August 29, 2005

JobMon: An Interactive GRID Job Monitor

C. Steenberg
California Institute Technology

E. Lipeles, S.C. Hsu, F. Würthwien
University of California San Diego

Abstract

This document covers the conceptual design of a system for users to monitor the status of GRID jobs in real time. The JobMon system allows users to view process status and log files interactively as the job runs. This facilitates debugging and reduces unnecessary resource consumption due to user error.

1 Introduction

When a user submits a GRID job a large number of things can go wrong even after the job starts. For example, resources such as data handling systems and databases can be broken or inaccessible or the user can make simple errors in the assembly of the job. In order to quickly and efficiently detect and diagnose problems, users need to be able to access the running job before completion. Simple task such as being able to read log files in real time, or see when the job is consuming CPU can be of great value. The JobMon[3] system implements a secure and authenticated method for users to access running GRID jobs. It is an generalization of the tools originally constructed for the CDF experiment [1]. The general design of the JobMon system is driven by two factors major factors: locating and establishing communication with the job, and authenticating this connection. Authentication and security are particularly important because the user will be able to execute commands and access data on the job's computing resources.

1.1 Components of the System

There are three separate places where code is run in an individual transaction. There is the persistent Clarens web-service[2], generally located at the execution site. The job runs a *jobmond* process which runs the communication and executes the commands on the worker node. The *jobmond* persists for as long as the job is executing. Finally there is the client code, which a users executes when they want to interact with the remote job, this persists only until the transaction has completed. This configuration is diagrammed in Figure 1.

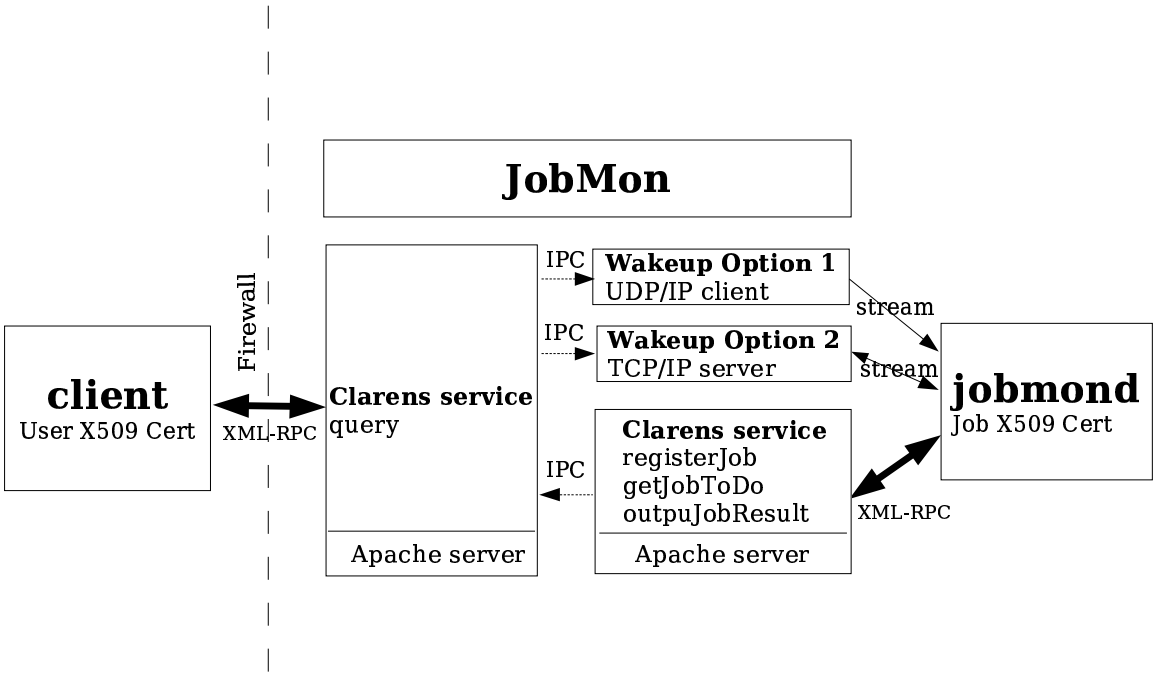

Figure 1: Diagram of the System Components and Authentication.

The central server has the primary responsibility of authenticating the user and implementing the access control that limits users to interacting with jobs for which they have permission.

While the *jobmond* persists for as long as the job is executing, it cannot be used as a TCP or other kind network server since it is assumed that many sites will disallow user jobs from listening on worker node ports. In order to accommodate this a wake-up/call-back mechanism has been implemented. When the central server needs to send a command to a worker node the server calls a wake up module, which can have multiple different implementations depending on the site. The currently existing implementations are UDP broadcast and a central TCP server with open connections to all jobs. Each of these as different limitations. Once awake, *jobmond* communicates using secure and authenticated Clarens calls.

1.2 The Sequence of Events

Job start up:

1. At the time of job submission the user must include the *jobmond* code along with the job PKI X.509 cert/key pair, unless this is to be obtained later, e.g. using kerberos.

2. When the job begins, *jobmond* locates the server using the location service.

3. It then registers itself on the *JobMon* server and retrieves the configuration. This allows the configuration to be both site dependent and dynamic.

4. If the retrieved wake-up method is:

   - UDP broadcast, *jobmond* will start a UDP server to wait for coming connection.
   - TCP broadcast, *jobmond* will make a TCP/IP socket client connection to the JobMonTCPServer which sits in the Clarens server node and whose location is specified in the configuration.

   There are no encryptions or authentications during wake-up processes.

Handling a user request:

1. When a user would like to interact to the job, he or she makes a JobMon query service as client shown in the Figure with user PKI X.509 certificate. The subjects of X509 cert for user and job need not to be the same but should contains enough information for access control. The Clarens system handles the actual X509 authentication process.

2. After authentication, the web-service can then send a non-encrypted wake-up message through UDP/IP or TCP/IP wake-up mechanism to all registered *jobmond*.

3. On receiving the wake-up message, *jobmond* will will respond if it has been specified in the wake-up call by making the getJobToDo call to the Clarens server.

4. After executing the command *jobmond* returns the results with the outputJobResult call to the server.

Inside the Clarens server the user-side communicates withe the job-side using interprocess communication (FIFOs).

1.3 Authentication

The Clarens infrastructure is responsible for authenticating the user and the job using the X509 certificates they provide. Once that has been completed the JobMon code allows for matching between the user and job GRID subject using regular expression based criteria. A system administrator configuring their JobMon service can the the match criteria. This is generally simply matching the user name and authority extracted the user subject to the user name and authority extracted from the job subject.

For example with kx509 generated certificates from Fermilab, we require that the certificate name and the user id match. This is specified with a configuration entry of the form:

- User:

  ```
  /DC=gov/DC=fnal/O=Fermilab/OU=People/CN=<name> /UID=<user>
  ```

- Job:

  ```
  /DC=gov/DC=fnal/O=Fermilab/OU=Robots/CN=cdf
                  /CN=<name>/0.9.2342.19200300.100.1.1=<user>
  ```

Procedurally the strings corresponding to the flags "⟨name⟩" and "⟨user⟩" are required to be the same in both the user certificate (first) and the job certificate (second).

1.4 Performance

In order to remove completed jobs from the server registry, jobs are required to periodically reregister. The load these reregistrations place on the Clarens server limits the overall number of jobs a single server can support. Tests have shown that dual 3.4 GHz Xeon can support 4 Hz of registration calls and maintain a 1 second latency. This corresponds to supporting a 3000 job farm with a 12 minute reregistration time. For a 10 Hz rate the latency is increased to ≈ 10 seconds. The since the reregistration is only to identify jobs that are no longer running the reregistration time can be set fairly long. The only cost associated with this is that jobs that do not terminate cleanly will be left in the list of known jobs for longer.

References

[1] *Central Analysis Facility - Run II computing in CDF,* `http://cdfcaf.fnal.gov/`

[2] *Clarens Grid-Enabled Web Services Framework* , `http://clarens.sourceforge.net/`

[3] *JobMon: an Interactive GRID Job Monitor,* `http://physics.ucsd.edu/~schsu/project/JobMon/`