-jobmanager \
-conf ${sysconfdir}/globus-jobmanager.conf -type lsf \
-rdn jobmanager-lsf -machine-type cluster
```

When you are configuring a Condor service, use the configuration file option

```
-conf ${sysconfdir}/globus-jobmanager-condor.conf
```

instead of the one displayed in the examples above. After deployment, this file (<globus-deploy_dir>/globus-jobmanager-condor.conf) has to be edited before it can be used (add information specific to the Condor flock you intend to use). A sample default globus-jobmanager-condor.conf file is:

```
-home /usr/local/globus1.1.3/globus-deploy-1.1.3
-e  /usr/local/globus1.1.3/globus-deploy-1.1.3/libexec
-globus-org-dn 'dc=ncsa, dc=uiuc, dc=edu, o=Grid'
```

```
-globus-gatekeeper-host 'modi4.ncsa.uiuc.edu'
-globus-gatekeeper-port '2119'
-globus-gatekeeper-subject
'/O=Grid/O=Globus/CN=modi4.ncsa.uiuc.edu'
-globus-host-dn 'hn=modi4.ncsa.uiuc.edu, dc=ncsa, dc=uiuc, dc=edu,
o=Grid'
-globus-host-cputype mips
-globus-host-manufacturer sgi
-globus-host-osname irix
-globus-host-osversion 6.5
# Edit the following two lines to complete
# the configuration of your condor jobmanager
-condor-arch <unknown>
-condor-os   <unknown>
```

The values for the options in the last two lines of this file need to be defined.

Check the <globus-install_dir>/etc <architecture-flavor>/globus-sh-commands.sh file for properly defined local commands pertinent to the scheduling system supporting the newly added service.

If you have multiple deployments of the Globus Toolkit running on the same host, then please read the information contained in the FAQ:

http://www.globus.org/about/faq.html

Daemons: globus-daemons.conf

The appropriate Globus daemons are initiated via the globus-daemons.conf configuration file. For example, the file

<globus-install_dir>/etc/globus-daemons.conf

might contain an entry for a daemon that collects site-specific information. An entry for such a daemon in the globus-daemons.conf file is:

```
#Start up the MDS daemon to collect site specific information
${sbindir}/grid-info-build-site
```

If the distributed MDS model is configured, then local MDS services are started from globus-daemons.conf and are automatically registered. A sample globus-daemons.conf file entry is:

```
grid-info-soft-register -f grid-info-resource-register.conf -- \
slapd -f grid-info-resource.conf -p <num> ...
```

The soft-registration monitor starts the local MDS (slapd) service and registers it with the site MDS service in a fault-tolerant manner.

Information Services: grid-info.conf

In order to properly configure the Globus Toolkit information services component, you may need to modify grid-info.conf. How you modify this file depends on whether or not your site is running the classic MDS model or the distributed MDS model.

Using the Classic MDS Model. The configuration parameters supplied while running `globus-setup` when the classic MDS model is chosen are stored in the file

<globus-install_dir>/etc/grid-info.conf

These parameters define MDS information, such as the host and port running an LDAP MDS directory server. A sample grid-info.conf file is:

```
#################################################################
#
# File: grid-info.conf
#
# Purpose: This file contains the configuration information
#          for accessing the MDS
#
#          This script is sourced by the mds-build procedure
#          to obtain site specific information
#
#          This information must be consist with information
#          stored at the Globus MDS site
#
#################################################################

# Information about the Root MDS server
GRID_INFO_HOST="mds-alliance.globus.org"
GRID_INFO_PORT="391"


# How long should a connection be held prior to assuming
# that the server is not functioning correctly. Time is in
# seconds
GRID_INFO_TIMEOUT="30"

# Root Distinguish Name (DN) handled by the Root MDS Server
GRID_INFO_BASEDN="o=Globus, c=US"


# Distinguish Name (DN) of the Organization
#   which may be the Root DN of the Local MDS Server
GRID_INFO_ORGANIZATION_DN="ou=The National Center for
Supercomputing Applications, o=The University of Illinois Urbana-
Champaign, $GRID_INFO_BASEDN"


# The Distinguish Name (DN) of the Administrator
GRID_INFO_ORGANIZATION_ADMIN_DN="cn=Directory Manager,
$GRID_INFO_ORGANIZATION_DN"


export GRID_INFO_ORGANIZATION_DN
export GRID_INFO_ORGANIZATION_ADMIN_DN
export GRID_INFO_TIMEOUT
```

Using the Distributed MDS Model. The configuration parameters supplied while running `globus-setup` when the distributed MDS model is used are stored in the file

<globus-install_dir>/etc/grid-info.conf

These parameters define MDS information, such as the host and port information for the site GIIS. A sample grid-info.conf file is:

```
#################################################################
#
# File: grid-info.conf
#
# Purpose: This file contains the configuration information
#          for accessing the MDS
#
#          This script is sourced by the mds-build procedure
#          to obtain site specific information
#
#          This information must be consist with information
#          stored at the Globus MDS site
#
#################################################################


# These values are  set by globus-setup

SETUP_GRID_INFO_MODEL="MDS_SITE_INDEX"
SETUP_GRID_INFO_HOST="hincapie.ncsa.uiuc.edu"
SETUP_GRID_INFO_PORT="2000"
SETUP_GRID_INFO_BASEDN=""
SETUP_GRID_INFO_ORGANIZATION_DN="dc=ncsa, dc=uiuc, dc=edu, o=Grid"
SETUP_GRID_INFO_ORGANIZATION_ADMIN_DN=""


if [ -z "${GRID_INFO_MODEL}" ] ; then
  GRID_INFO_MODEL="${SETUP_GRID_INFO_MODEL:-MDS_SITE_INDEX}"
fi


_base_dn="‘${bindir}/globus-domainname | \
         ${GLOBUS_SH_SED-sed} -e ’s/^/dc=/’ -e ’s/\./, dc=/g’‘"

case "${GRID_INFO_MODEL}" in

  MDS_CLASSIC)
    DEFAULT_GRID_INFO_HOST="mds11.globus.org"
    DEFAULT_GRID_INFO_PORT="391"
    DEFAULT_GRID_INFO_BASEDN="o=Globus, c=US"
    DEFAULT_GRID_INFO_ORGANIZATION_DN="${_base_dn},
${DEFAULT_GRID_INFO_BASEDN}"
    DEFAULT_GRID_INFO_ORGANIZATION_ADMIN_DN="cn=Directory Manager,
${DEFAULT_GRID_INFO_ORGANIZATION_DN}"
    ;;

  MDS_SITE_INDEX)
DEFAULT_GRID_INFO_HOST="modi4.ncsa.uiuc.edu"
    DEFAULT_GRID_INFO_PORT=""
    DEFAULT_GRID_INFO_BASEDN="o=Grid"
    DEFAULT_GRID_INFO_ORGANIZATION_DN="${_base_dn},
${DEFAULT_GRID_INFO_BASEDN}"
    DEFAULT_GRID_INFO_ORGANIZATION_ADMIN_DN=""
    ;;
    *)
     DEFAULT_GRID_INFO_HOST=""
     DEFAULT_GRID_INFO_PORT=""
     DEFAULT_GRID_INFO_BASEDN=""
     DEFAULT_GRID_INFO_ORGANIZATION_DN=""
     DEFAULT_GRID_INFO_ORGANIZATION_ADMIN_DN=""

esac



# Now set the actual values:


# Information about the Organizational MDS server
GRID_INFO_HOST="${SETUP_GRID_INFO_HOST:-
${DEFAULT_GRID_INFO_HOST}}"
GRID_INFO_PORT="${SETUP_GRID_INFO_PORT:-
${DEFAULT_GRID_INFO_PORT}}"

# How long should a connection be held prior to assuming that
# the server is not functioning correctly. Time is in seconds
GRID_INFO_TIMEOUT="30"

# Suffix for default Namespace
GRID_INFO_BASEDN="${SETUP_GRID_INFO_BASEDN:-
${DEFAULT_GRID_INFO_BASEDN}}"

# Site's Distinguish Name (DN) within the defined Namespace
GRID_INFO_ORGANIZATION_DN="${SETUP_GRID_INFO_ORGANIZATION_DN:-
${DEFAULT_GRID_INFO_ORGANIZATION_DN}}"

# Site's Distinguish Name (DN) of the Directory Manager
GRID_INFO_ORGANIZATION_ADMIN_DN="${SETUP_GRID_INFO_ORGANIZATION_AD
MIN_DN:-${DEFAULT_GRID_INFO_ORGANIZATION_ADMIN_DN}}"


export GRID_INFO_HOST
export GRID_INFO_PORT
export GRID_INFO_TIMEOUT
export GRID_INFO_ORGANIZATION_DN
export GRID_INFO_ORGANIZATION_ADMIN_DN
```

Resource Information: grid-info-resource.conf

Local resource information services provided by the GRIS generate data based on information within the grid-info-resource.conf file. The format of this file is: 1) a field for the Time to Live (TTL) values, 2) a field for the underlying command or utility for which the TTL values are being defined, and 3) a field for any arguments to the utility. A sample file looks like:

```
# TTL cmd [arg...]
600 grid-info-host
600 globus-version -ldif
30  globus-gram-scheduler
```

Each of the three commands listed in this file are Globus Toolkit support utilities. The TTL values, measured in seconds, may be adjusted to allow you to determine the level of service for local resource information, depending on the demand for such information for a particular resource.

Site Information: grid-info-site.conf configuration file

The site-wide MDS service grid-info-site.conf file is only used on machines on which a GIIS is running. If the GIIS was configured using the `globus-setup` command, the file specifies some TTL values and can be used to provide some form of access control as described in the comments of the file. A sample grid-info-site.conf file is:

```
#
# Entry w/ non-standard LDIF DN="conf" sets defaults
# and allows cachedir and regdir.
#
# LDIF entries with pattern-matched DNs (shell-style
# pattern matches) set values selectively, not including
# cachedir and regdir.
#
# Matches in order of appearance, so put most specific
# patterns first.
#
#  shell variable references are expanded prior to pattern
#  matching.
#
#  '*' can match any substring
#  '?' can match any character
#  all literals must be matched
#
# matched registration DNs use the parameters provided.
# non-matched registration are rejected.
#
dn: conf
regdir: ${localstatedir}/grid-info-site-regdir
cachedir: ${localstatedir}/grid-info-site-cache
defaultttl: 300
maxttl: 2880
minttl: 120
defaulttimelimit: 10
maxtimelimit: 300
mintimelimit: 5

# this entry example adds access-control only.
# it uses all the global configuration data set above,
# but only matches host registrations from the local org.
# one could refine it by adding domain-name suffix to
# the hostname pattern, e.g. "hn=*.local.org, service=..."
dn: service=MDS Resource, hn=*, service=MDS Registration, \
${GRID_INFO_ORGANIZATION_DN}
```

The internal settings override the compiled-in defaults. The registration constraints provide defaults or override values provided by remote registrants when necessary.

Resource Registration: grid-info-resource-register.conf configuration file

A GRIS service is deployed on each machine when the distributed MDS model is used. This service responds to queries from other systems on the Globus grid asking for information about the local machine. The GRIS service can be configured to register itself with centralized services (such as a GIIS) so that those services can pass on information about the machine to others. This feature is automatically configured when a GIIS is specified when `globus-setup` is run during the installation process.

```
#
# Each LDIF record describes one registration target.
 # May have zero or more.
#
# Currently supported "MDSreg" format:
#
#   dn: <LDAP add object DN>
#   regtype: mdsreg
#   reghn: <host to send reg to>
#   regport: <port to send reg to>
#   regperiod: <how often to send reg (seconds)>
#   [service attribute/value]...
#
# where service object entries depend on the type of
# LDAP objct being published.  For Globus 1.1.3 "MDS
# Resource" objects, the attributes are:
#
#   type: ldap
#   hn: <host of service being registered>
#   port: <port of service being registered>
#   rootdn: <DN suffix of service being registered>
#   ttl: <normally twice the value of regperiod>
#   timeout: 60
#   mode: cachedump
#   cachettl: 30
#
#
# local Globus site index
# deployed at ${GRID_INFO_HOST}:${GRID_INFO_PORT}...
# every 5 minutes with a 10 minute TTL on registration info
#
dn: service=MDS Resource, hn=resource.host.name, service=MDS \
Registration, ${GRID_INFO_ORGANIZATION_DN}
regtype: mdsreg
reghn: ${GRID_INFO_HOST}
regport: ${GRID_INFO_PORT}
regperiod: 300
type: ldap
hn: resource.host.name
port: 2135
rootdn: hn=resource.host.name, ${GRID_INFO_ORGANIZATION_DN}
ttl: 600
timeout: 60
mode: cachedump
cachettl: 30
```

```
#
# global GUSTO index listening at mds.isi.edu:980...
# every 24 hours with a 48 hour TTL on registration info
#
dn: service=MDS Resource, hn=resource.host.name, service=MDS \
Registration, ${GRID_INFO_ORGANIZATION_DN}
regtype: mdsreg
reghn: mds.isi.edu
regport: 980
regperiod: 86400
type: ldap
hn: crater.isi.edu
port: 2135
rootdn: hn=resource.host.name, ${GRID_INFO_ORGANIZATION_DN}
ttl: 172800
timeout: 60
mode: cachedump
cachettl: 30
```

Globus Toolkit Deployment

The deployment phase of installing the Globus Toolkit is when you supply configuration information for setting up the remaining Globus Toolkit file structure and other helper programs. See also “Managing the Configuration Files” in Chapter 5 on page 29. The deployment phase is also when daemons and their configuration files are copied to local disk, to avoid running off a remote filesystem.

Once you have configured the Globus Toolkit gatekeepers and the services associated with those gatekeepers, you are ready to deploy the Globus Toolkit. It must be deployed on each host for which you configured a gatekeeper.

The deployment phase copies a subset of the architecture-dependent commands in the services directory to a local directory. It then creates configuration files to start the Globus Toolkit daemons. These daemons support the gatekeeper and service configurations you have made in the setup phase.

Each deploy directory should be on a filesystem that is not NFS mounted. This is because some files require root read-only access and some executables are setuid root. Also, the daemons may not perform correctly if the files they require are not available.

When running the `globus-local-deploy` command, you will be asked if you wish to register your site with the Globus Project. This registration is for information purposes only. If you answer “yes” to this question, your site will be included in the Globus Project list of installed sites. We ask sites to register with us so that we may better understand the size and composition of our user community and provide better service to you in the future. Your answer to this question does not affect the way your system operates in any way.

When `globus-local-deploy` is run on the configured GIIS host, it will automatically deploy the additional GIIS components. When `globus-local-deploy` is run on other hosts, it will configure the GRIS service to dynamically register with the selected GIIS server in a fault-tolerant manner.

First-Time Deployment

As root, you should create a deployment directory <deploy-dir> on each host on which you will be deploying the Globus Toolkit. We recommend something like /opt/globus, but it can be anything.

Change the ownership of this directory to user globus.

As user globus:

```
% setenv GLOBUS_INSTALL_PATH <globus-install-dir>
% <globus-install-dir>/sbin/globus-local-deploy
```

When `globus-local-deploy` has completed, a set of instructions will be displayed. (A copy of these instructions is also written to the file <globus-deploy-dir>/tmp/globus-root-instructions.) These instructions include how to:

- Request a certificate for the gatekeeper
- Modify root services and file permissions
- Set proper file ownership and permissions (/opt/global/etc/globus-gatekeeper.key permissions must be 600 or 400; 400 is recommended)
- Reactivate inetd
- Install system startup and shutdown scripts
- Start up and shut down the system
- Test the installation (see "Verifying the Installation" below)

Performing the tasks for these instructions completes the Globus Toolkit deployment phase.

Previous Deployment

If you have previously deployed the Globus Toolkit, you should follow these instructions. If you have already deployed the Globus Toolkit at your site, you cannot simply redeploy a new version of the Globus Toolkit in the same directory as the previous deployment. There are two reasons for this:

- Some of the files in <deploy-dir> are owned by root.
- The contents and names of some files within <deploy-dir> are modified to reflect changes in the installation procedure.

You may reuse the certificate/key pair from the previous 1.1.x installation. To use a previous key pair, you can add the `-cert <certfile>` and `-key <keyfile>` arguments to the `globus-local-deploy` command. Before doing so, you should ensure that these files are readable by the user performing the deployment step.

NOTE: Under no circumstances should you deploy into an already existing deployment directory. If you decide to remove your previous deploy directory, you should first make a copy in a safe place of the keyfile, certfile, and grid-mapfile files you wish to reuse. If you do not wish to reuse an existing key pair, you can simply leave off the `-cert` and `-key` arguments, and a new certificate request will be generated for you.

Verifying the Installation

The `globus-setup-test` command can be used to verify that the Globus Toolkit was properly configured and that the Globus Toolkit gatekeeper is working properly. The test should not be run as root and must be run by a user id that has a certificate. The test confirms the functionality of communication and authentication with the gatekeeper. This command can also be used to test communication with and retrieving information from the MDS by using the `-h <host>` command line option.

The `globus-setup-test` will prompt for the pass-phrase and after successful authentication continue all the remaining tests. Information about the tests being conducted and the results of the tests are displayed to the screen from which `globus-setup-test` was run.

Example output of a successful test is:

```
modi4 186% globus-setup-test
Checking certificate directory ..........done.
Checking user certificate setup .........done.
Checking user key setup .................done.
Creating proxy certificate ..............
.Enter PEM pass phrase:
verify OK
....+++++
.....................+++++
.........................................done.
Checking user proxy setup ...............done.
Searching MDS for contact strings .......done
Testing "jobmanager" service  on modi4.ncsa.uiuc.edu
Authentication test .....................Success!
Submission test .........................Success!

Testing "jobmanager-lsf" service  on modi4.ncsa.uiuc.edu
Authentication test .....................Success!
Submission test .........................Success!

Testing Completed!
```

User Environment Setup

This section addresses what Globus Toolkit users need to know about their working environment, what you as an administrator need to know about system and user startup files, and how users obtain and manage certificates.

Environment Variables

The tools in the Globus Toolkit depend on two environment variables that identify the root of the Globus Toolkit installation directory and the complete path to the appropriate Globus tools binaries. These variables are:

`GLOBUS_INSTALL_PATH` environment variable

Path to the root of the Globus installation directory

`GLOBUS_PATH` environment variable

Path to the appropriate binaries
(i.e., <globus-install-path>/tools/<architecture-flavor>/bin)

These variables must be properly set in each user's environment. Once these variables are set the `GLOBUS_PATH` variable should be added to the user's `PATH` variable. The addition of the `GLOBUS_PATH` variable to the `PATH` variable ensures that the appropriate Globus tools will be used.

System and User Startup Files

Two environment variables used by the Globus Toolkit can be set for users during the login process in /etc/cshrc and /etc/profile. Because these files are executed during the login process the Globus Toolkit environment variables can be set for each user at this time.

A script to detect the appropriate values for these variables is provided as part of the Globus Toolkit. The system startup files can be used to invoke this script to set the user's Globus environment variables. For example, in the /etc/cshrc file the following lines could be added:

```
# Set path to Globus installation
setenv GLOBUS_INSTALL_PATH <globus_install_path>
if ( -r <GLOBUS_INSTALL_PATH>/etc/globus-user-setup.csh) then
        source <GLOBUS_INSTALL_PATH>/etc/globus-user-setup.csh
endif
```

In the /etc/profile file the following lines could be added:

```
# Set path to Globus installation
GLOBUS_INSTALL_PATH=<globus_install_path>
export GLOBUS_INSTALL_PATH
if [ -r <globus_install_path>/etc/globus-user-setup.sh ]
  then
   .  <globus_install_path>/etc/globus-user-setup.sh
fi
```

where `<globus_install_path>` represents the path to the Globus installation as defined during the Globus Toolkit installation process.

The user's *PATH* variable is set to include *GLOBUS_PATH* at this time. However, because users' local startup files are executed after system startup the *PATH* variable can be overwritten in their local startup files. Users should be notified of this possibility, therefore, to ensure that *GLOBUS_PATH* is in their *PATH* variable in their local startup script.

Requesting a Certificate

The Globus Toolkit uses SSLeay-based authentication that requires the use of an X.509 certificate. Therefore, in order to use the Globus Toolkit each user must obtain a certificate. The `grid-cert-request` program is used to request a certificate from the Globus CA. The `grid-cert-request` program uses information found in the Globus Toolkit installation tree, as well as user-supplied information, to form a request that should be emailed to the certificate authority. The `grid-cert-request` program prompts for all the needed information and provides instructions on how to complete the certificate request process.

Once the certificate is received from the CA, it is the user's responsibility to place the certificate in the appropriate location as instructed in the email reply from the CA.

NOTE: Your site may have or may want to establish its own method for obtaining a certificate. If that is the case, you will need to document this method locally.

Maintaining Certificates

Once a certificate has been obtained the certificate must reside in an appropriate location on an appropriate resource. By default, during the authentication process to a Globus Toolkit service, users' certificates are expected to be located in their home directories in a subdirectory called ".globus". Instructions about the placement of the certificate in such a directory are provided with the return of the certificate. The user's certificate should reside on the resource from which Globus Toolkit services will normally be requested. For security reasons, copies of certificates should not be kept on multiple resources.

Information about a certificate can be obtained by using the `grid-cert-info` program. The type of information that can be extracted from a certificate include the user's distinguished name, common name, and certificate start/end date.

Each certificate has an expiration date associated with it. A short time before a certificate is about to expire the owner is sent email suggesting that a certificate renewal request be made. This email will contain a text string referred to as the *challenge text*. The email will include instructions on how to request a renewal using the `grid-cert-renew` tool.

Other Installations

Now that you are done installing the Globus Toolkit, you may install the following related software packages:

ssh and ftp with GSI modifications–let you use your Grid credentials for remote login and high-speed file transfer

MPICH-G–a free installation of MPI that uses the Globus Toolkit to facilitate heterogeneous distributed computing

Chapter 6 Managing Globus

The Globus Toolkit has been designed in such a way that once properly installed, there is relatively little day-to-day maintenance. This chapter describes the tools available to you for managing the Globus Toolkit.

There are some differences if your system is only used for submitting Globus Toolkit jobs versus being used as a resource by other systems on the Globus grid. These differences are noted below.

Maintaining the ca-signing-policy.conf file

The ca-signing-policy.conf file specifies information about which certificates will be accepted for authentication. Modifying this file allows you to define a security policy for your site. The policy can be described by an Access Control List (ACL) mechanism, where for each resource, a list of valid entries is granted a set of access rights. The same policy can be implemented using a capability mechanism. Extended Access Control Lists (EACLs) extend the conventional ACL concept by allowing specification of conditional authorization policies, implemented as conditions on authentication and authorization credentials. An EACL is associated with an object and lists principals that are allowed to access this object and the type of access that is granted.

With release 1.1.3, the form in which the Globus CA generates certificates has changed. While you will not need to change the ca-signing-policy.conf file regularly, if you are not running Globus 1.1.3, you should edit it to modify the Globus Certificate Authority EACL entry so that new style Globus certificates will be recognized. Without this change, authentication for job submission requests using the new style certificates will fail. In Globus 1.1.3, the ca-signing-policy.conf file is created with the appropriate Globus Certificate Authority EACL. If you are not running Globus 1.1.3, the file does not contain the appropriate cond_subjects value associated with the new style certificates.

The ca-signing-policy.conf file is located in

```
$GLOBUS_INSTALL_PATH/share/certificates
```

For example, if you are running Globus 1.1.2, your EACL entries may have the format:

```
# Format:
#-------------------------------------------------------------------------
#  token type  | def.authority |                value
#--------------|---------------|-----------------------------------------

# EACL entry #1|
 access_id_CA      X509       '/C=US/O=Globus/CN=Globus Certification Authority'
 pos_rights        globus     CA:sign
 cond_subjects     globus     '"/C=us/O=Globus/*" "/C=US/O=Globus/*"'
# NCSA CA
 access_id_CA      X509       '/C=US/O=National Computational Science
Alliance/OU=Certification Authority'
 pos_rights        globus     CA:sign
 cond_subjects     globus     '/C=US/O=National Computational Science Alliance/*'
# end of EACL
```

In this example, to ensure recognition of all old and new style Globus certificates, you need to change the EACL entry #1 so the string “/O=Grid/O=Globus” is added to the list of cond_subjects:

```
# EACL entry #1|
 access_id_CA      X509       '/C=US/O=Globus/CN=Globus Certification Authority'
 pos_rights        globus     CA:sign
cond_subjects      globus     '"/C=us/O=Globus/*" "/C=US/O=Globus/*""/O=Grid/O=Globus/*"'
# end of EACL
```

Maintaining grid-mapfile

The system administrator is responsible for the maintenance of grid-mapfile. This file contains the information needed by the Globus gatekeeper to map a request for Globus Toolkit services from a user with a particular global subject name to a local user login name or account on that system. The file contains records consisting of user distinguished names and the local login name or account that should be used on that system. If a match exists, then the requested Globus Toolkit service is provided and will be invoked under the appropriate user login name or account as determined in grid-mapfile.

Three tools exist to facilitate the maintenance of this file.

- `grid-mapfile-add-entry`
  Add an entry to the grid map file
- `grid-mapfile-delete-entry`
  Delete an entry from the grid map file
- `grid-mapfile-check-consistency`
  Check the grid map file for any inconsistencies.

Adding a Globus Toolkit User

For users to be able to submit jobs to a resource managed by the Globus Toolkit, they need to be entered into the grid-mapfile ACL. This file is <deploy-dir>/etc/grid-mapfile. Its purpose is to map a Globus Toolkit credential to a local user's login name. The Globus Toolkit administrator on the site can map the holder of any Globus Toolkit credential to any local user name. It is up to the Globus Toolkit administrator to verify that the user's Globus Toolkit identity matches the user's local user name.

The grid-mapfile file is a plain-text file, containing a quoted Globus credential name (the subject of an X.509 certificate) and an unquoted local user name. Each subject name in grid-mapfile must be listed only once. However, multiple identities may map to a shared local name. It is up to the Globus administrator to ensure that grid-mapfile entries do not violate any site security policies.

You must supply the distinguished name exactly as it appears in the certificate. Not doing so will result in an authentication failure. You can get this information directly from the certificate by using the following command:

```
% grid-cert-info [-file <certificate file>] -subject
```

For example, the command

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

will print the certificate subject for the X.509 certificate in the file ~/.globus/usercert.pem, the default location for a user's certificate file.

The utility `grid-mapfile-add-entry` may prove to be helpful to add entries to mapfiles in a correct manner.

Log Files

If problems occur, the log files are a source for determining the cause. A set of log files are maintained in the <globus_deploy_path>/var/ directory.

- globus-gatekeeper.log

  This log file contains information about the Globus Toolkit gatekeeper, specifically the gatekeeper connection activity. The log file is usually most useful for debugging Globus Toolkit gatekeeper problems, including user authentication and job submission.

  Examine this log for information on problems associated with gatekeeper connectivity.

- globus-system.log

  This log file contains information about the Globus Toolkit daemons. Messages are written when daemons are started and stopped and if there are any errors associated with these actions.

  Examine this log for information on Globus Toolkit daemon status.

- globus-daemons.log

  This log file contains information about any errors from any of the Globus Toolkit daemons.

- globus-gram-reporter.log (classic MDS model only)

  This log file contains information about the `globus-gram-reporter` script transactions. These transactions are updates to the MDS with information about the host environment including data such as CPU load, number of available processors, and queuing system information.

Examine this log for information on the status of `globus-gram-reporter` updates to the MDS.

- globus-info-system.log

  This log contains information about errors in MDS transactions.

Starting Up and Shutting Down Globus

The Globus Toolkit provides two scripts to gracefully start up and shut down itself. These scripts can be run manually at any time or can be executed automatically during routine system startup and shutdown times. These scripts are intended to be placed in the appropriate system directories along with other scripts that perform similar tasks for other system services. The scripts can be found in

<globus-deploy-directory>/sbin/S*XX*globus

<globus-deploy-directory>/sbin/K*XX*globus

The scripts can be copied to the appropriate system directory, such as:

- /etc/rc.d/rc5.d/S[*nn*]globus (under RedHat)
- /etc/rc3.d/S[*nn*]globus (under Solaris)
- /sbin/rc3.d/S[*nn*]globus (under HPUX)

where *nn* is a number between 00 and 99.

The `SXXglobus` script starts up the Globus Toolkit when invoked with a `start` argument. Thus, to properly execute this script, make a call to this script from the system startup script using `start` as the argument. For example, under BSD add an entry that calls `SXXglobus start` in the /etc/rc.local file.

Similarly, to properly execute the `KXXglobus` script make a call to this script from the appropriate system startup script using `stop` as the argument.

Tasks Requiring Root Privilege

Most tasks associated with the management of the Globus Toolkit do not need to be performed as root but rather can be performed as the user id used to run Globus Toolkit daemons, presuming the appropriate permissions have been set on the Globus Toolkit files.

The few tasks that do require root privilege include:

- Maintaining the /etc/services and /etc/inetd/conf file entries for the Globus Toolkit
- Managing the gatekeeper certificate
- Changing ownership or permissions on Globus Toolkit files
- Managing Globus Toolkit files specifically owned by root

Troubleshooting

To find specific information about common problems you might run into when installing or administering the Globus Toolkit check out the current list of frequently asked questions. The FAQ is at http://www.globus.org/about/faq.html.

Appendix A Commands

This appendix has descriptions of the commands available in the Globus Toolkit. The commands are grouped together by key areas of Globus Toolkit functionality. Any special usage or administration-specific information are noted in the descriptions.

To get help, on your Unix command line, enter a command followed by the `-usage` or `-help` option. You will find more further information on the commands at http://www.globus.org/v1.1/programs.

Security Commands

Command	Description
`grid-cert-request`	Creates a new certificate request and private key.
`grid-cert-info`	Displays certificate information. Unless the optional -file argument is given, the default location of the file containing the certificate is assumed to be the $X509_USER_CERT. If X509_USER_CERT not set, $HOME/.globus/usercert.pem
`grid-cert-renew`	Creates a new key and renewal request for a Globus certificate. The Globus Certificate Authority (CA) will notify a user when the user's certificate is about to expire. The notification message also contains a challenge (a text string) that grid-cert-renew will embed in the renewal request, for higher security.
`grid-change-pass-phrase`	Changes the "pass phrase" that protects your private key. If the -file argument is not given, the default location of the file containing the private key is assumed to be $X509_USER_KEY If X509_USER_KEY not set, $HOME/.globus/userkey.pem
`grid-check-ca-policy-file`	Obtains policy file (EACL) from the default location, parses it, builds the GAA internal policy structure and prints it out. Reports encountered parse errors.
`grid-check-ca-sig`	grid-check-ca-sig <signer> <subject> Obtains policy file (EACL) from the default location, parses it, builds the GAA internal policy structure and checks if the specified signer is allowed to sign certificates for specified subject.

Command	Description
`grid-inquire-policy-info`	grid-inquire-policy-info <signer>. obtains policy file (EACL) from the default location, parses it, builds the GAA internal policy structure, finds EACL entry corresponding to the signer and returns a list of authorized rights and corresponding conditions, if any.
`grid-proxy-init`	Creates a proxy certificate that can be used for authentication without having to enter the protecting pass-phrase.
`grid-proxy-info`	Displays proxy certificate information. Common name of the subject (/C=US/O=Globus/CN=user1); common name of the proxy generator; proxy information: (limited of full proxy); strength of proxy (number of bits used in the key); and remaining time. It can also work in another mode, verifying that it is a valid proxy, which means that the following requirements are met: the proxy exists, the proxy is not a limited proxy, the proxy has not expired and is active for another H hours, and the proxy is encrypted using at least B bits Unless the optional -file argument is given, the default location of the file containing the proxy certificate is assumed to be the $X509_USER_PROXY. If X509_USER_PROXY not set, /tmp/x509up_uXXX, where XXX is the local UID#.
`grid-proxy-destroy`	Removes any user proxy certificates generated using grid-proxy-init. It can also be used as a general-purpose "safe-remove" program, in that it opens the file and replaces the information contained therein with garbage before deleting it. If no arguments are given, the proxy file at the default location (or at the location pointed to by env.var X509_USER_PROXY, if set) will be destroyed.
`grid-security-config`	This script will ask some questions about site specific information. This information is used to correctly generate the Grid Security Infrastructure for your site.

Job Submission Commands

Command	Description
`globusrun`	Run a single executable on a remote site. The job startup is done using the GRAM or DUROC Globus services. Also, the GASS service can be used to provide access to remote files and for redirecting standard output streams. In addition to starting jobs, `globusrun` can be used to list previously started jobs or do authentication tests to GRAM gatekeepers.

Command	Description
`globus-setup-test`	Verifies credentials setup. Also verifies whether the user has submission capabilities to a certain gatekeeper. If no arguments are given, all gatekeepers on the local host will be tested.
`globus-job-cancel`	Cancels a job previously started using `globus-job-submit`.
`globus-job-run`	Allows the user to run a job at one or several remote resources. It translates the program arguments to a RSL request and uses `globusrun` to submit the job.
`globus-job-clean`	Kills the job if it is still running and cleans the information concerning the job.
`globus-job-get-output`	For the job specified, gets the standard output or standard error resulting of the job execution.
`globus-job-status`	Display the status of the job. See also globus-get-output to check the standard output or standard error of your job.
`globus-job-submit`	For batch job submission (i.e., submitting a job to a queue via some local scheduling manager). Allows the user to submit a job to a remote resources, using the same command-line syntax as `globus-job-run` does. The program translates the program arguments into an RSL request and uses `globusrun` to submit the job in batch submission mode, that is, after the job is submitted, no connection exists between the local host and the remote host. `globus-job-submit` prints out a job id after successful submission of the job to a remote resource. Unless stdout/stderr are specified, the program output will be buffered at the remote site and can be retrieved at any time with `globus-job-get-output`. If the output is buffered, the user must make sure to run `globus-job-clean` when the program output is no longer needed.

Information Services Commands

Command	Description
`grid-info-add` 1.1.2 only	Modifies the GIS server based on the contents of input file. adds one or several objects to the MDS, according to the LDIF entries in the file pointed by the -file argument. If the -file argument is omitted, the program assumes this information is available on stdin. To be able to store information in the MDS, the password of the Directory Manager for the local organization must be provided. If the -password argument is not given, the password will be asked for interactively by grid-info-add.
`grid-info-create` 1.1.2 only	Modifies the MDS server based on the contents of the input file.
`grid-info-gen-filter-template` 1.1.2 only	To provide a bootstrapping of scripts that all follow the same convention. And to remove the burden from the developer by inserting the appropriate code to perform input validation and to guarantee output conformance.
`grid-info-host` 1.1.2 only NOTE: this tool doesn’t even support any of its options	Creates a set of CLDIF entries for the host ‘${bindir}/globus-hostname‘ where options are: -usage (Displays this message); -version (Displays the current version number); -f[ile] hostfile (Creates only CLDIF entries for host listed in hostfile); -all (Creates CLDIF entries for all hosts at the site)…
`grid-info-host-search` 1.1.3	Queries a GRIS on a specified host and port. By default the local host GRIS and associated default GRIS port of 2135 are used.
`grid-info-interfaces` 1.1.2 only	For the current host, build the associated GlobusNetworkInterface and GlobusNetworkInterfaceImage templates.
`grid-info-modify` 1.1.2 only	Update an existing database entry. Metadata is inserted by `grid-info-update`.
`grid-info-networks` 1.1.2 only	For the current site, build the associated GlobusNetwork and GlobusNetworkImage templates. For the current site, build the associated network objects based on the built computational resource nodes for the site.

Command	Description

`grid-info-prep`
 	
`1.1.2 only`
	To convert CLDIF to LDIF with simple format and error checking. This script is directly tied to the format and structure of a "commented LDIF" template file.

`grid-info-remove`
 	
`1.1.2 only`
	See grid-info-add.
	
	deletes one or more objects to the MDS, according to the LDIF entries in the file pointed by by the -file argument. If the -file argument is omitted, the program assumes this information is available on stdin. To be able to delete information from the MDS, the password of the Directory Manager for the local organization must be provided. If the -password argument is not given, the password will be asked for interactively by `grid-info-remove`.

`grid-info-search`
 	
`1.1.2`
	Searches the GIIS.
	
	Sends one or more queries to the GIIS and displays the result on stdout. The query QUERY is a RFC-1558 compliant LDAP search filter. By default, the object's name and all its attributes are displayed: this can be narrowed down by specifying what attributes to display after the query.

`grid-info-site`
 	
`1.1.2 only`
	Usage: $PROGRAM_NAME [hostname | -f[ile] hostfile]. Creates a set of CLDIF entries for the host \"hostname\, where options are: -usage (displays this message); -version (displays the current version number); -f[ile] hostfile (creates only CLDIF entries for host listed in hostfile).The hostfile contains a list (one per line) of host names and an optional location of the Globus bin directory. The default for this directory is $bindir, e.g., host-dns-name [bindir]

`grid-info-update`
 	
`1.1.2 only`
	See grid-info-add.
	
	modifies one or more objects to the MDS, according to the LDIF entries in the file pointed by the -file argument. If the -file argument is omitted, the program assumes this information is available on stdin. To be able to delete information from the MDS, the password of the Directory Manager for the local organization must be provided. If the -password argument is not given, the password will be asked for interactively by `grid-info-remove`.

`grid-mapfile-add-entry`
 	
`both 1.1.2 and 1.1.3`
	can be used to add entries to the mapfile.
	
	For example the following command would add a user to the mapfile: grid-mapfile-add-entry -dn "/C=US/O=Globus/O=State University/CN=Joe User" -ln juser. You must type in the distinguished name exactly as it appears in the certificate. Not doing so will result in an authentication failure.

`grid-mapfile-check-consistency`
 	
`both 1.1.2 and 1.1.3`
	checks the consistency of the Grid mapfile.
	
	Options:-help, -usage (displays help); -version (displays version); -mapfile FILE, -f FILE Path of gridmap to be used.

Command	Description
`grid-mapfile-delete-entry` both 1.1.2 and 1.1.3	deletes an entry from the Grid mapfile. Options: -help, -usage (displays help); -version (displays version); -dn <DN> Distinguished Name (DN) to add; -ln <local name> Local Login Name (LN) to map DN to; -dryrun, -d (shows what would be done but will not delete the entry; -mapfile file, -f file (path of gridmap file to be used)

Other Tool Commands

Command	Description
`config-guess`	Program provided under GNU license from Free Software foundation that attempts to guess a canonical system name.
`globus-gass-server`	A utility that allows the user to start up a stand-alone GASS server, to which he can upload or download files from locally accessible filesystems. When no "Enable" or "Disable" options are specified, the globus-gass-server runs with the default options of -l , -t -u, -r, -w, -o, and -e. By default, the globus_gass_server will output the base URL it is listening on and then remain in the foreground.
`globus-gass-server-shutdown`	This command allows the user to shut down a previously started (remote) GASS server, *but* the GASS server must have been started with the -c command line option (client-destroy) in order for this command to take effect.
`globus-hostname`	This is a simple shell script that acts like the Unix hostname. Returns the system hostname and make some additional checks to ensure a fully qualified hostname. Setting the environment variable GLOBUS_HOSTNAME to a non-null string will cause `globus-hostname` to return that value instead. This is useful for specifying the use of certain network interfaces when communicating etc.
`globus-hostname2contacts`	Converts a hostname to a list of resource manager contact strings. Returns contact strings on stdout, identifying services that provided by gatekeeper(s) running on the host specifed. If no type or service is specified, the program will return matching GRAM jobmanager services, in the following order: contacts with job manager types other than fork and poe; contacts with job manager type poe; and contacts with job manager type fork. A contact string is on the form: "host:port/servicename:certificate subject".

Command	Description
`globus-gass-cache`	Lists/modifies the contents of a GASS cache Allows a user to manipulate the contents of a local or remote GASS cache. Each entry in a GASS cache consists of a URL, local filename, list of tags and a reference count for each tag. The most common tags are "null" (no tag specified) and JobID URLs (when the program is using the GASS cache when started through GRAM; all JobID tags are automatically deleted for a particular job when the GRAM job manager completes). When the last tag for a URL is removed, the local file is removed from the cache.
`globus-gass-cache-destroy`	Will destroy (clean without coherence check) the Globus cache on all the machines listed in ~/.globus/my-contacts, or in <file> if -f option used, or only on the machine specified with the -t option.
`globus-list-my-contacts`	Creates a list of machine (gatekeeper contacts) on which the user has "globus access".
`globus-netstat`	Hides the implementation-specifics of netstat and reformats the output to be consistent across architectures, producing a subset of Unix System V netstat output.
`globus-sh-exec`	Sources the globus-sh-tools file, then executes a user script. Allows the user to run a script adhering to the Bourne shell format on a remote machine without having to worry about correct paths to various Unix commands. To facilitate this, globus-sh-exec provides to the user script a set of GLOBUS_SH_PROG variables, pointing to the location of program PROG. About 125 commands are currently defined in this way. In addition the to the GLOBUS_SH_* variables, the variable bindir is defined, pointing to the Globus tools bin directory. The variables sbindir, libexecdir and a couple more are defined in the same fashion. A rudimentary PATH variable set to cover most system commands is also defined before the script is run. globus-sh-exec also recognizes a GASS URL as script argument, and handles it correctly. Additional optional arguments are passed on to the user provided script.
`globus-version`	Shows version number. Returns version information of currently installed Toolkit, such as patch level, release date, etc. If no arguments are given, the assumed option is "-string".

Command	Description
`globus-deploy-path`	globus-deploy-path prints the full path to the local deploy directory, which contains the machine-specific setups etc. The location of the deploy tree is determined as follows, in the following order: the path given by env.var. GLOBUS_DEPLOY_PATH; the path given to globus-local-deploy (only applies to deployed copies of globus-deploy-path); and the deploy path given by the GlobusService object in the MDS. If the local deploy directory cannot be located, "<not found>" is returned.
`globus-development-path`	Prints the full path to the "development" directory (include files and libraries) that corresponds best to the flavor indicated by the command-line options (e.g., pthreads, debug, and 64-bit) The development subtree in the Globus install directory contains several different "flavors", mostly affecting the communications library Nexus. (Nexus has the ability to use underlying vendor-provided high-performance communications libraries.) The program auto-senses the architecture on which it is running. If the desired flavor has not been built when installing Globus, "<not found>" is returned. If no arguments are given, "-debug -standard -nothreads -32 -64" is assumed.
`globus-install-path`	Prints the full path to the Globus install tree. The location of the install tree is as follows, in the following order: the path given by env.var. GLOBUS_INSTALL_PATH and the path that was given as "prefix" when installing Globus. If an installed tree cannot be located, "<not found>" is returned.
`globus-personal-gatekeeper`	Options: -help, -usage (displays usage), -version (displays version), -ebug (displays extra output), -start [-jmtype <type>] [-port <port>] (starts a new gatekeeper, mapping default service to a jobmanager. By default, the jobmanager is configured with jmtype=fork. The option -port can be used to restrict the gatekeeper to use a particular port. The default is to let the system choose a port. -list (scans for active personal gatekeepers. If found, an authentication test is made to determine if the gatekeeper is still functioning. If the authentictation test succeeds, the contact to that gatekeeper is printed out.) -directory <contact> (eturns the temporary directory used by the personal gatekeeper associated with <contact>. The directory contains grid-mapfile, log file, etc.; -kill <contact> (Finds and kills the personal gatekeeper associated with <contact>); -killall (finds all personal gatekeepers running on the local host and kills them.)
`globus-rcp`	Remote copies using GASS and Globus submission. Many options. Allows the user to copy files to and from remote locations. It attempts to as much as possible mimic the functionality of cp and rcp.

Command	Description
`globus-tools-path`	Prints the full path to the tools directory in the Globus install tree, tailored for the current architecture. The program auto-senses the architecture on which it is running. If the tools directory cannot be located, an empty string is returned. The default search for a services directory can be overridden by setting the environment variable GLOBUS_TOOLS_PATH.
`globus-services-path`	Prints the full path to the services directory in the Globus install tree, tailored for the current architecture. The program auto-senses the architecture on which it is running. If the services directory cannot be located, an empty string is returned. The default search for a services directory can be overridden by setting the environment variable GLOBUS_SERVICES_PATH.
`globus-tilde-expand`	Expands the leading tilde sign (~) (and the specified user name if provided) to the full path of the user's home directory (or of the specified user), the same way the C shell proceeds. This command is intended to be used in command interpreters that do not perform such expansions, such as /bin/sh. The expanded string is returned without any carriage return termination character, to allow easy concatenation in a shell script.
`globus-url-copy`	Remote file copy using URL syntax Copies a file specified by sourceURL to a location specified by destURL, using the GASS transfer API. All protocols supported by GASS (local file, http, https, ...) are supported. Piping to/from stdin/stdout (setting source/dest argument = '-') is supported.

Appendix B Options to globus-install

There are close to 300 options to the `globus-install` script. They are presented in this appendix in tabular form grouped by functionality. Except for options to `configure`, which are preceded by double hyphens, all options to `globus-install` are preceded by a single hyphen.

Configuration Options

-cache-file=FILE	Cache test results in FILE
-help	Print this message
-no-create	do not create output files
-quiet, -silent	do not print ‘checking...’ messages
-version	Print the version of autoconf that created configure

Directory and File Name Options

-prefix=PREFIX	install architecture-independent files in PREFIX [/usr/local]
-exec-prefix=EPREFIX	install architecture-dependent files in EPREFIX [same as prefix]
-bindir=DIR	user executables in DIR [EPREFIX/bin]
-sbindir=DIR	system admin executables in DIR [EPREFIX/sbin]
-libexecdir=DIR	program executables in DIR [EPREFIX/libexec]
-datadir=DIR	read-only architecture-independent data in DIR [PREFIX/share]
-sysconfdir=DIR	read-only single-machine data in DIR [PREFIX/etc]
-sharedstatedir=DIR	modifiable architecture-independent data in DIR [PREFIX/com]

-localstatedir=DIR	modifiable single-machine data in DIR [PREFIX/var]
-libdir=DIR	object code libraries in DIR [EPREFIX/lib]
-includedir=DIR	C header files in DIR [PREFIX/include]
-oldincludedir=DIR	C header files for non-gcc in DIR [/usr/include]
-infodir=DIR	info documentation in DIR [PREFIX/info]
-mandir=DIR	man documentation in DIR [PREFIX/man]
-srcdir=DIR	find the sources in DIR [configure dir or ..]
-program-prefix=PREFIX	prepend PREFIX to installed program names
-program-suffix=SUFFIX	append SUFFIX to installed program names
-program-transform-name=PROGRAM	run sed PROGRAM on installed program names

Host Type Options

-build=BUILD	Configure for building on BUILD [BUILD=HOST]
-host=HOST	Configure for HOST [guessed]
-target=TARGET	Configure for TARGET [TARGET=HOST]

Feature and Package Options

-disable-FEATURE	do not include FEATURE (same as --enable-FEATURE=no)
-enable-FEATURE[=ARG]	Include FEATURE [ARG=yes]
-with-PACKAGE[=ARG]	Use PACKAGE [ARG=yes]
-without-PACKAGE	do not use PACKAGE (same as --with-PACKAGE=no)
-x-includes=DIR	X include files are in DIR
-x-libraries=DIR	Xlibrary files are in DIR

-enable and -with Options

-with-gssapi	select cleartext or ssleay [ssleay]
-enable-alpha	include Alpha components
-enable-beta	include Beta components
-enable-globusrun	configure globusrun and dependents
-enable-globus-common	configure globus_common
-enable-dc	configure DC (dataconversion)
-enable-utp	configure UTP (timers)
-enable-gssapi	configure GSSAPI
-enable-gaa	configure GAA
-enable-globus-io	configure globus_io
-enable-netlogger	configure netlogger
-enable-nexus	configure Nexus
-enable-rsl	configure RSL
-enable-gram	configure GRAM
-enable-gass	configure GASS (remote file access)
-enable-duct	configure DUCT
-enable-duroc	configure DUROC
-enable-hbm	configure HeartbeatMonitor
-enable-mds	configure MDS
-enable-ldap	configure Umich LDAP
-enable-startup	configure RC startup scripts
-with-tmpdir=DIR	temporary files in DIR [PREFIX/tmp]
-with-local-tmpdir=LDIR	local temporary files in LDIR [/tmp]

-with-secure-tmpdir=SECDIR	local temporary files in SECDIR [LDIR]
-with-globus-prefix=DIR	top directory of Globus installation
-with-globus-services-prefix=DIR	directory for architecture-specific Globus services
-with-globus-tools-prefix=DIR	directory for architecture-specific Globus tools
-with-deploy-prefix=DIR	root uid files in DIR [EPREFIX]
-with-globus-uid=	user id of Globus system tasks
-with-globus-gid=	group id of Globus system tasks
-enable-new-mpich-makefile	compatiblity with post-MPICH 1.1.2
-with-domain-name=DOMAIN	only used if the FQN is unknown
-with-ssl-path=DIR	use SSLeay files in DIR
-with-gsi-base-dn=value	DN of the local site
-with-ldap-path=DIR	use LDAP files in DIR
-with-ldap-host=value	Name of the local LDAP host
-with-ldap-port=value	Port number for the local LDAP host
-with-ldap-base-dn=value	DN of the local site
-enable-ldap-debug	enable ldap debugging output
-enable-uofm	enable U Mich Extensions LDAP
-enable-userinterface	enable user interface from LDAP library
-enable-cldap	enable connectionless LDAP
-enable-ldap-referrals	enable referrals in LDAP
-enable-cache	enable local caching in LDAP library
-with-ssl-path=DIR	use SSLeay files in DIR
-with-gsi-base-dn=value	DN of the local site
-with-threads	build target with threads

-with-thread-library=PATH	path to thread library files
-with-thread-includes=PATH	path to thread include files
-enable-debug	compile in debugging features
-enable-64bit	build 64-bit objects (SGI Irix6.x and HP HPUX11.x only)
-with-mpl	include the IBM SP MPL protocols
-with-mpi	include the MPI protocols
-with-inx	include the Paragon INX protocols
-enable-sharedlib	enable building of shared libraries
--with-TARGET-CC=value	set TARGETMT and TARGETST CC
--with-TARGET-CPP=value	set TARGETMT and TARGETST CPP
--with-TARGET-CFLAGS=value	set TARGETMT and TARGETST CFLAGS
--with-TARGET-CXX=value	set TARGETMT and TARGETST CXX
--with-TARGET-CXXCPP=value	set TARGETMT and TARGETST CXXCPP
--with-TARGET-CXXFLAGS=value	set TARGETMT and TARGETST CXXFLAGS
--with-TARGET-LD=value	set TARGETMT and TARGETST LD
--with-TARGET-LDFLAGS=value	set TARGETMT and TARGETST LDFLAGS
--with-TARGET-LIBS=value	set TARGETMT and TARGETST LIBS
--with-TARGET-AR=value	set TARGETMT and TARGETST AR
--with-TARGET-ARFLAGS=value	set TARGETMT and TARGETST ARFLAGS
--with-TARGET-RANLIB=value	set TARGETMT and TARGETST RANLIB
--with-TARGET-CXX_WORKS=value	set TARGETMT and TARGETST CXX_WORKS
--with-TARGET-CROSS=value	set TARGETMT and TARGETST CROSS
--with-TARGET-F77=value	set TARGETMT and TARGETST F77
--with-TARGET-F77FLAGS=value	set TARGETMT and TARGETST F77FLAGS

--with-TARGET-F90=value	set TARGETMT and TARGETST F90
--with-TARGET-F90FLAGS=value	set TARGETMT and TARGETST F90FLAGS
--with-TARGETST-CC=value	set TARGETST CC
--with-TARGETST-CPP=value	set TARGETST CPP
--with-TARGETST-CFLAGS=value	set TARGETST CFLAGS
--with-TARGETST-CXX=value	set TARGETST CXX
--with-TARGETST-CXXCPP=value	set TARGETST CXXCPP
--with-TARGETST-CXXFLAGS=value	set TARGETST CXXFLAGS
--with-TARGETST-LD=value	set TARGETST LD
--with-TARGETST-LDFLAGS=value	set TARGETST LDFLAGS
--with-TARGETST-LIBS=value	set TARGETST LIBS
--with-TARGETST-AR=value	set TARGETST AR
--with-TARGETST-ARFLAGS=value	setTARGETST ARFLAGS
--with-TARGETST-RANLIB=value	set TARGETST RANLIB
--with-TARGETST-CXX_WORKS=value	set TARGETST CXX_WORKS
--with-TARGETST-CROSS=value	set TARGETST CROSS
--with-TARGETST-F77=value	set TARGETST F77
--with-TARGETST-F77FLAGS=value	set TARGETST F77FLAGS
--with-TARGETST-F90=value	set TARGETST F90
--with-TARGETST-F90FLAGS=value	set TARGETST F90FLAGS
--with-TARGETMT-CC=value	set TARGETMT CC
--with-TARGETMT-CPP=value	set TARGETMT CPP
--with-TARGETMT-CFLAGS=value	set TARGETMT CFLAGS
--with-TARGETMT-CXX=value	set TARGETMT CXX

--with-TARGETMT-CXXCPP=value	set TARGETMT CXXCPP
--with-TARGETMT-CXXFLAGS=value	set TARGETMT CXXFLAGS
--with-TARGETMT-LD=value	set TARGETMT LD
--with-TARGETMT-LDFLAGS=value	set TARGETMT LDFLAGS
--with-TARGETMT-LIBS=value	set TARGETMT LIBS
--with-TARGETMT-AR=value	set TARGETMT AR
--with-TARGETMT-ARFLAGS=value	set TARGETMT ARFLAGS
--with-TARGETMT-RANLIB=value	set TARGETMT RANLIB
--with-TARGETMT-CXX_WORKS=value	set TARGETMT CXX_WORKS
--with-TARGETMT-CROSS=value	set TARGETMT CROSS
--with-TARGETMT-F77=value	set TARGETMT F77
--with-TARGETMT-F77FLAGS=value	set TARGETMT F77FLAGS
--with-TARGETMT-F90=value	setTARGETMTF90
--with-TARGETMT-F90FLAGS=value	set TARGETMT F90FLAGS
--with-SERVICE-CC=value	set SERVICE CC
--with-SERVICE-CPP=value	set SERVICE CPP
--with-SERVICE-CFLAGS=value	set SERVICE CFLAGS
--with-SERVICE-CXX=value	set SERVICE CXX
--with-SERVICE-CXXCPP=value	set SERVICE CXXCPP
--with-SERVICE-CXXFLAGS=value	set SERVICE CXXFLAGS
--with-SERVICE-LD=value	set SERVICE LD
--with-SERVICE-LDFLAGS=value	set SERVICE LDFLAGS
--with-SERVICE-LIBS=value	set SERVICE LIBS
--with-SERVICE-AR=value	set SERVICE AR

--with-SERVICE-ARFLAGS=value	set SERVICE ARFLAGS
--with-SERVICE-RANLIB=value	set SERVICE RANLIB
--with-SERVICE-CXX_WORKS=value	set SERVICE CXX_WORKS
--with-SERVICE-CROSS=value	set SERVICE CROSS
--with-SERVICE-F77=value	set SERVICE F77
--with-SERVICE-F77FLAGS=value	set SERVICE F77FLAGS
--with-SERVICE-F90=value	set SERVICE F90
--with-SERVICE-F90FLAGS=value	set SERVICE F90 FLAGS
--with-HOST-CC=value	set HOST CC
--with-HOST-CPP=value	set HOST CPP
--with-HOST-CFLAGS=value	set HOST CFLAGS
--with-HOST-CXX=value	set HOST CXX
--with-HOST-CXXCPP=value	set HOST CXXCPP
--with-HOST-CXXFLAGS=value	set HOST CXXFLAGS
--with-HOST-LD=value	set HOST LD
--with-HOST-LDFLAGS=value	set HOST LDFLAGS
--with-HOST-LIBS=value	set HOST LIBS
--with-HOST-AR=value	set HOST AR
--with-HOST-ARFLAGS=value	set HOST ARFLAGS
--with-HOST-RANLIB=value	set HOST RANLIB
--with-HOST-CXX_WORKS=value	set HOST CXX_WORKS
--with-HOST-CROSS=value	set HOST CROSS
--with-HOST-F77=value	set HOST F77
--with-HOST-F77FLAGS=value	set HOST F77FLAGS

--with-HOST-F90=value	set HOSTF90
--with-HOST-F90FLAGS=value	set HOST F90FLAGS
-disable-macros	use function calls instead of macros
-enable-critical-path-timer	time the critical path
-enable-poll-range-checking	check poll ranges
-enable-resource	nexus resource abstractions
-enable-qos	quality of service overseer
-with-tcp	include the TCP/IP protocol
-with-udp	include the UDP/IP protocol
-with-totem	include the Totem protocol
-with-gram-job-manager	GRAM job manager: condor, easymcs, lsf, loadleveler, poe, orfork[fork]
-with-gram-gatekeeper	Install the gram gatekeeper [yes]
-with-gram-port	port number of the gatekeeper [2119]
-disable-gram-test	don't configure GRAM test
-enable-duroc-debug	enable DUROC debug messages
-disable-duroc-warnings	disable DUROC warning messages

HBM Data Collector Options

-with-hbm-dc-host=value	name or IP number of the host with the default
-with-hbm-dc-cmdport=value	command (and status request) port number for default HBM Data Collector [8337]
-with-hbm-dc-hbport=value	heartbeat port number for default HBM Data Collector [12003]
-with-hbm-dc-eval-interval=value	interval (secs) at which HBM Data Collector evaluates clients and writes checkpoints to stable storage[10 seconds]

-with-hbm-lm-ps-executable=value	location of ps for HBM Local Monitor to use for client evaluation [/bin/ps]
-with-hbm-lm-hb-interval=value	interval (secs) in which the HBM generates heartbeats [5 seconds]
-with-mds-root-dn=value	Root DN of the MDS server
-with-mds-host=value	Name of the local MDS host
-with-mds-port=value	Port number for the local MDS host
-with-mds-base-dn=value	DN of the local site
-with-mds-admin-dn=value	DN of the local site's administrator
-with-ldap-path=DIR	use LDAP files in DIR
-with-ldap-host=value	Name of the local LDAP host
-with-ldap-port=value	Port number for the local LDAP host
-with-ldap-base-dn=value	DN of the local site
-enable-ldap-debug	enable ldap debugging output
-enable-uofm	enable U of Michigan Extensions LDAP
-enable-userinterface	enable user interface from LDAP library
-enable-cldap	enable connectionless LDAP
-enable-ldap-referrals	enable referrals in LDAP
-enable-cache	enable local caching in LDAP library
-disable-gass-file-api	don't configure the gass file api library
-disable-gass-cache	don't configure the gass cache library
-disable-gass-client	don't configure the gass client library
-disable-gass-server	don't configure the gass server library
-disable-gass-server-ez	don't build the gass server ez library
-enable-gass-tests	configure the gass tests library

-disable-tools-aix-dynaload	don't configure AIX dynamic loader tools
-disable-tools-aix-shared-lib	don't configure AIX shared lib tools
-enable-tools-gass-cache	configure GASS cache tools
-enable-tools-globusrun	configure globusrun tool
-with-thread-library=PATH	path to thread library files
-with-thread-includes=PATH	path to thread include files
-with-threads	build target with threads
-with-thread-library=PATH	path to thread library files
-with-thread-includes=PATH	path to thread include files
-enable-debug	compile in debugging features
-enable-64bit	build 64-bit objects (SGI Irix 6.x and HP HPUX 11.x only)
-with-mpl	include the IBM SP MPL protocols
-with-mpi	include the MPI protocols
-with-inx	include the Paragon INX protocols
-enable-sharedlib	enable building of shared libraries
--with-TARGET-CC=value	set TARGETMT and TARGETST CC
--with-TARGET-CPP=value	set TARGETMT and TARGETST CPP
--with-TARGET-CFLAGS=value	set TARGETMT and TARGETST CFLAGS
--with-TARGET-CXX=value	set TARGETMT and TARGETST CXX
--with-TARGET-CXXCPP=value	set TARGETMT and TARGETST CXXCPP
--with-TARGET-CXXFLAGS=value	set TARGETMT and TARGETST CXXFLAGS
--with-TARGET-LD=value	set TARGETMT and TARGETST LD
--with-TARGET-LDFLAGS=value	set TARGETMT and TARGETST LDFLAGS
--with-TARGET-LIBS=value	set TARGETMT and TARGETST LIBS

--with-TARGET-AR=value	set TARGETMT and TARGETST AR
--with-TARGET-ARFLAGS=value	set TARGETMT and TARGETST ARFLAGS
--with-TARGET-RANLIB=value	set TARGETMT and TARGETST RANLIB
--with-TARGET-CXX_WORKS=value	set TARGETMT and TARGETST CXX_WORKS
--with-TARGET-CROSS=value	set TARGETMT and TARGETST CROSS
--with-TARGET-F77=value	set TARGETMT and TARGETST F77
--with-TARGET-F77FLAGS=value	set TARGETMT and TARGETST F77FLAGS
--with-TARGET-F90=value	set TARGETMT and TARGETST F90
--with-TARGET-F90FLAGS=value	set TARGETMT and TARGETST F90FLAGS
--with-TARGETST-CC=value	set TARGETST CC
--with-TARGETST-CPP=value	set TARGETST CPP
--with-TARGETST-CFLAGS=value	set TARGETST CFLAGS
--with-TARGETST-CXX=value	set TARGETST CXX
--with-TARGETST-CXXCPP=value	set TARGETST CXXCPP
--with-TARGETST-CXXFLAGS=value	set TARGETST CXXFLAGS
--with-TARGETST-LD=value	set TARGETST LD
--with-TARGETST-LDFLAGS=value	set TARGETST LDFLAGS
--with-TARGETST-LIBS=value	set TARGETST LIBS
--with-TARGETST-AR=value	set TARGETST AR
--with-TARGETST-ARFLAGS=value	set TARGETST ARFLAGS
--with-TARGETST-RANLIB=value	set TARGETST RANLIB
--with-TARGETST-CXX_WORKS=value	set TARGETST CXX_WORKS
--with-TARGETST-CROSS=value	set TARGETST CROSS
--with-TARGETST-F77=value	set TARGETST F77

--with-TARGETST-F77FLAGS=value	set TARGETST F77FLAGS
--with-TARGETST-F90=value	set TARGETST F90
--with-TARGETST-F90FLAGS=value	set TARGETST F90FLAGS
--with-TARGETMT-CC=value	set TARGETMT CC
--with-TARGETMT-CPP=value	set TARGETMT CPP
--with-TARGETMT-CFLAGS=value	set TARGETMT CFLAGS
--with-TARGETMT-CXX=value	set TARGETMT CXX
--with-TARGETMT-CXXCPP=value	set TARGETMT CXXCPP
--with-TARGETMT-CXXFLAGS=value	set TARGETMT CXXFLAGS
--with-TARGETMT-LD=value	set TARGETMT LD
--with-TARGETMT-LDFLAGS=value	set TARGETMT LDFLAGS
--with-TARGETMT-LIBS=value	set TARGETMT LIBS
--with-TARGETMT-AR=value	set TARGETMT AR
--with-TARGETMT-ARFLAGS=value	set TARGETMT ARFLAGS
--with-TARGETMT-RANLIB=value	set TARGETMT RANLIB
--with-TARGETMT-CXX_WORKS=value	set TARGETMT CXX_WORKS
--with-TARGETMT-CROSS=value	set TARGETMT CROSS
--with-TARGETMT-F77=value	set TARGETMT F77
--with-TARGETMT-F77FLAGS=value	set TARGETMT F77FLAGS
--with-TARGETMT-F90=value	set TARGETMT F90
--with-TARGETMT-F90FLAGS=value	set TARGETMT F90FLAGS

Appendix C Index

NOTE: Individual commands from the text are indexed. Individual commands, which are included in Appendix A, are not listed individually in this index. Appendix A commands are referenced by functionality (e.g., security commands or information services commands).