Job Submission Tutorial

PPARC Summer School,
NeSC Edinburgh
Guy Warner, NeSC Training Team

The aim of this tutorial is to walkthrough submitting programs (called jobs) to the NGS and retrieving the output. This tutorial does not require any knowledge of programming and the programs used are already in your account. An account has already been created for you on the NGS node at the Rutherford Appelton Laboratory and all the jobs in this tutorial will be sent there to run. All commands (highlighted like **this**) should be entered in a terminal window on "pub-234". Through-out this document examples of output have been inserted and they appear in a box like

```
this
```

1. At the end of the previous tutorial you should have destroyed your grid-proxy. All the job submission and control programs used by the NGS are part of the Globus Toolkit (version 2.4) and depend on your having a valid proxy. Launch a new proxy with

   grid-proxy-init

2. The simplest command for job submission is globus-job-run. This minimum parameters used by this command are where to send the job and what the program to run is. Submit your first job with the command

   globus-job-run grid-data.rl.ac.uk /bin/hostname -f

   grid-data.rl.ac.uk is the head node (a node users can directly access) at R.A.L.

   ```
   [gcw@lab-07 gram]$ globus-job-run grid-data.rl.ac.uk /bin/hostname -f
   grid-data.rl.ac.uk
   ```

   Since you directly told globus-job-run to run on this node, the result should hardly surprise you.

3. The next step is therefore to use a system of accessing all the nodes at R.A.L. To do this submit the job to a job queue running on the head node. When a suitable node becomes available your job will be submitted to from this queue. It is possible for multiple queues to exist at a single site so when you tell globus-job-run where to run you provide the name of the queue. This time run the command

   globus-job-run grid-data.rl.ac.uk/jobmanager-pbs /bin/hostname -f

   where jobmanager-pbs is the name of the queue.

   ```
   [gcw@lab-07 gram]$ globus-job-run grid-data.rl.ac.uk/jobmanager-pbs /bin/hostname -f
   grid-data12.rl.ac.uk
   ```

4. You may have noticed that globus-job-run waits for your job to complete before exiting. The standard output of the job is sent directly to your terminal. Whilst this is not a problem for very short and simple jobs, this is not a good system for long jobs. Long jobs need to be submitted and occasionally checked, and then the output may be retrieved, possibly many hours or days later. The solution to this is to use globus-job-submit, a command very similar to globus-job-run except that it outputs a unique identity (uid) string for your job and then exits. The job is still running and other commands exist for checking on the job, retrieving the job's output and cleaning up after the job. Enter the command

globus-job-submit grid-data.rl.ac.uk/jobmanager-pbs /bin/hostname -f

All subsequent commands depend on this uid to identify the job. If you are not familiar with Linux use <Ctrl>-<insert>to copy highlighted text and <Shift>-<Insert>to paste. In the following commands ***replace <uid> with the uid of your job***. To check on your job and find its status use the command

globus-job-status <uid>

Repeat this command every few seconds until your job has achieved the status of "Done". In this context "Done" means your job has finished as far as the grid middleware is concerned, it does not necessarily mean your job did what you expected it to do. Next retrieve the standard output with

globus-job-get-output <uid>

and finally clean up any temporary files created by your job with

globus-job-clean <uid>

Answer "Y" when asked if you are sure.

```
[gcw@lab-07 gram]$ globus-job-submit grid-data.rl.ac.uk/jobmanager-pbs /bin/hostname -f
https://grid-data.rl.ac.uk:64001/1415/1110129853/
[gcw@lab-07 gram]$ globus-job-status https://grid-data.rl.ac.uk:64001/1415/1110129853/
DONE
[gcw@lab-07 gram]$ globus-job-get-output https://grid-data.rl.ac.
uk:64001/1415/1110129853/
grid-data12.rl.ac.uk
[gcw@lab-07 gram]$ globus-job-clean https://grid-data.rl.ac.uk:64001/1415/1110129853/

WARNING: Cleaning a job means:
- Kill the job if it still running, and
- Remove the cached output on the remote resource

Are you sure you want to cleanup the job now (Y/N) ?
Y

Cleanup successful.
```

5. The examples so far have involved running a standard system program (hostname). The next stage is therefore to submit a simple custom program. The first program to use is "myjob.sh" which has been installed within your account. This program just prints the present working directory, the hostname again and also all the environmental variables currently set. Run the following commands to view and run this script on your local machine

   cd ~/gram
   cat myjob.sh
   ./myjob.sh

```
[gcw@lab-07 gram]$ cat myjob.sh
#!/bin/sh
echo $PWD
hostname -f
env

[gcw@lab-07 gram]$ ./myjob.sh
/home/gcw/gram
lab-07.nesc.ed.ac.uk
MANPATH=/opt/globus/man::/opt/edg/share/man:/opt/lcg/share/man:/opt/edg/man
HOSTNAME=lab-07.nesc.ed.ac.uk
GRID_PROXY_FILE=/tmp/x509up_u501
LCG_LOCATION_VAR=/opt/lcg/var
TERM=xterm
```

Now try running this job using globus-job-submit in the same way as previously shown. You will find that you got no output. Something is not working here. To diagnose the fault it is necessary to view the logfile that can be found in your account on the head node. Log on to the head node using gsissh (a version of ssh modified to use GSI authentication).

gsissh -p 2222 grid-data.rl.ac.uk

The log file will be in the top level of your account. Only jobs that fail keep their logfile, successful jobs result in the logfile being removed. If you have multiple log files in your account you can identify the relevant logfile by comparing your uid to the filenames. For example if you have a uid

https://grid-data.rl.ac.uk:64001/**2487**/1110130165/

then the relevant logfile is

gram_job_mgr_**2487**.log

When the logfile is viewed you will see that it contains a lot of information (replace the XXXX appropriately)

cat gram_job_mgr_XXXX.log

The relevant line may be found using

grep -a job-failure-code gram_job_mgr_XXXX.log

You should now have that your job failed with error code 5 (at the time of writing this tutorial a minor problem with the training accounts is presenting this error number being correctly reported, so don't worry if you don't get this exact output). If you look at a the list of error codes (see GRAM Error Codes) you will see that error code 5 means the executable was not found. This is because the script myjob.sh does not exist on the node running the job, it only exists on your local machine. To make the file exist on the node it is necessary to send the script with your job, a process called Staging.

```
[gcw@lab-07 gram]$ gsissh -p 2222 grid-data.rl.ac.uk
Last login: Sun Mar 6 13:45:32 2005 from lab-07.nesc.ed.ac.uk
ClusterVision Red Hat Enterprise 3.0 on Intel distribution v0.9

Use the following commands to adjust your environment:

'module avail' - show available modules
'module add ' - add a module

You should at least load a module for a compiler and a mpich
version of choice in your .tcshrc or .bashrc file.

----------------------------------------------------------------
```

```
[ngs0249@grid-data ngs0249]$ cat gram_job_mgr_2487.log
3/6 17:29:25 JM: Security context imported
3/6 17:29:25 JM: Adding new callback contact (url=https://lab-07.nesc.ed.ac.uk:20001/,
mask=1048575)
3/6 17:29:25 JM: Added successfully
....
[ngs0249@grid-data ngs0249]$ grep -a job-failure-code !$
grep -a job-failure-code gram_job_mgr_2487.log
job-failure-code: 5
```

Before continuing exit the remote machine

exit

Try running the same job but with the command (note the extra -s):

globus-job-submit grid-data.rl.ac.uk/jobmanager-pbs -s ./myjob.sh

This time you should be able to retrieve the expected output.

6. The previous programs have only used standard output for displaying their results. Many programs will output to files as well as to standard output and hence the question is what happens to these files. Inspect the file myjob2.sh in the same directory of your account. You will see that the environmental variables are now saved in a file called myenv.txt. ***Run this job and get it's output*** (and clean up).

   You will notice you still got the present working directory and hostname on the standard output. The file myenv.txt is actually written to your home directory on the head node. Rather than view this file by logging on to the head node, copy the file to your current directory instead:

   gsiscp -P 2222 grid-data.rl.ac.uk:myenv.txt .

   You can now view the file myenv.txt.
7. Returning to the topic of staging, if you are running the same job multiple times it is better if the program is stored somewhere accessible by a node running this job. Your account on the head node is the ideal place for this. In your account is a file called "myhostname.c" which is a simple piece of C code that prints out the hostname. Compile this code and initially submit this program as a job using staging. To compile this code use the command:

   gcc myhostname.c -o myhostname

   Now upload myhostname.c to the head node using gsiscp and compile (and test) it there. You will now be able to run the job without using staging.Since the program is sitting in the top level of your account on the head node you will not need to provide a path to the executable.

```
[gcw@lab-07 gram]$ gsiscp -P 2222 myhostname.c grid-data.rl.ac.uk:
[gcw@lab-07 gram]$ gsissh -p 2222 grid-data.rl.ac.uk
Last login: Sun Mar 6 18:18:11 2005 from lab-07.nesc.ed.ac.uk
ClusterVision Red Hat Enterprise 3.0 on Intel distribution v0.9
...
[ngs0249@grid-data ngs0249]$ gcc myhostname.c -o myhostname
[ngs0249@grid-data ngs0249]$ ./myhostname
host is grid-data.rl.ac.uk
[ngs0249@grid-data ngs0249]$ exit
logout
Connection to grid-data.rl.ac.uk closed.
```

8. For the final stage of this tutorial the issue of job dependencies is looked at. In your account is a fortan77 file called sdot_example.f. This program computes the dot product of two vectors, taking the first five entries of the first vector and alternating entries from the second vector. For those tutees less interested in Mathematics, the answer is 10. This code relies on a set of libraries that are available on

all of the core NGS sites (at least) in the module intel-math. ***Log on to the head node*** and load the intel-math module using the command

module load intel-math

The command to compile the code is

f77 -w sdot_example.f -lmkl_ia32 -lguide -lpthread -o sdot_example

```
[ngs0249@grid-data ngs0249]$ module load intel-math
[ngs0249@grid-data ngs0249]$ f77 -w sdot_example.f -lmkl_ia32 -lguide -lpthread -o sdot_example
[ngs0249@grid-data ngs0249]$ ./sdot_example
SDOT = 10.
[ngs0249@grid-data ngs0249]$ exit
```

Once you have compiled the code test the program by running it and then log off the head node. Next try submitting the job in the usual way. You should find that the job fails to produce the expected output. So far the command globus-job-get-output has been used to retrieve the standard output, but it can also be used to retrieve any standard error messages that have been produced in attempting to run your job. Use the below command with the uid of the job that has just failed

globus-job-get-output -err <uid>

from which you should see that the your program could not find a library it depends on to run. This is because submitted jobs need to be told to load the appropriate modules in the same way you loaded the modules when logged into the head node. This time submit your job with the command (all on one line)

globus-job-submit grid-data.rl.ac.uk/jobmanager-pbs -x '&(jobtype=single) (environment=(NGSMODULES intel-math))' sdot_example

and the job should now run successfully.

9. If you have time left in the tutorial try creating some jobs of your own. You could also explore the other modules available on the NGS core nodes. When logged on to the head node use

module avail

to get a list of the available modules. Alternatively you could explore the other options supported by the globus commands. All of the globus commands provide a detailed description of their usage by running the command with -help as the only command line parameter.