THE HEP SOFTWARE FOUNDATION (HSF)

HSF-TN-2016-02
10.5281/zenodo.1469629
February 4, 2016

Machine/Job Features

M. Alef[1], T. Cass[2], J.J. Keijser[3], A. McNab[4], S. Roiser[2], U. Schwickerath[2], I. Sfiligoi[5]

[1] *Karlsruhe Institute of Technology*
[2] *CERN*
[3] *NIKHEF*
[4] *University of Manchester*
[5] *Fermi National Accelerator Laboratory*

Abstract

Within the HEPiX virtualization group and the WLCG MJF Task Force, a mechanism has been developed which provides access to detailed information about the current host and the current job to the job itself. This allows user payloads to access meta information, independent of the current batch system or virtual machine model. This information includes the performance of the node and the remaining run time for the current job.

1 Introduction

Within the HEPiX virtualization group [1] and the WLCG MJF Task Force [2], a mechanism has been developed which provides access to detailed information about the current host and the current job to the job itself. This allows user payloads to access meta information, independent of the current batch system or virtual machine model.

The proposed schema is made to be extensible so that additional information can be added. The purpose of this note is to define the specifications and use case of this schema. It should be seen as the source of information for the actual implementation of the scripts required by the sites to provide it.

2 Aims

- The proposed schema must be unique and leave no room for interpretation of the values provided. For this reason, basic information is used which is well defined across sites.
- Host and job information can be both static (like the HS06 [3] rating of the hardware) and dynamic (eg shutdown time may be set at any time by the site.)
- Job specific files will be readable and possibly owned by the user and residing on a /tmp like area
- The implementation, that is the creation of the files and their contents, can be highly site specific. A sample implementation can be done per batch system in use, but it is understood that sites are allowed to change the implementation, provided that the created numbers match the definitions given in this note.

3 Use cases

The use cases considered in developing the protocols included:

1. The job needs to calculate the remaining time it is allowed to run.
2. The job needs to know how long it was already running.
3. The job wants to know the performance of the processors allocated to it in order to calculate the remaining time it will need to complete (for CPU intensive jobs).
4. A host needs to be drained, and the payload needs to be informed of the planned shutdown time.
5. A multiprocessor user job on a non-exclusive node needs to know how many threads or processes it is allowed to start. This especially useful in a late-binding scenario where the pilot reserved the processors and the user payload needs to discover this.

6. A user job wants to know how many processors are allocated to the current job.

7. A user job wants to know the maximum amount of scratch disk it is allowed to use.

8. A user job wants to set up memory limits to protect itself from being killed by the batch system automatically.

4 Definitions

On VM-based systems, references to "jobs" are to be interpreted as "virtual machines" and "machines" as "hypervisors".

When jobs are running within virtual machines, the entity that provides the system level configuration or contextualization of the VM acts as the resource provider referred to in the rest of this note.

5 Environment variables

For each job, two environment variables may be set, with the names $MACHINEFEATURES and $JOBFEATURES.

These environment variables are the base interface for the user payload. Their values must be provided for the job by the resource provider.

In the case of virtual machines on IaaS cloud platforms, the virtual machine may discover the values to set for the environment variables from "machinefeatures" and "jobfeatures" metadata keys provided by resource provider via the cloud infrastructure. These metadata keys should only be accessed once in the lifetime of each virtual machine. Alternatively, the values to set may be supplied as part of the contextualization of the virtual machines.

6 Directories

The environment variables point to directories created by the resource provider. Inside, the file name is the key, the contents are the values, so that files can be referred to with expressions like $MACHINEFEATURES/shutdowntime . The directory name should not include the trailing slash. These directories are either local directories in the filesystem or sections of the URL space on an HTTP(S) server. The user positively determines whether the files are to be opened locally or over HTTP(S) by checking for a leading slash or the prefix http:// or https:// respectively. Typically this can achieved using library functions which can transparently handle local files and remote URLs when opening files.

Unlike metadata keys, the key/value files may be accessed multiple times to check for changes in value or in the absence of caching by the user. An HTTP(S) server may provide HTTP cache control and expiration information which the user may use to reduce the number of queries. All files in the directories must be readable by both the user and

the resource provider services, and have file names which only consist of lowercase letters, numbers, and underscores.

7 $MACHINEFEATURES

Host-specific key/value pairs which are all:

- Found in the directory pointed to by $MACHINEFEATURES
- Readable by the user who is executing the original job. In the case of pilots this would be the pilot user at the site.
- Required unless the resource provider cannot determine their value.
- Static unless otherwise stated.

total_cpu Number of processors which may be allocated to jobs. Typically the number of processors seen by the operating system on one worker node (that is the number of "processor :" lines in /proc/cpuinfo on Linux), but potentially set to more or less than this for performance reasons. (Use case 3.)

hs06 Total HS06 rating of the full machine in its current setup. HS06 is measured following the HEPiX recommendations [3], with HS06 benchmarks run in parallel, one for each processor which may be allocated to jobs. (Use case 3.)

shutdowntime Shutdown time for the machine as a UNIX time stamp in seconds. The value is dynamic and optional. If the file is missing, no shutdown is foreseen. (Use case 4.)

grace_secs If the resource provider announces a shutdown time to the jobs on this host, that time will not be less than grace_secs seconds after the moment the shutdown time is set. This allows jobs to begin packages of work knowing that there will be sufficient time for them to be completed even if a shutdown time is announced. This value is required if a shutdown time will be set or changed which will affect any jobs which have already started on this host.

8 $JOBFEATURES

Job specific key/value pairs which are all:

- Found in the directory pointed to by $JOBFEATURES
- Readable and possibly owned by the user who is executing the original job. In the case of pilots this would be the pilot user at the site.

- Required unless the resource provider cannot determine their value.
- Created before the job starts and static unless otherwise stated, or unless the batch system has a recognised way of changing the parameters of the job in a way the job is guaranteed to be aware of. For example, if there is a mechanism for a job to release processors, then the resource provider may update allocated_cpu when this happens.

allocated_cpu Number of processors allocated to the current job. (Use case 5.)

hs06_job Total HS06 rating for the processors allocated to this job. The job's share is calculated by the resource provider from per-processor HS06 measurements made for the machine. (Use case 3.)

shutdowntime_job Dynamic value. Shutdown time as a UNIX time stamp in seconds. If the file is missing no job shutdown is foreseen. The job needs to have finished all of its processing when the shutdown time has arrived. (Use case 1.)

grace_secs_job If the resource provider announces a shutdowntime_job to the job, it will not be less than grace_secs_job seconds after the moment the shutdown time is set. This allows jobs to begin packages of work knowing that there will be sufficient time for them to be completed even if a shutdown time is announced. This value is static and required if a shutdown time will be set or changed after the job has started.

jobstart_secs UNIX time stamp in seconds of the time when the job started on the worker node. For a pilot job scenario, this is when the batch system started the pilot job, not when the user payload started to run. (Use case 2.)

job_id A string of printable non-whitespace ASCII characters used by the resource provider to identify the job at the site. In batch environments, this should simply be the job ID. In virtualized environments, job_id will typically contain the UUID of the VM.

wall_limit_secs Elapsed time limit in seconds, starting from jobstart_secs. This is not scaled up for multiprocessor jobs. (Use case 1.)

cpu_limit_secs CPU time limit in seconds. For multiprocessor jobs this is the total for all processes started by the job. (Use case 1.)

max_rss_bytes Resident memory usage limit, if any, in bytes for all processes started by this job. (Use case 8.)

max_swap_bytes Swap limit, if any, in bytes for all processes started by this job. (Use case 8)

scratch_limit_bytes Scratch space limit if any. If no quotas are used on a shared system, this corresponds to the full scratch space available to all jobs which run on the host. User jobs from EGI-registered VOs expect the "max size of scratch space used by

jobs" value on their VO ID Card [4] to be available to each job in the worst case. If there is a recognised procedure for informing the job of the location of the scratch space (eg EGI's $TMPDIR policy [5]), then this value refers to that space. (Use case 7.)

9 Summary

This note describes how the $MACHINEFEATURES and $JOBFEATURES variables may be set and used by jobs to obtain meta information from resource providers in a uniform way across different batch and virtual machine systems.

The following key/value pairs have been defined:

$MACHINEFEATURES	$JOBFEATURES
total_cpu	allocated_cpu
hs06	hs06_job
shutdowntime	shutdowntime_job
grace_secs	grace_secs_job
	jobstart_secs
	job_id
	wall_limit_secs
	cpu_limit_secs
	max_rss_bytes
	max_swap_bytes
	scratch_limit_bytes

References

[1] T. Cass, "Environmental Information on WN", Grid Deployment Board, CERN, 13 June 2012, retrieved from https://indico.cern.ch/event/155069/

[2] "Machine / Job Features Task Force", https://twiki.cern.ch/twiki/bin/view/LCG/MachineJobFeatures

[3] "HEP-SPEC06 Benchmark", https://w3.hepix.org/benchmarks/

[4] "The VO ID Card system", http://operations-portal.egi.eu/vo/help

[5] P. Solagna, "EGI policy for the TMPDIR environment variable usage", EGI Document 1119, retrieved from https://documents.egi.eu/public/ShowDocument?docid=1119