

Execution Management: Key Concepts

Execution Management: Key Concepts

Overview

The Globus Toolkit provides a service to submit, monitor, and cancel jobs on Grid computing resources called GRAM. In GRAM, a Job consists of a computation and, optionally, file transfer and management operations related to the computation. Some users, particularly interactive ones, benefit from accessing output data files as the job is running. Monitoring consists of querying and subscribing for status information such as job state changes.

Grid computing resources are typically managed by a local resource manager which implements allocation and prioritization policies while optimizing the execution of all submitted jobs for efficiency and performance according to site policy. GRAM is not a resource scheduler, but rather a protocol engine for communicating with a range of different local resource schedulers using a standard message format.

For more detailed information about the concepts behind the software implementation, see [Chapter 2, GT 5.0.2 GRAM5 Approach](#).

Table of Contents

| | |
|--|---|
| 1. Conceptual details | 1 |
| 1. Targeted job types | 1 |
| 2. Component architecture | 1 |
| 3. Security | 1 |
| 4. Job Management | 2 |
| 5. Data Management | 2 |
| 2. GRAM5 Approach | 3 |
| 1. Introduction | 3 |
| 2. Component architecture approach | 3 |
| 3. GRAM Client-Server Interactions | 4 |

List of Tables

| | |
|-------------------------------------|---|
| 2.1. GRAM5 Service Components | 3 |
|-------------------------------------|---|

Chapter 1. Conceptual details

A number of concepts underly the purpose and motivation for GRAM. These concepts are divided into broad categories below.

1. Targeted job types

GRAM is meant to address a range of jobs where arbitrary programs, reliable operation, stateful monitoring, credential management, and file staging are important. GRAM is not meant to serve as a "remote procedure call" (RPC) interface for applications not requiring many of these features, and furthermore its interface model and implementation may be too costly for such uses. The GRAM5 service protocols and implementation will always involve multiple round-trips to support these advanced features that are not required for simple RPC applications.

2. Component architecture

Rather than consisting of a monolithic solution, GRAM is based on a component architecture at both the protocol and software implementation levels. This component approach serves as an ideal which shapes the implementation as well as the abstract design and features.

| | |
|---------------------------------|--|
| Service model | For GRAM5, the globus-gatekeeper daemon and GSI library are used for secure communications and service dispatch. The globus-job-manager daemon implements the job management and file transfer functionality. |
| Local Resource Manager Adapters | GRAM provides a scripted plug-in architecture to enable extension with adapters to control a variety of local resource systems. |

3. Security

| | |
|--------------------------------------|--|
| Secure operation | GRAM5 uses SSL-based protocols to establish identity or provide other security tokens needed to authorize GRAM5 service requests. Once authorized, each instance of the job service runs as a local POSIX user. GRAM5 restricts job monitoring and management operations to those who are authorized by the local site policy. |
| Local system protection domains | To protect users from each other, the GRAM5 job manager and the jobs it starts are executed in separate local security contexts. Additionally, GRAM mechanisms used to interact with the local resource are design to minimize the privileges required and to minimize the risks of service malfunction or compromise. |
| Credential delegation and management | A client delegates some of its rights to the GRAM service in order to allow it to perform file transfer on behalf of the client and send state notifications to registered clients. Additionally, GRAM5 provides per-job credentials so that job instances may perform further authentication with other services. |
| Audit | GRAM uses a range of audit and logging techniques to record a history of job submissions and critical system operations. These records may be used to assist with accounting functions as well as to further mitigate risks from abuse or malfunction |

4. Job Management

| | |
|-------------------------|---|
| Reliable job submission | GRAM5 provides a two-phase commit protocol for reliable job submission. When used, GRAM5 will not stage files or submit a job to the LRM without acknowledgement from the client that it is able to contact the job management service. Similarly, at job termination time, GRAM5 jobs can also require that the client acknowledge the job completion before GRAM clears state associated with the job. |
| Job cancellation | GRAM provides a mechanism for a client to cancel its jobs at any point in the job life cycle. |

5. Data Management

| | |
|--------------|--|
| Data staging | GRAM5 provides for high-performance transfers of files between the compute resource and external data storage elements before and after the job execution. Data can be staged from HTTP, GASS, and GridFTP services. |
|--------------|--|

Chapter 2. GT 5.0.2 GRAM5 Approach

1. Introduction

The GRAM5 software implements a solution to the job-management problem described in [Key Concepts](#). This solution is specific to operating systems following the POSIX programming and security model.

2. Component architecture approach

GRAM5's job management services interact with local resource managers and other service components of GT 5.0.2 in order to support job execution with coordinated file staging.

2.1. GRAM5

The GRAM5 service architecture consists of several components which work together to authenticate users, manage jobs, interface with the LRM, and stage files. These components are described in the following table.

Table 2.1. GRAM5 Service Components

| | |
|--------------------------------------|--|
| Gatekeeper | The globus-gatekeeper service performs authentication and service startup for the GRAM5 job management service. One instance of this daemon is created for each job submission, but these instances are short-lived. |
| Job Manager | The globus-job-manager daemon processes job requests and coordinates file transfers. There is one long-lived instance of this per user per LRM and one short-lived instance per job. |
| Job Manager Script | The globus-job-manager-script.pl process interacts with the local resource manager via the LRM adapter and file servers via the globus-url-copy program. There are 1-5 instances of this per user per LRM. |
| Job Manager LRM Adapter | The LRM adapter to interact directly with the LRM. It is loaded into the Job Manager Script component when it starts. |
| Scheduler Event Generator | The globus-job-manager-event-generator process parses LRM-specific data relating to job startup, execution, and termination into an LRM-independent data format. There is one instance of this program per LRM. |
| Scheduler Event Generator LRM Module | The LRM module for the scheduler event generator processes LRM state to produce events that the Scheduler Event Generator writes to an event log. |
| GRAM Audit Database | The Job Manager can be configured to write audit records to a file. The globus-gram-audit program loads those records into an SQL database. |

2.2. External Components used by GRAM5

2.2.1. Local resource manager

GRAM5 uses a local resource manager (LRM) to schedule and run jobs on a compute element. GRAM5 supports several common LRM systems (Condor, PBS, LSF) and can also be configured to run jobs without an LRM.

2.3. Internal Components used by GRAM5

2.3.1. Scheduler Event Generator

The **globus-job-manager-event-generator** program provides the job monitoring capability for the GRAM5 service. Plugin modules provide an support for the LRMs supported by GRAM5.

2.3.2. Fork Starter

GRAM can be configured to use the Fork Starter program to start and monitor job processes in place of a LRM. This allows GRAM to use the SEG interface with fork jobs.

3. GRAM Client-Server Interactions

The GRAM client interacts with the **globus-gatekeeper** to start a GRAM Job Manager process or to "ping" a GRAM Job Manager service. The **globus-gatekeeper** program starts a Job Manager process when the client initiates a job request or a version info query. In either of these cases, the gatekeeper creates a new Job Manager process, passing a file descriptor that is the connection back to the client, a file descriptor containing the HTTP message body, and sets the `X509_USER_PROXY` environment variable to point to the path of the delegated proxy. The **globus-gatekeeper** program exits after this process is started. When the client does a "ping" request, the **globus-gatekeeper** program reads the service configuration file for the Job Manager service and verifies that the executable exists, but does not start the Job Manager.

The GRAM client interacts with the **globus-job-manager** program as it receives the response to a job request of version query, using the security context established during its communication with the **globus-gatekeeper** program. It can then send messages to the job contact returned from the job request to cancel a job, check its status, signal it, register or unregister a callback contact, or renew a job proxy. If it registers a callback contact either during job submission or after the job has submitted, the **globus-job-manager** will send https messages to the client with status updates when the GRAM job state changes.