Globus Quick Start Guide

Globus Software Version 1.1.3 and 1.1.4   June 2001
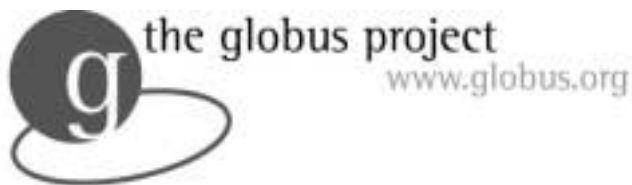
the globus project
www.globus.org

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

© 1999, 2000, 2001 by the University of Chicago and the University of Southern California

All rights reserved except that you are permitted to make copies of this documentation as long as the copyright and other notices written in the documentation are preserved.

All brand and product names are trademarks or registered trademarks of their respective holders.

The University of Chicago and the University of Southern California give no warranty, express or implied, for the software or documentation provided, including, without limitation, warranty of merchantability and warranty of fitness for a particular purpose.

This document was produced as a community effort by the Information Power Grid (IPG) group at NASA Ames Research Center, Code IN; the San Diego Supercomputer Center; Argonne National Laboratory; the National Computational Science Alliance; and the University of Southern California's Information Sciences Institute.

Please report errors in this document to documentation@globus.org

This guide is available in HTML at http://www.ipg.nasa.gov/ *-- User Support -- Documentation*

Contents

Summary of New Features in Globus 1.1.3

Release 1.1.3 of the Globus Toolkit provides a more scalable and flexible Grid Information Service (GIS), also known as MDS, and streamlined interfaces for the Globus Resource Allocation and Management (GRAM) tools. Additionally, release 1.1.3 includes a new version of the globus_io library with significantly improved performance. The following differences affect the user interface:

- `grid-cert-request`: Several organizations now have their own Certificate Authority (CA). Ask your local Globus administrator for information on obtaining a Certificate. See page 15.

- Resource contact strings for "fork" services are no longer routinely suffixed with `/jobmanager-fork`. If you do not specify a job manager in your `globusrun`, `globus-job-run`, or `globus-job-submit` command, your job will be submitted to the default job manager on the remote system, which is often fork. If you want to specify a non-default job manager, do so by appending `/jobmanager` and the name of the job scheduler, such as `/jobmanager-lsf` or `/jobmanager-pbs`, to the hostname. See page 28

- GIS/MDS: The Globus group no longer runs a centralized MDS server. Each site optionally will have its own organization server, which is used for any `grid-info` commands. See page 34.

Known Deficiencies in 1.1.3

When using the Globus Toolkit on IRIX systems, the `sproc_mpi` and `pthreads_mpi` libraries are not built properly. We do not recommend using these libraries on IRIX with Globus 1.1.3. This problem is corrected in 1.1.4.

Summary of New Features in Globus 1.1.4

Release 1.1.4 of the Globus Toolkit provides support for MPICH-G2, a new grid-enabled MPI implementation.

The MPICH-G2 release requires minor optimizations and bug fixes to the Globus data conversion library, globus_io library, and GRAM components. These changes are all backward-compatible with previous versions of the Globus Toolkit. There are no changes to Globus protocols or APIs.

If you do not intend to use MPICH-G2, there is no reason to install release 1.1.4 of the Toolkit. If you are installing the Globus Toolkit for the first time and you don't expect to use MPICH-G2, we recommend that you install release 1.1.3. Release 1.1.3 remains a fully-supported version of the Globus Toolkit.

Coming soon: Globus 2.0 is in alpha testing now.

Chapter 1   Extremely Quick Start

Here are the briefest instructions for getting started with Globus Version 1.1.3 or 1.1.4.

1. Get accounts on Globus-enabled machines. See page 13 for contact information.

2. Login to a Globus-enabled machine.

3. Set up your environment and path, if necessary.

4. Ask your local Globus administrator for information on obtaining a Certificate. See page 13 for contact information.

5. Enter `grid-proxy-init`; enter passphrase when asked.

6. Enter `globusrun -a -r` **<hostname>** to test that Globus is running on that host.

7. Enter `globus-job-run` *<hostname>* `/bin/echo "Hello World."`

8. Use *<globus command>* `-help` for information on Globus commands.

Chapter 2   Introduction

Grids are super Internets for high-performance computing: worldwide collections of high-end resources—such as supercomputers, storage, advanced instruments, and immersive environments. These resources and their users are often separated by great distances and connected by high-speed networks.

The Globus development team has created a set of underlying Grid services and a software toolkit for using the geographically distributed resources on Grids.

This chapter contains background information only, no instructions. This manual assumes you know Unix. The "%" sign represents the Unix prompt in examples.

What are Grids?

Grids bring together geographically and organizationally dispersed computational resources, such as CPUs, storage systems, communication systems, real-time data sources and instruments, and human collaborators. If you are developing an application that requires geographically distributed high-end resources, you will want to know about Grids. Grids are new—many of the enabling technologies have not yet been invented. If you are reading this in the early 2000s, you are one of the Grid pioneers.

Current testbed Grids will eventually become "the Grid." In much the same way that the nationwide electric power grid connects sources of electricity, the information Grid will connect sources and users of high-performance computing cycles, allowing these cycles to be generated in one place and used in another. Remote collaborators will work together on large-scale projects. Scientists will be able to access extremely data-intensive instruments from across the country, in real time.

The goal of the Grid community is to provide dependable, consistent, pervasive access to high-end resources.

What is Globus?

The Globus software toolkit facilitates the creation of usable Grids, enabling high-speed coupling of people, computers, databases, and instruments. With Globus, you can run your gigabyte-per-time-step dataset job on two or more high-performance machines at the same time, even though the machines might be located far apart and owned by different organizations. Globus software helps scientists deal with very large datasets and complex remote collaborations.

Globus software is used for large distributed computational jobs, remote instrumentation, remote data transfer, and shared immersive spaces. For example,

- Globus used in live runs at the world's brightest x-ray source :
  http://www.mcs.anl.gov/~laszewsk/xray/cmt/gallery/

- Distributed supercomputing, smart instruments, teleimmersive applications:
  http://www.globus.org/overview/applications.html

Globus Features

Globus is designed to offer features such as uniform access to distributed resources with diverse scheduling mechanisms; information service for resource publication, discovery, and selection; API and command-line tools for remote file management, staging of executables and data; and enhanced performance through multiple communication protocols.

Testbeds

Grids are a community effort. University, commercial, and government computer science communities contribute to the development of Grids. Significant testbeds are up and running.

Grids use Globus services to provide uniform access to a large collection of resources for a particular Grid community. There are many Grids in progress. Work is ongoing to enable interoperation of these various testbeds; however, most likely you will be using one particular testbed.

The first testbed was GUSTO. The major current testbeds are the National Technology Grid and the Information Power Grid. See Grids and Local Accounts, page 13, for contact information.

The Integrated Grid Architecture
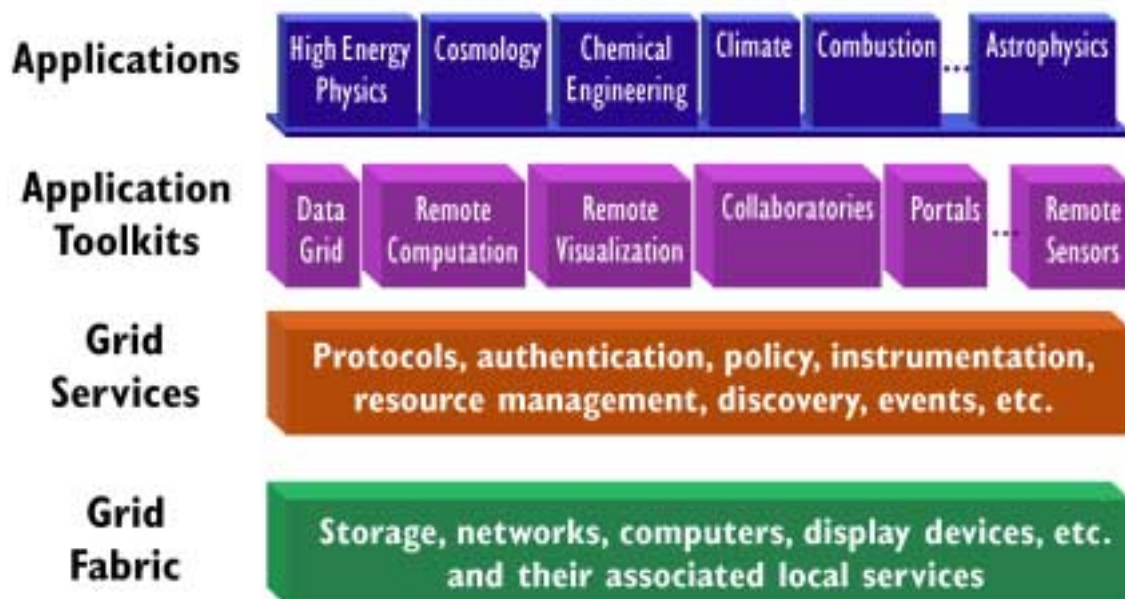
Grid Fabric: Layer One

The *fabric* of the Grid comprises the underlying systems, computers, operating systems, networks, storage systems, and routers—the building blocks.

Grid Services: Layer Two

Grid services *integrate* the components of the Grid fabric. Examples of the services that are provided by Globus:

GRAM

The Globus Resource Allocation Manager, GRAM, is a basic library service that provides capabilities to do remote-submission job start up. GRAM unites Grid machines, providing a common user interface so that you can submit a job to multiple machines on the Grid fabric. GRAM is a general, ubiquitous service, with specific application toolkit commands built on top of it.

GIS

The Grid Information Service, GIS, also known as the Metacomputing Directory Service, MDS, provides information service. (The terminology is still being discussed.) You query GIS (MDS) to discover the properties of the machines, computers and networks that you want to use: how many processors are available at this moment? What bandwidth is provided? Is the storage on tape or disk? Is the visualization device an immersive desk or CAVE? Using an LDAP (Lightweight Directory Access Protocol) server, GIS provides middleware information in a common interface to put a unifying picture on top of disparate equipment.

GSI

The Grid Security Infrastructure, GSI, is a library for providing generic security services for applications that will be run on the Grid. Application programmers use the `gss-api` library for adding authentication to a program. GSI provides programs, such as `grid-proxy-init`, to facilitate login to a variety of sites, while each site has its own flavor of security measures. That is, on the fabric layer, the various machines you want to use might be governed by disparate security policies; GSI provides a means of simplifying multiple remote logins. The information in this guide is based on a PKI security system; the Kerberos installation of Globus is not covered.

Application Toolkits: Layer Three

Application toolkits use Grid Services to provide higher-level capabilities, often targeted to specific classes of application.

For example, the Globus development team has created a set of Grid service tools and a toolkit of programs for running remotely distributed jobs. These include remote job submission commands (`globusrun`, `globus-job-submit`, `globus-job-run`), built on top of the GRAM service, and MPICH-G2, a Grid-enabled implementation of the Message Passing Interface (MPI).

A number of groups are also developing a range of other toolkits such as support for distributed management of large datasets, collaborative visualization, and online instrumentation. These are not discussed in this document.

Specific Applications: Layer Four

A rich variety of applications have been developed that build on services provided by the three layers just described. For example, OVERFLOW is a thin-layer Navier-Stokes flow solver for structured grids that has been modified to operate on multiple supercomputers via the use of MPICH-G2.

Other application areas being targeted to the Grid include high-energy physics, cosmology, chemical engineering, climate, combustion, and astrobiology.

Purpose of This Document

This *Globus Quick Start Guide* tells you how to get started using Globus Grid services and the tools built on top of those services to build applications to exploit geographically distributed resources. The instructions herein are brief and do not take you much past "Hello World," but do provide the essentials required to get started.

Support and Usage Help

Address questions to your local system administrator. Also refer to the Bibliography and Reference on page 41. For usage information about any Globus command, type the command with the -help or -usage option:

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

All programs have a man page with more elaborate explanations than provided by -help or -usage. http://www.globus.org/v1.1/programs/ for information.

About this Guide

Audience

This guide is intended for novice Globus users.

Assumptions

Before you start using Globus, you will need to fulfill these basic requirements:

Know Basic Unix

This manual assumes that you know at least the fundamentals of Unix. You can find good Unix information at http://www.ugu.com/sui/ugu/show?help.beginners. Like Unix, Globus *is* case sensitive, but Globus Resource Specification Language (RSL) attribute names are not.

Globus Is Installed

You probably will find Globus already installed on your Grid resources.

Globus provides distinct server and client software. Server software is intended for machines with remote access by a variety of users; server software is usually installed only on high-end

resources. Client software is for personal computers. At some institutions, Globus is installed on the fileservers for personal workstations, at other institutions, users ssh into machines running Globus. On the major Grids, you do not have to download any software to run Globus.

The information in this guide is based on a PKI security system; the Kerberos installation of Globus is not covered.

Organization and Conventions

The chapters are presented in step-by-step sequence. Only the bare essentials are included. A glossary defines terms. Globus commands start with `globus` or `grid`. "Resources" refers to computers, instruments, or other machines (as opposed to data).

Conventions used in this guide:

`%`	Unix prompt; do not enter
<italics>	information to be substituted. For example, *<src-dir>* means enter in your source directory pathname. Also used for emphasis.
`[  ]`	parameters that sometimes may be omitted
`\`	indicates that a command line continues on the next line

How to Use This Manual Online

If you are reading the PDF version online, use the button marked with a T or ABC to make text selectable; then you can copy the examples and paste them into your Unix command line. Be careful not to copy the backslashes (\) into the middle of your command lines. Also watch for disappearing hyphens (-).

Chapter 3   Grid Certificate

In this chapter, you obtain a Grid Certificate to have single sign-on access to all of the Grid machines on which you have accounts. Your Grid Certificate is like a passport; it establishes your identity. This is what you do:

- Obtain accounts on geographically distributed machines.

- Check, and if necessary, set your GLOBUS_INSTALL_PATH environment variable and your path to Globus tools.

- On the website for your Grid, follow instructions for obtaining a certificate.

Why You Want a Certificate

With a Grid certificate, you only have to enter your password once per Globus session. The Grid certificate is like a passport. At first, you take the time to establish your identity to the proper authorities, and then you are issued a certificate and private key. The certificate and key give you geographically distributed access. You can use remote Grid machines without reentering your password. You do not have to reestablish your identity at each "border."

Grids and Local Accounts

Before using Globus, you will need to get accounts on the Grid computers you plan to use. The two major testbeds for Globus are listed below; many other Grids are being formed. You can start your own Grid with the information in the Globus Toolkit 1.1.3 System Administration Guide available at http://www.globus.org/toolkit/documentation/

If You Already Have One or More Accounts at One of The Institutions Below

If you already have a regular account, you still need to apply for Grid accounts; these will give you access to additional machines. Go to the website (see below) and follow instructions.

If You Do Not Already Have a Regular Account

Getting a new account is non-trivial, and will take at least a few days. The more compute cycles you request, the more complicated the process.

Where to Apply

- **IPG**, the Information Power Grid—this is NASA's testbed for Grid research. See http://www.ipg.nasa.gov/ – Basic Usage – IPG Quick Start Guide. IPG includes NASA Langley Research Center, NASA Glenn Research Center, and NASA Ames Research Center/NAS.

- **National Technology Grid**—this is the testbed created by the National Computational Science Alliance (NCSA) and the National Partnership for Advanced Computational Infrastructure (NPACI).

  To apply for NCSA accounts, see http://access.ncsa.uiuc.edu/
  or send email to allocations@ncsa.uiuc.edu

  For NPACI, see http://www.npaci.edu/Globus/quickstart.html and
  http://www.npaci.edu/Globus/

Your Globus Home and Remote Machines

Pick a machine to be your Globus home machine.

Your *Globus* home is the machine where your `usercert.pem` and `userkey.pem` files will reside. When you apply for your Grid Certificate, do so from a machine that you have chosen to be your Globus home; your certificate and key (`.pem` files) will be installed there.

How Globus is deployed and installed varies from institution to institution. At SDSC and NASA, the Globus server is deployed on several high performance machines, and the Globus client is installed on several hundred workstations. Most users run Globus commands from their workstations, and their workstations are their home machines. The client machines submit jobs to be run on the servers where Globus is deployed.

Globus commands allow you to run executables on remote machines from your home machine. You only log in once, and you can run jobs on machines all over the country. Globus offers remote access without login. You only need to have your certificate and key (`.pem` files) on one machine, and that is your Globus home machine.

You can have more than one home. If for some reason you want to log onto a remote machine and run executables from there, you will need to copy your `.globus` directory to the remote machine, and reset the remote `.pem` file permissions, as described in Step 3, page 16. Permissions often change when you copy the files. But the point of Globus is that you can use remote machines without logging in.

You do not have to download any software in order to run Globus. If the Globus client is not installed on your workstation, simply login to a machine where Globus is installed.

Set Your Environment and Path

This is a critical step! Perhaps your kind and thoughtful system administrators have done it for you. To find out, on your Globus home machine (see above), type:

```
% which globus-job-submit
```

If you get a response that looks something like:

```
/software/globus-1.1.3/release/tools/mips-sgi-irix6.5/bin/globus-job-
submit
```

your path and environment variable have been set for you. If not, you will have to work with your system administrators to get it right. You need to set the GLOBUS_INSTALL_PATH environment variable, and add the Globus tools directory to your default search path. This path varies per system.

NOTE: Your previous Globus settings may create havoc. Get rid of them.

You can use the following commands within your Unix shell, or add them to your own dot files. Substitute *<path>* with the path to where Globus is installed on your system:

```
(csh, tcsh):
            setenv GLOBUS_INSTALL_PATH <path>
            source $GLOBUS_INSTALL_PATH/etc/globus-user-setup.csh
(bash, ksh):
            export GLOBUS_INSTALL_PATH=<path>
            . $GLOBUS_INSTALL_PATH/etc/globus-user-setup.sh
(sh)
            GLOBUS_INSTALL_PATH=<path>
            export GLOBUS_INSTALL_PATH
            . $GLOBUS_INSTALL_PATH/etc/globus-user-setup.sh
```

Look at your path by using the echo command. Check to see that there is no older Globus path preceding the new one; if there is an older path, remove it or move it to after the new one. Failure to fix this will give you older versions of Globus commands.

```
% echo $PATH
% echo $GLOBUS_INSTALL_PATH
```

Grid Certification

Once you have Grid accounts and have set your environment and path, you are ready to apply for a Grid Certificate.

Step 1—Apply and Retrieve

New in Version 1.1.3: Most of the Grid organizations that run Globus software now have their own Certificate Authority (CA). For information on how to obtain a certificate, see the **websites** listed in Grids and Local Accounts, page 13.

Whatever method you are using to get a Certificate, **apply from your Globus home** (see page 14). Follow all instructions presented by the certification program. When asked to enter a pass phrase, use at least 8 characters, including mixed-case alpha characters and at least one numeral or punctuation mark; you can enter a whole long phrase.

NOTE: You are the only person or thing that knows your pass phrase. Do not forget it! No one can retrieve it for you later. If you forget your pass phrase, start over (new certificate).

Globus CA --If for some reason you need to get a certificate from the Globus CA, as opposed to your own Grid's CA, login to your Globus home computer and execute the `grid-cert-request` command, then respond as directed.

Step 2—Save Your Certificate if Necessary

Your `usercert.pem` and `userkey.pem` belong in your `.globus` directory on your Globus home machine (see page 14); your application procedure includes instructions for this. **Most application programs do this for you**. The `.pem` files contain your signed public key and certificate, now saved where Globus knows to find them.

Step 3—Change Permissions

Change permissions on `.pem` files in your `.globus` directory. Enter `ls -l` to list your files. Use the `chmod` command to change permissions:

```
% chmod 400 userkey.pem
% chmod 444 usercert.pem
```

Checking Version

The commands in this book are based on Globus 1.1.x. To check your version, enter:

```
% globus-version
```

If you are not getting a valid version number such as 1.1.3, see Setting Environment and Path, page 14.

grid-mapfiles and Certificate Test

There is something called the grid-mapfile on every Globus resource. In order for you to use that resource, your subject name from your `certificate.pem` file must be added to the grid-mapfile. Some institutions do this automatically, some have a command for you to do it, some have an administrator who will do it eventually. To check if you are in the grid-mapfiles, from your home machine, enter:

```
 % globus-setup-test <hostname>
```

For example:

```
% globus-setup-test tfglobus.sdsc.edu
```

You will be asked to enter your PEM pass phrase; that is the pass phrase you gave when applying for your credential. If you get through
`Checking user proxy setup ..............done`,
your credentials are OK. Congratulations!

Otherwise, your subject name has not yet been added to the grid-mapfile. If you get
…`Failure` later in the response, there may be a system problem, or maybe `globus-setup-test` is not working (problems have been reported with `globus-setup-test` in some installations).

Ready to run

You are done. After you have successfully run `globus-setup-test`, you are ready to get a Grid proxy, which is easy and instant. See next page. Don't forget that pass phrase. As you run into bumps in the road, remember that you are a Grid pioneer. Do not expect all the roads to be paved. (Do not expect roads.) Grids do not yet run smoothly.

Chapter 4   Proxy Credential

In this chapter you obtain a Globus proxy by typing `grid-proxy-init`. The proxy gives you single sign-on capability for 12 hours (default). While your proxy is active, you can log into any Grid resource without reentering your pass phrase.

The command `grid-proxy-info -all` displays contents/status of your proxy credential.

Obtaining a Proxy Credential

When you have obtained accounts on some Grid testbed machines, and have a Grid Certificate properly saved (see previous chapter), you are ready to get a proxy. Basically, the proxy activates your Grid Certificate for a limited time. The Globus proxy is based on a public key security method that allows single sign-on anywhere a user has accounts on a Grid. One proxy is good for 12 hours (default). When you start another session on another day, you get a new proxy.

Using the grid-proxy-init Command

At your system prompt, enter `grid-proxy-init` and you are prompted for your Globus pass phrase.

```
% grid-proxy-init
```
prints:

```
Enter PEM pass phrase:
.  .........................+++++
.  .................+++++
%
```
You can set options with command line arguments to `grid-proxy-init`; type the command with the `-usage` option to see the possibilities. For example, you can set the time for the proxy to be in effect; use `grid-proxy-init -hours <nhours>`. While your proxy is in effect, you do not have to enter your pass phrase on any Grid machine. If your proxy expires, simply rerun `grid-proxy-init` to create a new proxy.

Destroying a Proxy

To get rid of your proxy when you are finished with it, enter `grid-proxy-destroy`. The proxy will self-destruct after 12 hours (or whatever duration you set), but it is a good idea to destroy it yourself when you are finished.

Determining Your Proxy Status

Use `grid-proxy-info -all` to display the contents or test the status of your proxy.

Chapter 5   Running a Job

In this chapter you will run a few simple Globus jobs, using `globus-job-run` and `globus-job-submit`. Then you will run `globusrun` with an RSL (Resource Specification Language) script. See page 30 for a whole chapter on RSL. The Globus Resource Allocation Manager (GRAM) authenticates and processes the requests for resources for remote application execution, and allocates the required resources.

Commands to Run a Job

globus-job-run

`globus-job-run` is the basic command for running Globus jobs. `globus-job-run` runs in the foreground and defaults to sending output to your terminal. In its basic form, it is roughly equivalent to `rsh`, but has considerably more functionality for running complex jobs on the Grid.

globus-job-submit

`globus-job-submit` is for submitting jobs to a remote batch job scheduler such as LSF or the Portable Batch System (PBS). With `globus-job-submit`, you can submit a job, log out, and log back in later to collect the output. That is, `globus-job-submit` runs in the background and defaults to sending output to the machine running the command. Retrieve output with `globus-job-get-output`, and then clean up with `globus-job-clean`.

globusrun

The `globusrun` command runs scripts written in the Globus Resource Specification Language (RSL). See the next chapter for details on RSL, page 30.

`globusrun` can run jobs either in the foreground or background, and can send output to your terminal or to the machine running the command. The trend in Globus software development is toward considering `globusrun` as middleware, which can be used by application specific shell scripts to manage job submission. In fact, `globus-job-run` and `globus-job-submit` are simply shell scripts that use `globusrun` for job submission, but present a simpler interface to users.

mpirun

MPICH-G2, an implementation of the Message Passing Interface (MPI) standard based on Globus, has an `mpirun` command that can be used to submit MPI parallel jobs to multiple computers in a Grid. See note on page vi, regarding IRIX problem in 1.1.3 (fixed in 1.1.4).

globus-sh-exec

The globus-sh-exec command allows the user to run a Bourne shell script on a remote machine without having to worry about correct paths to various UNIX commands. See page 33 and http://www.globus.org/details/programs/globus-sh-exec.html

Are You Ready to Run?

These examples assume you are logged into your Globus home. See page 14. If you have followed the directions in the previous chapters, you have accounts on Grid resources (computers or instruments), you have a Grid Certificate, you are in the resources' grid-mapfiles, and you have a current grid proxy.

The % in the examples indicates the Unix prompt (yours may vary); do not type it. The backslash (\) in long examples means that the following line should be a continuation of the current line; if you copy a line from this manual, be sure the backslash falls at the end of your command line, not in the middle.

In the previous chapter, you obtained a grid proxy. Check to see if your proxy is current.

```
% grid-proxy-info  -all
```
In the response from grid-proxy-info, your timeleft should have some time:
```
 timeleft : 11:42:20
```
If not, reenter grid-proxy-init.

To see that your credentials are in orders:

```
% globus-setup-test <host_name>
```
For example:

```
% globus-setup-test   evelyn.nas.nasa.gov
```
May print:

```
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.

Testing "jobmanager-pbs" service  on evelyn.nas.nasa.gov
Authentication test .....................Success!
Submission test .........................Success!

Testing "jobmanager-fork" service  on evelyn.nas.nasa.gov
Authentication test .....................Success!
Submission test .........................Success!

Testing Completed!
```
Notice that the response tells you which job managers are currently available. The authentication test is testing your credential; the submission test is testing the job manager. If

you get `Submission test .............Failure!`, or, at the bottom of the response: `No contact string found for gatekeeper <`*hostname*`>`, use some other resource. If you get:

` Authentication test ....................Failure!`

there may be a problem with `globus-setup-test`. Talk to the system administrator of the resource you are trying to use, or just proceed in spite of the failure notice.

To see if your resource is available:

` % globusrun -a -r <`*your_resource*`>`

For example:

` % globusrun -a -r evelyn.nas.nasa.gov`

If you get, "ERROR: resolving resource manager," the Grid Information Services (see page 34) may be malfunctioning, the machine may be down, or Globus may not be running.

Once you have run `grid-proxy-init` and `globusrun -a -r`, with successful results, you are good to go.

Host Information

See Default Job Managers on page 28 for how to find which job manager is the default. To find out what hosts are on a system or other information about hosts, use the Grid Information Service, page 34, or the website for your Grid.

globus-job-run: Hello World

The basic syntax of `globus-job-run` is the same as `rsh`.

Syntax:

` % globus-job-run  <hostname>  </path/executable>  <arguments>`

Example:

` % globus-job-run  evelyn.nas.nasa.gov  /bin/echo  Hello World`

prints:

` Hello World`

The Globus command (`globus-job-run`) comes first, then the hostname of the machine to run the command on (in this example, `evelyn.nas.nasa.gov`), then the program to run (the Unix `echo` program), and then the arguments (`Hello World`) to the program. The output of the program is `Hello World`. When the job is complete `globus-job-run` terminates.

Now try it on a geographically distant machine (on which you have an account). That is, from your Globus home, run the `echo` command on a remote machine. The `echo` executable already exists on the remote machine, as a standard Unix command. You should try all of the commands in this book on remote machines, to get a feel for moving around the Grid. First check to see if you are ready to run on the remote machine. Then run. For example:

` % globusrun -a -r  rogallo.larc.nasa.gov`

```
% globus-job-run  rogallo.larc.nasa.gov /bin/echo "Hello World"
```
See http://www.globus.org/v1.1/programs/globus-job-run.html for additional examples.

globus-job-run With Files

For these examples, you will write files to use with `globus-job-run`. These are not RSL files, and this is not `globusrun`; these are just executables that `globus-job-run` will execute on your specified host. We have attempted to write files that will work on any system, but if you have trouble with these files, just use your own executables. It is assumed you are logged on to your Globus home (page 14), that is, you are logged on to the machine that has your `.pem` certificate files.

Create a file named `hw`. (In `vi`, use `i` to start inserting text, `esc-colon-wq` to save and quit. Alternately, you can create the files with any editor. You can use `cat` to display your file.) The `globus-hostname` is a literal Globus command; do not substitute a host name. This file uses the standard Unix commands `echo` and `bc` (calculator). The first line of your file must start with `#` in the first position.

```
% vi hw
#!/bin/csh -f

set tools_bin = `$GLOBUS_INSTALL_PATH/bin/globus-tools-path -bindir`

echo ""
echo -n "Hello Globus World from " ; $tools_bin/globus-hostname
echo -n "sum is "
echo "scale=4; $1+$2" | /usr/bin/bc -l
echo arg zero = $0
echo ""
```

`[esc-colon-wq to save and quit]`

```
% chmod +x hw
```
Then use `globus-job-run` to run your files. If your Globus host (*hostname*) is also the machine where your `hw` file is located:

Syntax:

```
% globus-job-run  <hostname> <path/executable> <arguments>
```
For example, if `hw` is in your current directory:

```
% globus-job-run denali.mcs.anl.gov hw 50 60
```
Prints:

```
Hello Globus World from denali.mcs.anl.gov
Sum is 110
arg zero =  ./hw
```

Staging Files

If your executable file is on your home machine (the machine from which you are issuing Globus commands) and you wish to run that executable on a remote machine, you will need to *stage* your executable. To stage your executable file `hw`, which resides on your home

machine, over to a remote machine, execute it, and automatically remove the staged copy after the program has finished, use the `-s` (`-stage`) option:

Syntax:

```
% globus-job-run <remote host> -stage  <path/executable> <arg arg>
```

For example, if `hw` is in your current directory:

```
% globus-job-run evelyn.nas.nasa.gov -stage  hw 4 6
```

prints something like:

```
Hello Globus World from evelyn.nas.nasa.gov
sum is 10
arg zero = /u/walatka/.globus/.gass_cache/globus_gass_cache_984165472
```

The `-stage` option temporarily puts your executable on a remote machine. Note that `$0` returned the path to the staged executable. To copy files to or from a remote machine, as opposed to just staging them, use `globus-rcp`. See page 36.

Numbering Arguments

Note the numbering of the command line elements. `$0` is the executable that is being passed to the Globus command. The next element (50) is `$1`, and so forth. The host name can be retrieved with the Globus command `globus-hostname`. Notice how the command and variables are used in the file you wrote.

```
% globus-job-run     denali.mcs.anl.gov       hw  50  60
                                               $0  $1  $2
```

Subjobs and Multiple Commands

Subjobs

Now make some new files, `jobA`, `jobB`, and `jobC`. You can use `cat` to view your file.

```
 % vi jobA
#!/bin/csh -f

set tools_bin = `$GLOBUS_INSTALL_PATH/bin/globus-tools-path -bindir`

echo ""
echo -n "I am job A on " ; $tools_bin/globus-hostname
echo -n "Sum is "
echo "scale=4; $1+$2" | /usr/bin/bc -l
echo arg one = $1
echo arg two = $2
echo ""
```

Back on the Unix command line:

```
    % sed 's/job A/job B/' jobA > jobB
    % sed 's/job A/job C/' jobA > jobC
    % chmod +x jobA jobB jobC
    % ls -l jobA jobB jobC
    -rwx------    1 walatka  mrj              249 Mar  8 18:04 jobA
    -rwx------    1 walatka  mrj              249 Mar  8 18:33 jobB
    -rwx------    1 walatka  mrj              249 Mar  8 18:33 jobC
```

To start a multi-request with subjobs, the `-:` delimiter denotes the start of each subjob. (If no path is given, and files are not staged, the executables are assumed to reside in your home directory on each of the remote hosts.) The `-np` option states the number of processors to use.

```
% globus-job-run -args 3 5 \
-: evelyn.nas.nasa.gov   -np 2 -stage ./jobA \
-: pitcairn.mcs.anl.gov  -np 2 -stage ./jobB 8 9 \
-: denali.mcs.anl.gov          -stage ./jobC
```

prints something like:

```
I am job A on evelyn.nas.nasa.gov
Sum is 8
arg one =  3
arg two =  5


I am job A on evelyn.nas.nasa.gov
Sum is 8
arg one =  3
arg two =  5


I am job B on pitcairn.mcs.anl.gov
Sum is 17
arg one =  8
arg two =  9


I am job B on pitcairn.mcs.anl.gov
Sum is 17
arg one =  8
arg two =  9


I am job C on denali.mcs.anl.gov
Sum is 8
arg one =  3
arg two =  5
```

[As of 2001-03-08, you may get a DUROC error message.] Parameters can be given job-wide (with the -args option). These parameters can be overridden at a per-subjob level. In the above example, the executable running on pitcairn gets two other arguments to sum up. You may find that your results vary. Experiment with changes to the command.

Multiple Commands

To run multiple commands on the same machine, use this syntax:

```
% globus-job-run <hostname> /bin/sh -c \
"command1  ; command2"
```

For example:

```
% globus-job-run ico16.mcs.anl.gov /bin/sh -c \
"cd foo ; ls -l"
```

Batch Job Submissions

`globus-job-submit` is similar to `globus-job-run`, but is intended for batch job submission. You can remotely submit a job to a scheduling manager such as PBS, log out, and log back in later to collect the output. `globus-job-submit` defaults to caching the output on the remote machine, for later manual retrieval using the `globus-job-get-output` command. Via a command line interface, you can submit jobs to any single resource; multi-resource requests are not supported. To see the options use `globus-job-submit -help`. The `-maxtime` option sets the maximum number of minutes to run the job—wall or cpu time, depending on how the host is set up. (`globus-job-submit` is sometimes broken.)

Globus-job-submit Examples

If you copy and paste, beware of missing hyphens (-) and misplaced backslashes (\).

```
% vi pi-test
#!/bin/csh -f
echo "The value of pi to $1 decimal places is approximately "
echo "scale=$1; 4*a(1)" | /bin/bc -l

% chmod 740 pi-test
% globusrun -a -r evelyn.nas.nasa.gov
% globus-job-submit evelyn.nas.nasa.gov/jobmanager-pbs \
-maxtime 20 -stage ./pi-test 30
```

Prints, for example:

```
https://evelyn.nas.nasa.gov:13196/10323394/974139820/
%
```

The https URL returned by `globus-job-submit` is the job contact string, which uniquely identifies the job. Make a note of the url before moving on (save it in a file). The extra `%` denotes that this command returns immediately after submission. Use the URL to get status/output from your job. If `globus-job-get-output` doesn't work, see also Retrieving Output, below.

Syntax:

```
% globus-job-status <url>
% globus-job-get-output <url>
```

For example:

```
% globus-job-get-output \
https://evelyn.nas.nasa.gov:13196/10323394/974139820/
```

Another globus-job-submit example:

```
% globus-job-submit ico16.mcs.anl.gov -np 32 -maxtime 120 jobA 6 8
```

prints something like:

```
https://ico16.mcs.anl.gov:60106/17916/942265377/
%
```

This example submits a request for 120 minutes of compute time on 32 nodes to the scheduler on the IBM SP at Argonne National Laboratory. If the scheduler is the default job manager on your requested resource, you do not need to specify it.

globus-job-submit caches the output at the remote site. You can override this by specifying -stdout and -stderr. See Retrieving Output, below. See also http://www.globus.org/v1.1/programs/globus-job-submit.html .

globusrun

The globusrun command runs scripts written in the Globus Resource Specification Language (RSL). Below, RSL is discussed briefly; see also the chapter on RSL, page 30. The RSL script may be entered on the command line, or saved as a file. Here is an RSL script that could be saved as hello.rsl.

```
&  (count=1)
   (executable=/bin/echo)
   (arguments="Hello Globus World")
```

Using the globusrun Command

To run the hello.rsl script shown above, use globusrun.

```
% globusrun -s -r <host name> -f hello.rsl
```

After the Unix prompt, enter globusrun with the -s option to send output to stdout. Use the -r option to indicate that a resource name follows. The -f option indicates a filename follows, then your filename. For example:

```
% globusrun -s -r evelyn.nas.nasa.gov -f hello.rsl
```

"Hello Globus World" ought to appear on your screen. Are you having problems? See Ready to run, above.

Optionally, a resource name takes the form *<host name>*/jobmanager-*<scheduler type>* if you want to specify a service other than the default, which is usually fork. For example, a service name of "jobmanager-pbs" would be used to submit a job to a PBS scheduler.

globusrun takes your RSL script and parcels its instructions out to specified resources. The RSL script by itself is a resource specification, not an executable. In addition to starting jobs, globusrun can be used to list previously started jobs, query the status of previously started jobs, parse RSL request strings, and perform authentication tests to GRAM gatekeepers. Type globusrun -help to see the options.

Note: The globusrun command does not use the -stage option; with globusrun, the –s option starts a GASS server (see , page 36) for moving files. The variable $(GLOBUSRUN_GASS_URL) can be used to access files local to the submission machine via GASS; see examples below. Also, you can use globus-rcp, to move files from one machine to another.

More globusrun Examples

Example One

```
% globusrun -s -r pitcairn.mcs.anl.gov  '&(executable=my_prog'
```

Resolve the contact string for the fork job manager service on host pitcairn, and then submit the program my_prog to that resource. The standard output and standard error of my_prog will be displayed by globusrun.

Example Two

```
% globusrun –s -r ico16.mcs.anl.gov/jobmanager-easymcs \
'&(executable=$(GLOBUSRUN_GASS_URL)/usr/local/my_prog) \
(stdout=/tmp/output1)(stderr=/tmp/error1)(count=3)'
```

The –s option starts a GASS server; see page 36. Submit three copies of the local program /usr/local/my_prog to the EASY scheduler at ico16.mcs.anl.gov, and send the output to the remote /tmp directory.

Example Three

```
% globusrun –w -r pitcairn.mcs.anl.gov \
'&(executable="$(GLOBUS_TOOLS_PATH)/bin/globus-url-copy") \
(arguments="file:/tmp/myout.1" $(GLOBUSRUN_GASS_URL)/tmp/myout.1)'
```

Copy a file from pitcairn to the local host, using the GASS transfer protocol. The -w option gives the remote process write permission on the local GASS server started by globusrun. Note that because the colon (:) is a reserved RSL character, the first argument must be quoted. See the next chapter and http://www.globus.org/gram/rsl_spec1.html for more information.

Checking, Killing, Retrieving, and Cleaning Jobs

Is My Job Running? What is My Job ID?

In general, if your Unix prompt has not returned after a globus-job-run or globusrun command, your job is still running. You can use Globus to remotely run a Unix command such as ps, which prints information about active processes. Remote access without login!

```
% globus-job-run <hostname> /bin/ps  –u <username>"
```

Example:

```
% globus-job-run rogallo.larc.nasa.gov /bin/ps –u walatka
```

Note: the location and command–line arguments of ps may vary across systems.

To query the status of a batch job, enter globus-job-status followed by the job contact string URL (see Batch Job Submissions above). For example:

```
% globus-job-status \
https://ico16.mcs.anl.gov:60106/17916/942265377/
```

The different states are PENDING (waiting in the queue), ACTIVE (running), SUSPENDED, DONE, and FAILED.

See also http://www.globus.org/hbm/heartbeat_spec.html for information on the HeartBeat Monitor, a simple, highly reliable mechanism for monitoring the state of processes.

Retrieving Output

You can retrieve the cached output from a batch job while the job is running, or after; use the https URL that was returned when you submitted the job.

```
% globus-job-get-output \
https://ico16.mcs.anl.gov:60106/17916/942265377/
```

should print something like:

```
my_id 0 numprocs 32, sum = 14 : now sleeping for 90 minutes
```

```
my_id 1 numprocs 32, sum = 14 : now sleeping for 90 minutes
my_id 2 numprocs 32, sum = 14 : now sleeping for 90 minutes
[...]
```

Sometimes `globus-job-get-output` does not work or is not available. In that case, you could use Globus to access the remote machine without logging in. For example (substitute your hostname):

```
% globus-job-run ico16.mcs.anl.gov /bin/sh -c \
"cd .globus/.gass_cache ; ls -l"
```

Prints something like:

```
-rwxr-xr-x   1 walatka  mrj 1223 Dec 20 15:31 globus_gass_cache_977355133
-rwxr-xr-x   1 walatka  mrj   31 Dec 20 15:31 globus_gass_cache_977355134
```

The files whose names start with `globus_gass_cache_` (and with the right date-time) are the output from your `globus-job-submit` command. The number before the date gives the size of the file in bytes. Note that in some cases, `not_ready` will be appended to the filenames. To copy the output into your current directory, use `globus-rcp`, with your hostname followed by a colon (:) and the path `.globus/.gass_cache/` followed by the filename, followed by a space and then a dot (dot=current directory). The filename is `globus_gass_cache_` appended with the last number in the URL that was returned when you submitted the job (you saved that, right?).

Syntax:

```
% globus-rcp hostname:.globus/.gass_cache/filename .
```

For example:

```
% globus-rcp \
turing.nas.nasa.gov:.globus/.gass_cache/globus_gass_cache_980883383 .
```

See more about moving remote files in GASS Options, page 36.

Killing a Job

If you interrupt the `globusrun` or `globus-job-run` process (by entering CTRL-C or sending it a SIGINT), your jobs should automatically be canceled.

The job cancellation routines (triggered by CTRL-C or SIGINT) have a timeout to allow an intelligent job to cleanup. Therefore, you should expect some delay before the globusrun or globus-job-run process exits. Don't interrupt this cleanup by typing CTRL-C again.

Canceling/Cleaning a Job

To cancel a job, use `globus-job-cancel` with the contact string; for example:

```
% globus-job-cancel \
https://ico16.mcs.anl.gov:60106/17916/942265377/
```

which prints:

```
Are you sure you want to cancel the job now (Y/N) ? y
Job canceled.
NOTE: You still need to clean files associated with the job by
running globus-job-clean <jobID>
```

Cached output from the job is not removed. You can still use `globus-job-get-output` to retrieve it after you have cancelled the job. The `-force` option bypasses the interactive prompt.

To cancel a job if it is still running and remove the cached output from the remote host, enter:

```
% globus-job-clean https://ico16.mcs.anl.gov:60106/17916/942265377/
```

which prints:

```
WARNING: Cleaning a job means:
- Kill the job if it still running
- Remove the cached output on the remote host
Are you sure you want to cleanup the job now (Y/N) ? y
Cleanup successful.
```

Note: Globus does not routinely clean up the cached output files of completed jobs; you need to periodically `rm globus_gass_cache_*` in the `.globus/.gass_cache` directory on the machine to which you submitted the job(s).

Default Job Managers

Old way:

```
% globus-job-run   evelyn.nas.nasa.gov/jobmanager-fork
```

New in version 1.1.3:

```
% globus-job-run   evelyn.nas.nasa.gov
```

Resource contact strings, also called "job contact strings," are the part of the Globus command line where you specify which host to use, for example, `evelyn.nas.nasa.gov`. Resource contact strings for default job manager services are no longer routinely suffixed with `/jobmanager-<service-name>`. For example, resource contact strings for "fork" services are no longer suffixed with `/jobmanager-fork` if fork is the default job manager. If you do not specify a job manager in your `globusrun`, `globus-job-run`, or `globus-job-submit` command, your job will be submitted to the default job manager on the remote system, which is often fork.

Use `globus-setup-test` to see what job managers are available on your resource. To see which job manager is the default, check your Grid's web pages, usually under Resources. Alternately, you can run a globus job to check the Globus files. Substitute your resource for `whitcomb.larc.nasa.gov` in the example below; the default job manager name will follow `-type` in the response:

```
% globusrun -s -r whitcomb.larc.nasa.gov \
   '&(executable=/bin/cat) \
     (arguments=$(GLOBUS_DEPLOY_PATH)/etc/globus-services)'
```

If you want to specify a non-default jobmanager, do so by appending `/jobmanager` and the name of the job scheduler, such as `/jobmanager-lsf` or `/jobmanager-pbs`, to the hostname. For example:

```
% globus-job-run   evelyn.nas.nasa.gov/jobmanager-pbs \
    /bin/echo   "Hello World."
```

What Happens When You Submit a Job

This is a complicated process, but in streamlined form it looks like this:

1. You run `globusrun` with an RSL script (or `globus-job-run` or `globus-job-submit`) and specify where to run.

2. The Globus command contacts something called a "gatekeeper" on the remote host and performs mutual authentication. This means that the remote host knows who you are, and you know who the remote host is.

3. The gatekeeper (on the remote host) contacts a job manager service with your request. The job manager will decide how to run your job depending on the service name that is used. If you did not specify a service, the default service is used, usually "fork," which will run the job immediately. If you specified a job scheduler (e.g., /jobmanager-pbs), your job will be submitted to the scheduler and will run as the scheduler allows. See page 28, "Default Job managers."

4. Your job completes.

Globus Resource Allocation Manager

The Globus Resource Allocation Manager (GRAM) authenticates and processes the requests for resources for remote application execution, and allocates the required resources. It also prints updated information regarding the capabilities and availability of computing resources to the Grid Information Service (see page 34).

GRAM provides an API for submitting and canceling a job request, as well as checking the status of a submitted job. You write the specifications on the command line, or in a Resource Specification Language (RSL) script. The specifications are processed by GRAM as part of the job request.

Your program can call the GRAM C API directly and supply it with an RSL.

GRAM manages jobs that use only one resource. For more information, see http://www.globus.org/gram For a tutorial, see http://www.globus.org/gram/tutorial_gssapi_ssleay.html

The commands in this chapter are "GRAM tools."

Dynamically Updated Request Online Co-allocator

Dynamically Updated Request Online Co-allocator (DUROC) co-allocates multiple resources and manages multiple GRAM jobs. See http://www.globus.org/duroc for more information.
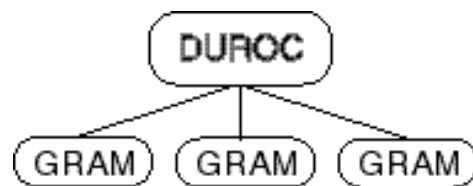
Chapter 6   Globus Resource Specification Language (RSL)

The Globus Resource Specification Language (RSL) provides a common language to describe jobs and the resources required to run them. The Globus Resource Allocation and Management (GRAM) and DUROC components both use RSL.

Practical RSL

The easiest way to learn RSL is to start with examples and modify them to suit your needs. The `globus-job-run` tool's `-dumprsl` option can convert command-line arguments into RSL. Using this feature, you can create sample RSL expressions. Again, if you copy and paste these examples to your command line, watch out for missing hyphens (`-`) or misplaced backslashes (`\`).

For example:

```
% globus-job-run -dumprsl evelyn.nas.nasa.gov \
/bin/echo "Hello, Globus world."
```

Prints:

```
&(executable="/bin/echo")
  (arguments="Hello, Globus world.")
```

The command tells `globus-job-run` to print an RSL expression describing the job it would have submitted if the `-dumprsl` option had not been specified. This RSL script defines the program to be executed (`/bin/echo`) and the arguments to the program (the string `"Hello, Globus world."`). Any `globus-job-run` command can be converted to RSL in a similar manner.

Once you have an RSL expression, you may use the `globusrun` command to submit the corresponding job. The `globusrun -r` option specifies that a resource name follows. The following command demonstrates how to use the RSL expression above.

```
% globusrun -r evelyn.nas.nasa.gov \
'&(executable=/bin/echo)(arguments="Hello, Globus world.")'
```

Note that white space and newline characters within an RSL string are typically ignored unless they appear within quotes. Experimenting with other `globus-job-run` commands will allow you to infer how RSL works. Keep in mind, however, that using the `globus-job-run` command in this way will show you only a limited set of RSL's features.

RSL Syntax

This section provides an abbreviated overview of RSL, covering the most common uses. Some advanced features of RSL are presented here in simplified form or are not represented here at all.

The simplest RSL expression looks something like the following.

```
&(executable=/bin/ls)
```

The expression above simply specifies an executable program to be run. No arguments are provided, and no specification is made regarding the resource on which this job should be run. This type of simple expression is known as a *relation*. Relations associate an attribute name with a value. In this case, the attribute executable is associated with the string /bin/ls. This tells GRAM and DUROC tools that /bin/ls is the program to be run when the job is executed.

While the expression above is syntactically correct, RSL expressions nearly always include more than one relation. Relations can be combined in a single expression using a *conjunction*. A conjunction looks like this:

```
&(relation 1)(relation 2)…(relation n)
```

Any number of relations can be listed after the & symbol. All relations must appear within parenthesis. All of the relations are assumed to apply to a single resource. For example:

```
&(executable="/bin/ls")(arguments="-l")
```

GRAM Attributes

Some of the relations that GRAM and DUROC tools understand are listed below. Note that RSL attribute names are case-insensitive; for example, you can use (maxcputime=10) rather than (maxCpuTime=10). See also http://www.globus.org/gram/gram_rsl_parameters.html

Relation	Example	Meaning
(executable=string)	(executable="/bin/ls")	An executable program file
(arguments=list)	(arguments= -l -d .)	Arguments to the program
(directory=string)	(directory="/tmp")	The job's active directory
(environment=list)	(environment= (LD_FLAGS "-o") (COPTS "-g -c"))	Environment variables to be set for the job
(stdin=string) (stdout=string) (stderr=string)	(stdin=/tmp/cmds)	Where stdin, stdout, or stderr should come from/go to (can be files or URLs)
(count=integer)	(count=5)	The number of processes to be run (default is 1)

`(hostCount=integer)`	`(hostCount=2)`	The number of CPUs on which to run the processes
`(maxCpuTime=integer)`	`(maxCpuTime=10)`	Maximum CPU run time in minutes
`(maxMemory=integer)`	`(maxMemory=100)`	Maximum memory per process in megabytes
`(minMemory=integer)`	`(minMemory=5)`	Minimum memory per process in megabytes
`(jobType=string)`	`(jobType=condor)`	Specifies how to start the processes within the job

Resource Co-allocation

The DUROC component of the Globus Toolkit allows resources to be co-allocated for a single job. That is, multiple resources can be allocated simultaneously for a single job and used in parallel. Co-allocation is specified in RSL using the + symbol. For example:

```
+(&(resourceManagerName=flash.cs.uchicago.edu)
        (count=1)
        (executable= my_app1)
    )
    (&(resourceManagerName=tfglobus.sdsc.edu)
        (count=2)
        (executable=my_app2)
    )
```

The above example specifies a single job that will involve one process running on a computer at the University of Chicago and two processes (using a different executable file) running on a computer at the San Diego Supercomputer Center.

Additional Information

This section presented the most commonly used features of RSL. Advanced features are described further in the following places.

http://www.globus.org/gram/gram_rsl_parameters.html	Complete parameter list
http://www.globus.org/gram/	Details on GRAM and RSL
http://www.globus.org/duroc/	Details on DUROC and RSL

RSL With Shell Script

The examples in this section show a brief Bourne shell script that invokes an RSL script that invokes a Bourne shell script that runs an application, Overflow-d2.

The `globus-sh-exec` command allows the user to run a script on a remote machine without having to worry about correct paths to various UNIX commands. See http://www.globus.org/details/programs/globus-sh-exec.html

`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.

The First Shell Script

This shell script, named `platform_independent.sh1`, uses `globusrun` with the RSL script named `platform_independent.rsl`.

```
#! /bin/sh
$GLOBUS_PATH/globusrun -s -r evelyn.nas.nasa.gov -f \
platform_independent.rsl &
```

The RSL File

This RSL script is named platform_independent.rsl

```
&
(count=4)
(maxTime=20)
(jobType=single)
(directory=/scratch1/yarrow/RSLtest)
(executable=/usr/prg/pkg/globus/1.1.3/tools/mips-\
sgi-\irix6.5/bin/globus-sh-exec)
(arguments=/u/yarrow/tc/Globus/platform_independent.sh2)
(stdout=platform_independent.out)
(stderr=platform_independent.err)
(environment=(GASS_URL $(GLOBUSRUN_GASS_URL))
              (OSNAME $(GLOBUS_HOST_OSNAME)))
```

The Second Shell Script

This shell script, named platform_independent.sh2 is called by the RSL above.

```
SOURCEFILE=/scratch1/yarrow/X38.globus.6.copy/all.tar
EXECPROG=/scratch1/yarrow/Overflowd2.0.7.restore/Overflowd2.0.7/\
overflowd2.${OSNAME}
GLOBUS_PATH=/usr/prg/pkg/globus/1.1.3/tools/mips-sgi-irix6.5/bin
$GLOBUS_PATH/globus-url-copy ${GASS_URL}${SOURCEFILE} - | tar \
xf - $GLOBUS_PATH/globus-url-copy ${GASS_URL}${EXECPROG} - | cat > \
overflow-exec
chmod u+x overflow-exec
mpirun -np 4 ./overflow-exec
```

The shell script is getting `mpirun` to start Overflow-d2 as a four-processor parallel job.

Chapter 7   Grid Information Service

The Grid Information Service (GIS) contains information about the state of the Grid infrastructure. GIS is a general term that includes the Metacomputing Directory Service (MDS). "GIS" and "MDS" are often used interchangeably; definitions are still being resolved. Use the Globus `grid-info-*` commands to gather information from the GIS. This chapter does not address initialization or population of information into the GIS.

Terminology

GIS (MDS) is a service that allows the storage of information about the state of the Grid infrastructure. One of its services is to publish information via LDAP (Lightweight Directory Access Protocol), a protocol used to locate resources in a network.

The GIS information service has the ability to function as a white pages directory—for retrieving information associated with a particular name ("distinguished name"). Examples for such lookups are the number of CPUs and the operating system associated with a particular machine. The GIS also functions as a yellow pages directory—for retrieving a list of categorized entities. Such categories are defined by "object classes." Examples of such categories are lists of computers or people.

A powerful extension to the white and yellow pages function of the directory is the ability to augment the lookups with sophisticated Boolean search filters.

The most important change in the GIS/MDS from 1.1.2 to 1.1.3 is that the Globus group no longer runs a centralized MDS server. That means that each site optionally will have its own organization server, which is used for any searching commands.

Globus version 1.1.3 introduces two subsets of GIS: the Grid Resource Information Service (GRIS) and the optional Grid Information Index Service (GIIS). The new GIS model is push rather than pull, and will cache information from resources within a particular organization. For more information, see: http://www.globus.org/mds, and http://www.globus.org/toolkit/download/info-113.html

Tools

See http://www.globus.org/mds for GIS search tools other than the `grid-info-*` commands.

grid-info-search

The command `grid-info-search` allows searches on the GIS server based on search filters that conform to LDAP searches. The format is:

```
grid-info-search [ options ] <search filter> [attributes ]
```

To see the options available, and find out your host and port information, type:

```
 % grid-info-search -help
```

To list all distinguished names along with all attributes of the compute resources that are stored in the directory under the local organization:

```
 % grid-info-search "(objectclass=GlobusComputeResource)"
```

To view information from other organizations, you must know the host and port of the GIIS server at that organization, and include the `-mdshost` or `-h <host>`, `-mdsport` or `-p <port>`, and `-b <base>` options to `grid-info-search`. For example:

```
 % grid-info-search -h gis.ipg.nasa.gov -p 389 -b "o=globus,c=us"
 "(objectclass=GlobusComputeResource)"
```

To list all distinguished names of the compute resources that are stored under the local organization, along with the operating system type of each:

```
 % grid-info-search  "(objectclass=GlobusComputeResource)" dn ostype
```

To list the systems that have contact strings, and various types of job managers:

```
 % grid-info-search -h gis.ipg.nasa.gov -p 389 \
 "(objectclass=GlobusServiceJobManager)"
```

To restrict the attributes that are output, list them at the end of the `grid-info-search` command. Search filters are specified in Polish notation where `&` is the Boolean *and* operator and `|` is the Boolean *or* operator. It is a good practice to restrict the searches to objectclasses of interest in order to minimize the duration it takes for a query to return. As a general rule of thumb, the more precise the query, the shorter the response time. Tip: you also can use the Unix `grep` command to narrow your `grid-info-search` results. For example, to list hostnames:

```
 % grid-info-search "(objectclass=*)" | grep "objectname=service"
```

The new org DN namespace is the "domain component" explosion of the site's DNS domain under the "o=Grid" root; e.g., mcs.anl.gov becomes "dc=mcs, dc=anl, dc=gov, o=Grid" or isi.edu becomes "dc=isi, dc=edu, o=Grid".

Chapter 8   Accessing Remote Data

Globus Access to Secondary Storage (GASS) provides programs and C APIs for remotely accessing data. This chapter describes how to start a GASS server, and how to use `globus-url-copy` to transfer data to and from GASS servers. GASS programs allow data to be easily transferred from one machine to another.

GASS Options

GASS provides programs and C APIs for remotely accessing data. Several commands and programs support secondary storage functionality.

globus-rcp

`globus-rcp` is the recommended command for remote file transfer. The `globus-rcp` is the functional equivalent of the Unix `rcp` program, but uses GASS to perform its remote data transfer. You do not need to start a `globus-gass-server` by hand to use `globus-rcp`. Instead, `globus-rcp` will use GRAM to automatically start and stop remote GASS servers and clients for you. An example:

```
% globus-rcp evelyn.nas.nasa.gov:/tmp/foo /tmp/bar
```

Here the /tmp/foo file on evelyn is copied to the /tmp/bar on the machine running this command.

globus-gass-server

The `globus-gass-server` program is used to make the data on one computer available to remote clients. To run, enter:

```
% globus-gass-server &
```

prints

```
https://evelyn.nas.nasa.gov:20143
```

This sets up your machine to serve files to remote clients. The https URL that is output by the GASS server contains the host and port on which this server is listening for requests. It is used by clients to transfer data to and from this machine via the GASS server. The ampersand (`&`) runs the process in the background, returning your Unix prompt.

globus-url-copy

The `globus-url-copy` command is used to remotely access files from GASS servers, or from other servers that speak the http or https protocols (e.g., a web server). The general form of `globus-url-copy` is:

```
% globus-url-copy <fromURL> <toURL>
```

Examples:

```
% globus-url-copy https://evelyn.nas.nasa.gov:20143/tmp/foo \
file:/tmp/bar
% globus-url-copy https://evelyn.nas.nasa.gov:20143/tmp/foo -
```

In these examples, the `<fromURL>` is composed of the base URL that was returned by the `globus-gass-server` (`https://evelyn.nas.nasa.gov:20143`), followed by the path of a file accessible to `globus-gass-server` (`/tmp/foo`). In the first example, the `/tmp/foo` file is remotely transferred from `evelyn` to a local file named `/tmp/bar` (`file:/tmp/bar`). In the second example, the single dash (`-`) copies the contents of the remote `/tmp/foo` file to `stdout`.

Chapter 9   Globus Commands

This chapter shows you the commands available in Globus 1.1.x. On your Unix command line, type a Globus command followed by the `-usage` option to get help. The commands begin with either `globus` or `grid`. You will find information on the complete set of command-line programs in the Globus Toolkit at http://www.globus.org/v1.1/programs.

Remember that you can use Globus to run Unix commands or your own executables on remote machines without remote login. Refer to the command information in Chapter 5, page 18.

Globus 1.1.3 Tools

The following tools are available in *<your-globus-tools-path>* in the Globus Toolkit version 1.1. For usage information, including options available, type the command name with the `-usage` option. See also http://www.globus.org/v1.1/programs.

Security

`grid-cert-request`	Creates a new certificate request and private key. Ask your Globus administrator whether to use this command or a local alternative.
`grid-cert-info`	Displays certificate information.
`grid-cert-renew`	Creates a new key and renewal request for a Globus certificate.
`grid-change-pass-phrase`	Changes the pass phrase that protects your private key.
`grid-proxy-init`	Creates a proxy certificate that can be used for authentication without having to reenter the protecting pass phrase for each resource.
`grid-proxy-info`	Displays proxy certificate information. Use `-all` option.
`grid-proxy-destroy`	Removes your proxy certificate.

Job Submission

`globusrun`	Runs a single executable on a remote site with RSL script.
`globus-setup-test`	Verifies available job managers and your setup of credentials.
`globus-job-cancel`	Cancels a job previously started using `globus-job-submit`.

`globus-job-run`	Allows you 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 from the job execution. (If broken, see page 26.)
`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 like PBS).

Information Services

`grid-info-add`	Modifies the GIS server based on the contents of input file.
`grid-info-host-search`	Searches the GIS on a specified machine.
`grid-info-remove`	See `grid-info-add`.
`grid-info-search`	Searches the GIS.
`grid-info-site`	Searches all machines associated with a given site.
`grid-info-update`	See `grid-info-add`.

Other Tools

`globus-hostname`	This is a simple shell script that acts like the Unix `hostname` command.
`globus-hostname2contacts`	Converts a hostname to a list of resource manager contact strings. May be broken in 1.1.3 and 1.1.4.
`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`	Allows the user to run a script on a remote machine without having to worry about correct paths to various UNIX commands.
`globus-version`	Shows version number.
`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 (pthreads, debug, 64-bit, support for SHM, ...)

`globus-install-path`	Prints the full path to the Globus install tree.
`globus-rcp`	Remote copies using GASS and Globus submission. Many options.
`globus-tools-path`	Prints the full path to the tools directory in the Globus install tree, tailored for the current architecture.
`globus-services-path`	Prints the full path to the services directory in the Globus install tree, tailored for the current architecture.
`globus-tilde-expand`	Expands the leading tilde sign (~) (and the specified username if provided) to the full path of your home directory.

Chapter 10 Bibliography and Reference

Web and Email Resources

- http://www.globus.org
  the central site

- http://www.globus.org/v1.1/programs

  Man pages on Globus commands

- http://www.globus.org/about/faq.html
  Frequently Asked Questions.

- discuss@globus.org
  To ask question or discuss approaches in developing grid-aware
  applications via the Globus Toolkit. Subscribe by sending mail to
  majordomo@globus.org with `subscribe discuss` in the body of the
  message

- http://www.nas.nasa.gov/Software/p2d2/
  NAS debugging system for parallel and distributed programs

- documentation@globus.org
  Report errors in this document

- http://www.globus.org/about/contacts.html
  List of Globus contacts

- http://www-unix.globus.org/support/problem_submit.pl
  Submit Globus problems; bugs or questions

Technical Papers

Proc. 8th IEEE Symposium on High-Performance Distributed Computing, "Grids as
Production Computing Environments: The Engineering Aspects of NASA's Information
Power Grid", William E. Johnston, Dennis Gannon and Bill Nitzberg, 1999, HPDC8

See http://www.globus.org/research/papers.html for a large selection of papers.

The Grid Book

The Grid: Blueprint for a New Computing Infrastructure, Edited by Ian Foster and Carl
Kesselman: http://www.mkp.com/books_catalog/1-55860-475-8.asp

Chapter 11 Glossary

DUROC

Dynamically Updated Request Online Co-allocator

GASS

Global Access to Secondary Storage (remote file management)

GIS

Grid Information Service (formerly/also MDS, Metacomputing Directory Service) for locating and distributing characteristics of resources. Accesses the white pages and yellow pages served by LDAP. Used with a set of commands each beginning with `grid-info-`. See http://www.globus.org/mds

Globus

The Globus Project is a community effort, led by Argonne National Laboratory and the University of Southern California's Information Sciences Institute. Globus is developing the basic software infrastructure for computations that integrate geographically distributed computational and information resources. http://www.globus.org

GRAM

Globus Resource Allocation Manager
http://www.globus.org/gram

Grids

Widely distributed networks of high-performance computers, stored data, instruments, and collaboration environments.
http://www.globus.org/research/testbeds.html

HBM

HeartBeat Monitor that checks if a process is alive. The Globus Heartbeat Monitor (HBM) is designed to provide a simple, highly reliable mechanism for monitoring the state of processes. The HBM is designed to detect and report the failure of processes that have identified themselves to the HBM.
http://www.globus.org/hbm/heartbeat_spec.html

help

Type any command with the `-usage` or `-help` option for online help.

LDAP

Lightweight Directory Access Protocol, for accessing information about hardware, software, and status in a networked environment. Directory-server software available from numerous sources. http://www.globus.org/mds

LSF

   Load Sharing Facility, batch processing software.

Maxtime

   Maximum time a job should be allowed to run. Refers to wall clock time on some systems, CPU time on others.

MDS

   Metacomputing Directory Service, an aspect of GIS. http://www.globus.org/mds

MPI

   Message Passing Interface, the industry-wide standard protocol for passing messages between parallel processors.

MPICH

   A portable MPI model implementation.

MPICH-G2

   Message Passing Interface for wide-area networks; the Globus version of MPI. MPICH-G2 is in Globus version 1.1.4 and up; previous versions use MPICH-G. See http://www.niu.edu/mpi/

PBS

   Portable Batch System. Flexible batch processing software that operates on networked, multi-platform Unix environments, including heterogeneous clusters of workstations, supercomputers, and massively parallel systems. See http://www.pbspro.com/, or http://pbs.mrj.com/.

resources

   Computers, instruments, and immersive environments; machines.

RSL

   Resource Specification Language. A language that can be used with tools such as `globusrun` to specify resource requirements, such as what executable(s) to use, arguments to pass, etc.

usage

   Type any command with the `-usage` or `-help` option for online help.

Chapter 12 Index

.