Custom accounting for the Simple Linux Utility for Resource Management

Yuen Wan (Leanne), Lee
Lawrence Berkeley National Lab - NERSC
Berkeley, CA
TRUST REU
leannelee812@berkeley.edu

***Abstract*—An efficient way to manage the workload on High Performance Computing Clusters is to use Resource Managers. One such of the resource manager and job scheduling system for Linux clusters is the Simple Linux Utility for Resource Management (SLURM). At National Energy Research Scientific Computing Center (NERSC), users regularly submit and run hundreds of jobs. Thus, it is necessary to have an accounting tool to track their job history. This article describes a new tool to query and display the accounting data in SLURM.**

Keywords—NERSC; High Performance Computing; SLURM; Carver; accounting

I. Introduction

High Performance Computing plays a significant role in scientific research. It ranges from climate modeling, and investing new materials, to data analysis. At the Lawrence Berkeley National Laboratory, the National Energy Research Scientific Computing Center (NERSC) is a national user computing facility. NERSC provides computing and storage for over five thousands users. NERSC runs several large high performance computing systems: Edison, Hopper and Carver. These large systems house many CPUs and networking in massive racks supported by an advanced cooling system. These fundamental systems handle over hundreds of thousands of users jobs daily.

The focus of this project is the accounting tool inside of the workload management. Specifically, we will consider Carver: a liquid-cooled IBM iDataPlex system with over 1200 compute nodes [4]. With over 9000 processor cores, the peak performance is 106.5 teraflops per second. The performance in teraflops is fast, because one teraflops is equivalent to one trillion floating-point operations per second. A great example of teraflops is perform a calculation every second for over 30,000 years, while it takes one second in teraflops[7]. The workload on Carver is a mixture of serial and parallel jobs, as well as variety of job types: short jobs, long running ones, as well as a high-throughput workload.

NERSC users submit numerous jobs into the systems, and they need to recall the job information in a quick and easy way. NERSC already uses two different workload managers on its different systems: Univa GridEngine and Torque/Moab. However, Torque/Moab do not have the ability to track accounting information and GridEngine can only track information within a certain amount of time. As a test, we will use the, Simple Linux Utility for Resource Manager (SLURM), which is installed on a testbeds system within Carver. Similar to genepool, qqacct will be install into the SLURM batch system, which will create a consistent user interface to access batch job history.

II. Batch System

Batch system typically consists of two components: a resource manager and a job scheduler.

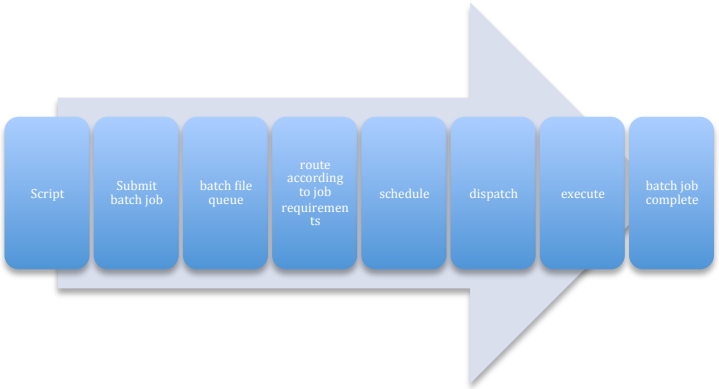

Figure 1: The figure shows how batch system works.

First, the user writes a script in any programming language. Second, the user submits batch scripts into a high performance computing system. Third, the batch system schedules a queue to determine if the batch files are short running job, long running job or regular running job. Fourth, the batch system would route each batch file according to their job requirements. While some jobs request a certain amount of compute nodes, others request a walltime. Fifth, the batch system schedules priority to the jobs. Sixth, the system dispatch jobs according to their priority. Seventh, the jobs are being executed. Finally, the batch jobs are completed after execution[2].

III. Simple Linux Utility for Resource Manager (SLURM)

SLURM is an open source code for Linux clusters, and it has about 500,00 lines of C code. SLURM is one of the most widely used resource managers on HPC systems, and is used on 5 of the 15 fastest supercomputers in the world [5]. SLURM is a user utility for high performance computing.

A. *Why choose SLURM*

First, SLURM is an open source, which means it does not require a license fee. However, SLURM is not absolutely free. There is a small fee for technical support service. Second, SLURM is easy to import because it provides the source code is available online. Instead of spending months to write the source code from the beginning, having the source code and installation guide available can enhance the efficiency to import SLURM into a big system. Third, SLURM is suitable for high computing systems because it can handle thousands jobs at one time. Fourth, it is easy to customize SLURM to suit one's preference. Since SLURM source code is an open source, one can add features into the code without changing the entire source code. SLURM is a low-cost and useful tool to very large systems.

B. *The architecture of SLURM*

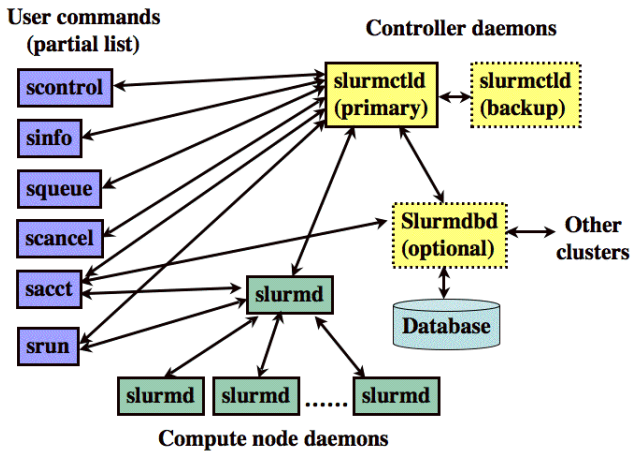

Figure 2: The figure shows the basic architecture of the SLURM workload manger [6].

Every batch system has a central controller. In SLURM, slurmctld is the controller that monitors state of resources and manage job queues. Then, slurmd is the compute node daemon that manages tasks and executes them. Slurmdbd is also a backup system and collects accounting information. Lastly, there are some user commands to display information about their queues and job information.

SLURM has plugins to support the basic structure. The scheduler plugin determines when SLURM schedules jobs and can change the job priority. The priority plugin assigns priorities to jobs. The authentication of communicates gives authentication for different parts of SLURM. There are more plugins in the SLURM workload manager. The most important one that related to this project is the accounting storage plugin, which stores information of the jobs in slurm database daemon.

IV. Contribution

At NERSC, there is a custom accounting query tool called qqacct that users can use to query the accounting database for current and historical data about their jobs. Since SLURM is a new batch system, it is useful to import qqacct into SLURM and provide a standard interface for users. The accounting tool will allow users to control their output. Instead of writing qqacct in C++ language, the qqacct for SLURM is written in python using the MySQLdb module to make queries to the accounting database. The main reason we choose python for qqacct is because python language is easy to extend. The qqacct is an interface to MySQL database server that provides Python database application programming interface. SLURM is configured to store its accounting information in a MySQL database.

A. *Connect to slurm accounting database*

```
#connect to the slurm accounting database
db = MySQLdb.connect(user='slurm',passwd='slurmT3st',host='localhost',
    unix_socket='/usr/syscom/opt/slurm/mysql/etc/mysql.sock',db='slurm_acct_db
    ');
```

In the beginning, it is important to follow mysql syntax to connect to the slurm accounting database on carver. In the figure above, there is an unique unix domain socket and it is used to communicate over to the slurm accounting database.

B. *Create an appropriate SQL query*

By using MySQL database queries, the users can filter and sort data within a database. A basic query is the SELECT statement. The SELECT statement can choose a particular column of data that the user wants to view.

C. *Get information from the database*

SLURM has built in some user commands, which the users can get information from the database. For instance, sinfo reports job status and squeue indicates which queue the job is on. In MySQL syntax, "show tables" and "describe" can get accounting data information from the database.

D. *Take arugments*

This program is to create a common interface to qqacct that runs across all the systems at NERSC. Thus, the arguments and syntax in slurm accounting database have to be similar to the arguments in GridEngine. The accounting program should be able to sort any details from the data and display them to the users. To take arguments from the users, I use argparse in python to add each of the commands. In this case, the program I created would take arguments of username, jobname, partition, time start and time end. For example, a user can pick a specific user and his or her jobname, then the system will sort out the data and display the

correct information. The user does not have to put all the details to extract data from the slurm accounting database; however, the username is set by default and must be included everytime.

E. Time format

In Unix time, the unix epoch starts on January 1st, 1970. If the user asks to display the end time or the start time, it will output the time in seconds between January 1st, 1970 to the actual time. The big number in seconds is not helpful because the users have to perform mathematical calculation in order to get the correct local time.

```
a = time.strftime('%Y-%m-%d %H:%M:%S', time.localtime(endTime))
```

By using the time module in the figure shows above, the code converts from unix epoch time in seconds to the local time. Therefore, the users are able to view the output in correct local time.

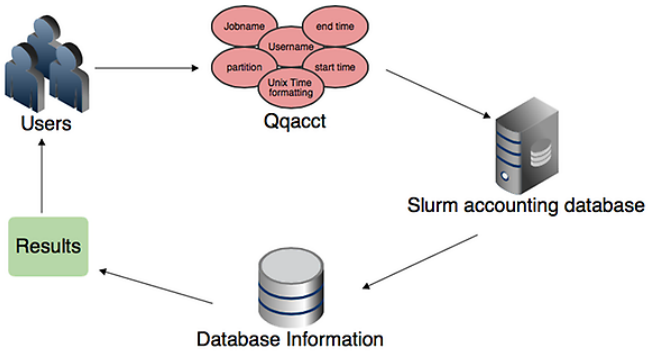

Figure 3

The figure above demonstrates how the qqacct tool works. The users make a query, which is a request for job information in a database. In qqacct, there is a query list where the users can choose which part of information they are interested to see. For example, the users can choose a specific user with a certain start time. After submitting the query, the system accesses into the slurm accounting database. Next, the system gets and sorts data from the database. Lastly, the sorted information returns back to the users. The qqacct is a useful tool to help the users to find the records of the specific jobs. The users can pick which data are outputted. Thus, the users can analyze their data without searching through thousands of jobs.

V. RESULTS

```
-bash-3.2$ python qqacct.py -u 57399 -j xhpl.sh -e 2014-07-20
Jobs associated with this user 57399
Jobs associated with this user with jobname xhpl.sh
1273 | xhpl.sh | reg_big | 02-18-2014  14:29:44
1313 | xhpl.sh | reg_big | 02-21-2014  15:45:59
1314 | xhpl.sh | reg_big | 02-21-2014  15:53:42
1315 | xhpl.sh | reg_big | 02-21-2014  16:05:36
3 | xhpl.sh | reg_big | 03-21-2014  13:58:24
5 | xhpl.sh | reg_big | 03-21-2014  14:03:36
6 | xhpl.sh | reg_small | 03-21-2014  14:14:05
7 | xhpl.sh | reg_med | 03-21-2014  14:14:34
8 | xhpl.sh | reg_med | 03-21-2014  14:15:05
11 | xhpl.sh | reg_med | 03-21-2014  17:42:20
118 | xhpl.sh | reg_small | 03-25-2014  14:34:04
21044 | xhpl.sh | reg_med | 05-06-2014  09:36:48
21046 | xhpl.sh | cdebug | 05-28-2014  14:58:22
21082 | xhpl.sh | cdebug | 07-01-2014  15:43:06
21083 | xhpl.sh | cdebug | 07-01-2014  15:48:21
21084 | xhpl.sh | cdebug | 07-01-2014  16:34:28
21085 | xhpl.sh | regular | 07-01-2014  16:38:52
21086 | xhpl.sh | regular | 07-01-2014  16:49:01
21087 | xhpl.sh | reg_big | 07-01-2014  16:59:29
```

Figure 4

In Figure 4, the user can select a single user with a particular job name and end date. With this query, the system will only run information for the specific user according to their user ID, the job name, and their jobs before the given end date. The output format is organized in columns and each section is separated by lines. This project will continue to be develop by importing more arguments, such as partition, modify time, time submit, priority, node allocation and more.

VI. FUTURE WORK

Qqacct is an effective accounting tool. The project will be continued to develop and to take more arguments and accept more switches. With qqacct, users would not have to learn a new interface to display their job information. It will become a common utility that works on other platform. With a common way to query batch accounting information across platforms it will be easy to present this information via a web interface, where users can obtain their data at NERSC website. Also, the qqacct is possible to import into other high performance computing systems, such as Edison and Hopper. Certainly, the qqacct will benefit NERSC users to acquire data in a fast and simple method.

ACKNOWLEDGMENT

This research is sponsored by TRUST (Team for Research in Ubiquitous Secure Technology) and NSF (National Science Foundation). The author would like to give a great appreciation and thank you to her mentors, James Botts and Jay Srinivasan for guiding and providing support along the research.

REFERENCES

[1] A. Dustman, "MySQL-Python," (Sourceforge), [online] 2005, http://mysql-python.sourceforge.net (Accessed: 05 July, 2014)

[2] D. Thakur. "What is Batch Processing Operating System?" *Computer Notes,* 2010, http://ecomputernotes.com/fundamental/disk-operating-system/batch-processing-operating-system (Accessed: 28 July, 2014)

[3] FAQ: Batch, *The Center For High Performance Computing,* [online] 08 November, 2013 http://www.chpc.utah.edu/docs.manuals/faq/batch.html (Accessed: 20 July, 2014)

[4] National Energy Research Scientific Computing Center, *NERSC,* [online] 1974, http://www.nersc.gov (Accessed: 10 June, 2014)

[5] Slurm Used on the Fastest of the TOP500 Supercomputers, *Slurm Workload Manager,* [online] 21 Feburary 2012, http://slurm.net (Accessd: 20 July, 2014)

[6] Slurm Workload Manager, *SchedMD,* [online] 3 Apil 2013, http://slurm.schedmd.com/slurm.html (Accessd: 05 July, 2014)

[7] Understanding measures of supercomputer performance and torage system capacity, *Inidiana Univeristy Knowledge Base,* [online] 2014 https://kb.iu.edu/d/apeq (Accessed 30 July, 2014)