The GridFactory web service

Frederik Orellana
Niels Bohr Institute
Copenhagen University
May 2008

Contents

1 The GridFactory computing system

GridFactory [GRIDFACTORY] is a system for aggregating computing resources on local and wide area networks in order to carry out heavy-duty batch calculations. The system has a hierarchical pull-architecture, with servers pulling compute jobs from other servers and at the end of the chain, so-called worker nodes pulling jobs from one or several servers. The services running on the servers are implemented through Apache modules. The standard modules mod_dav and mod_ssl plus a new module, mod_gacl are used to implement a file server with flexible access control based on X.509 identities and virtual organisations[1],. Compute jobs are described by records in a database – currently MySQL. The worker nodes interact with this database through a RESTful [REST] web service, also implemented as a new Apache module – mod_gridfactory. The web services making up a GridFactory server are the subject of the present paper.

1.1 Job description

A compute job is described by a record in a database table. This record contains information on where to download input files and upload output files, the current status of the job, the requirements of the jobs etc. All fields of a job record are readable through the mod_gridfactory web service.

1.2 Job submission

Submitting a job consists in creating a directory in the job spool directory on the server served via HTTPS, and subsequently creating a file called “job” in this directory. The file called “job” is a shell script that contains directives in-lined as comments. A spool daemon on the server scans the spool directory with regular intervals and when a new job is found, parses these directives in the “job” file and creates a corresponding record in the database.

1.3 Job execution

Jobs are picked up by worker nodes that each run a daemon that regularly scans the job database on the GridFactory server. When a worker node daemon finds a new job, it downloads input files from the server via HTTPS, boots up a virtual machine if needed or requested (by the job or by its own configuration), executes the job and uploads the output files via HTTPS. During such a cause of events, the worker node daemon regularly updates the database with information on the job status. This, it does via the mod_gridfactory web service.

1.4 Job query and manipulation

Once a job has been submitted, the only field of a job record that is modifiable by the submitter or the worker node is the status field. By modifying this field, the submitter can prompt the worker node daemon to kill, pause or resume the job or upload the current stdout and stderr of the job to the server – from where it can then be downloaded. Modifying this field also allows a worker node to request the job and report on its status.

[1] mod_gacl is described in a separate publication [MOD_GACL].

2 Interface specification

2.1 Job directives

The file “job” in the job directory can either be:

- a single directive pointing to the executable job script (containing the in-lined job directives)

```
#GRIDFACTORY -x [executable script]
```

- the executable job script itself – containing the in-lined directives

```
#GRIDFACTORY -n [job name]
#GRIDFACTORY -i [input file 1] [input file 2] ...
#GRIDFACTORY -e [executable 1] [executable 2] ...
#GRIDFACTORY -t [running time on a 2.8 GHz Intel Pentium
                processor]
#GRIDFACTORY -m [required memory in megabytes]
#GRIDFACTORY -z (request virtualisation)
#GRIDFACTORY -o [output files]
#GRIDFACTORY -r [runtime environment 1] [runtime environment 2] ..
#GRIDFACTORY -y [operating system]
#GRIDFACTORY -s [distinguished name identifying the submitter]
#GRIDFACTORY -v [allowed virtual organization 1]
               [allowed virtual organization 2]
#GRIDFACTORY -u [unique identifier of the form
                https://my.server/gridfactory/jobs/[identifier]]
```

List-arguments, like the list of input files, can be given either like a space-separated list of strings (not containing spaces) or by repeated use of the directive.

2.2 Database schema

For reference, below is the full schema of the “jobDefinition” database table.

Field	Type
identifier	varchar(255)
name	varchar(255)
csStatus	varchar(255)
userInfo	varchar(255)
inputFileURLs	text
outFileMapping	text
providerInfo	varchar(255)
stdoutDest	varchar(255)
stderrDest	varchar(255)
created	datetime
lastModified	datetime
outTmp	varchar(255)
errTmp	varchar(255)
jobID	varchar(255)
metaData	text
host	varchar(255)
gridTime	int(16)
memory	int(16)
executable	varchar(255)
executables	text
opSys	varchar(255)
runtimeEnvironments	text
allowedVOs	varchar(255)
virtualize	tinyint(4)

Table 1: Database chema of the job definition table.

2.3 Mime types

Mimetype	Description
gridfactory/jobrecords+text	A list of job records in the form of an SQL response (with newline as record separator and tab as field separator)
gridfactory/jobrecord+text	A job newline-separated job record with lines of the form [key]:[value]

Table 2: Mime types for request bodies and response documents.

A job record is represented by a tab-separated line of values corresponding to the fields of table 1 plus a value corresponding to an extra field, 'dbUrl', added by mod_gridfactory. This last field, 'dbUrl', is the URL representing a single job as seen from the worker node, i.e. the URL on the server where information on a specific job can be obtained. Such a job may have been submitted directly to the server or the server may have pulled it from another server. Thus 'dbUrl' is not to be confused with 'identifier', which is the URL to which the job was originally submitted; even in the first case, depending on the set-up, they may or may not agree.

2.4 Communication and security

In the GridFactory system, the network communication is always over HTTPS, only between submitter and server or worker node and server and always initiated by the submitter or the worker node – never by the server (pull paradigm).

The HTTPS communication is two-way authenticated, so all peers must posses a public X.509 certificate and a secret RSA key.

The GridFactory security set-up is described in more detail in [GRIDFACTORY].

2.5 Submitting a job and uploading input files

The string [identifier] is chosen by the submitter. The submission utility that comes with GridFactory generates a time-based UUID [UUID], but in principle, any string can be used[2].

URL	https://my.server/db/jobs/[identifier]
Method	MKCOL
Returns	201 Created
	401 Access Denied
	403 Forbidden
	405 Method Not Allowed

Table 3: Creating a job directory.

Example:

```
MKCOL http://lx08/db/jobs/3b16dcdd-2d5f-11dd-80f2-c3b981785945
- - - - - - - - - - - - - - - - - - - - - - - -
```

[2]Uniqueness of the full URL is guaranteed by the fact that two different directories cannot have the same path.

201 Created

URL	https://my.server/db/jobs/[identifier]/job
Method	PUT
Returns	201 Created
	200 OK
	401 Access Denied
	403 Forbidden
	405 Method Not Allowed

Table 4: Uploading a job file.

Example:

PUT http://lx08/db/jobs/3b16dcdd-2d5f-11dd-80f2-c3b981785945/job

- -

201 Created

2.6 Getting job records

URL	https://my.server/db/jobs/	
Method	GET	
Query string	csStatus=	Filter by status
	userInfo=	Filter by DN of submitter
	providerInfo=	Filter by DN of provider
	start=	The number of the first job to return - 0 is the number of the first job
	end=	The number of the last job to return
Returns	200 OK & TEXT (gridfactory/jobrecords+text)	
	401 Unauthorized	
	404 Not Found	

Table 5: Getting multiple job records.

Example:

GET https://pullsys.nbi.dk/grid/db/jobs/?status=ready&start=0&end=1

```
- - - - - - - - - - - - - - - - - - - - - - - -
200 OK


identifier  name  csStatus  userInfo  inputFileURLs  outFileMapping
        providerInfo  stdoutDest  stderrDest  createdlastModified  outTmp  errTmp
        jobID  metaData  host  gridTime  memory  executable  executables
        opSys  runtimeEnvironments allowedVOs  virtualize  dbUrl
https://orellana.nbi.dk/grid/jobs/3a86aacc-2d5f-11dd-80f2-c3b981785945  job  ready
        https://pullsys.nbi.dk/grid/jobs/3a86aacc-2d5f-11dd-80f2-c3b981785945/job
        https://pullsys.nbi.dk/grid/jobs/3a86aacc-2d5f-11dd-80f2-c3b981785945/stdout
        https://pullsys.nbi.dk/grid/jobs/3a86aacc-2d5f-11dd-80f2-c3b981785945/stderr  2008-05-
29 11:11:32  2008-05-29 11:11:32                                  -1  -1  job
                        0  https://lx08/db/jobs/3a86aacc-2d5f-11dd-80f2-c3b981785945
https://pullsys.nbi.dk/grid/jobs/3b16dcdd-2d5f-11dd-80f2-c3b981785945  job  ready
        https://pullsys.nbi.dk/grid/jobs/3b16dcdd-2d5f-11dd-80f2-c3b981785945/job
        https://pullsys.nbi.dk/grid/jobs/3b16dcdd-2d5f-11dd-80f2-c3b981785945/stdout
        https://pullsys.nbi.dk/grid/jobs/3b16dcdd-2d5f-11dd-80f2-c3b981785945/stderr  2008-05-
29 11:11:32  2008-05-29 11:11:32                                  -1  -1  job
                        0  https://lx08/db/jobs/3a86aacc-2d5f-11dd-80f2-c3b981785945
```

2.7 Getting a specific job record

URL	https://my.server/db/jobs/[identifier]
Method	GET
Returns	200 OK & TEXT (gridfactory/jobrecord+text)
	401 Unauthorized
	404 Not Found

Table 6: Getting a single job record.

Example:

```
GET http://lx08/db/jobs/3b16dcdd-2d5f-11dd-80f2-c3b981785945
- - - - - - - - - - - - - - - - - - - - - - - -
200 OK


identifier: https://orellana.nbi.dk/grid/jobs/3b16dcdd-2d5f-11dd-80f2-c3b981785945
name: job
csStatus: ready
userInfo:
inputFileURLs: https://pullsys.nbi.dk/grid/jobs/3b16dcdd-2d5f-11dd-80f2-c3b981785945/job
outFileMapping:
providerInfo:
stdoutDest: https://pullsys.nbi.dk/grid/jobs/3b16dcdd-2d5f-11dd-80f2-c3b981785945/stdout
```

stderrDest: https://pullsys.nbi.dk/grid/jobs/3b16dcdd-2d5f-11dd-80f2-c3b981785945/stderr
created: 2008-05-29 11:11:32
lastModified: 2008-05-29 11:11:32
outTmp:
errTmp:
jobID:
metaData:
host:
gridTime:-1
memory:-1
executable:job
executables:
opSys:
runtimeEnvironments:
allowedVOs:
virtualize:0

Notice that the identifier contains a host name different from the one of the host running the service. This indicates that the host running the service has pulled the job from somewhere else. The host name in the identifier is the name of the host to which job was originally submitted.

2.8 Modifying the status of a job

To modify an existing job, we'll PUT a text representation of a job to the URL of our job.

URL	https://my.server/db/jobs/[identifier]
Method	PUT
Request body	TEXT (gridfactory/jobrecord+text)
Returns	201 Created & Location
	401 Unauthorized
	404 Not Found
	415 Unsupported Media Type

Table 7: Modifying a single job record.

Since we don't want people to create new job records using PUT, only update existing job records, we'll return a '404 Not' Found error if the resource doesn't already exist.

Example:

PUT http://lx08/db/jobs/3b16dcdd-2d5f-11dd-80f2-c3b981785945

csStatus: ready

- -

201 Created

Notice all values of a job record can be changed, except for those corresponding to the keys 'created' and 'lastModified'. These are set by the server. The value of 'lastModified' is changed on each PUT, regardless of whether the job record is actually changed.

2.9 Downloading output files

URL	https://my.server/db/jobs/[identifier]/[file]
Method	GET
Returns	200 OK & BODY
	401 Unauthorized
	404 Not Found

Table 8: Getting a single job record.

Example:

GET http://lx08/db/jobs/3b16dcdd-2d5f-11dd-80f2-c3b981785945/output_file_1.data

- -

200 OK

File body

3 Bibliography

GRIDFACTORY, Frederik Orellana, "GridFactory a virtualised compute system for distributed clusters", *in preparation*

MOD_GACL, Frederik Orellana, “Authorization and virtual organisations with Apache, X.509 and GACL”, *in preparation*

REST, RESTful web services are described in e.g.
http://www.peej.co.uk/articles/restfully-delicious.html

UUID, International Telecommunication Union, “Information technology - Open Systems Interconnection - Procedures for the operation of OSI Registration Authorities: Generation and registration of Universally Unique Identifiers (UUIDs) and their use as ASN.1 Object Identifier components “, http://www.itu.int/ITU-T/studygroups/com17/oid.html