Paper SAS3430-2019

Introducing SAS® Workload Orchestrator, the New SAS® Grid Manager Workload Manager

Doug Haigh, SAS Institute Inc.

ABSTRACT

Prior to SAS® 9.4M6, there were two options for managing your workload on a SAS Grid—Platform Load Sharing Facility (LSF) from IBM and Apache Hadoop YARN. Starting with SAS 9.4M6, SAS will provide its own workload manager, SAS® Workload Orchestrator. This paper provides information about the new SAS Workload Orchestrator in the areas of design and configuration along with a comparison to the other workload managers that are supported for SAS® Grid Computing.

INTRODUCTION

Beginning with the initial release in SAS 9.1.3, SAS Grid Manager has supported the Platform Suite for SAS as its grid middleware provider. Platform Suite for SAS included Platform LSF as the workload manager and Platform Process Manager as its enterprise-class scheduling server. In SAS 9.4M3, SAS added SAS Grid Manager for Hadoop, taking advantage of Hadoop's YARN and Oozie subsystems for workload management and enterprise scheduling. Starting in 9.4M6, SAS now has its own grid provider, that contains SAS Workload Orchestrator for workload management and SAS Job Flow Scheduler for enterprise scheduling. The new SAS grid provider is now the standard offering in SAS Grid Manager, and the previous SAS Grid Manager product has been renamed to SAS Grid Manager for Platform.

The architecture of SAS Grid Manager is designed so that SAS products that leverage the grid do not need to know anything about the underlying grid provider. Code using grid-enabled SIGNON in SAS/CONNECT, code submitted by the SAS Grid Manager Client Utility (SASGSUB), or code that runs in a grid-launched workspace server do not need to change when moving from one grid provider to another.

The goal of this paper is to provide you with an understanding of the design, operation, configuration, and future of SAS Workload Orchestrator

DESIGN

TECHNOLOGY

SAS Workload Orchestrator is composed entirely of SAS threaded-kernel subsystems that serve as the fundamental building blocks of SAS 9 and SAS Viya products.

INTERFACE

SAS Workload Orchestrator was designed from the ground up with a REST API, enabling access from virtually any client technology or programming language. This REST API is used both by client applications and for inter-daemon communication, which effectively makes each daemon a web service. This allows administrators to open a single port in a firewall and be able to inspect or secure the traffic with known tools.

To make monitoring, management, and configuration as seamless as possible, an embedded HTML5 web interface is exposed by the master daemon. If a client attempts to access a host that is not the current master, it will be redirected to the current master.

SECURITY

Data-In-Motion

The interface to the grid and between the daemons in the grid uses REST APIs, which uses HTTP. Therefore, protecting the data in motion just requires support for the TLS protocol. Because SAS 9.4M6 does not currently have automatic SSL configuration like SAS Viya does, TLS support is off by default but can be enabled after installation and configuration.

Data-At-Rest

To protect sensitive information that might be stored in the grid (such as passwords), SAS Workload Orchestrator uses a customer defined grid password to derive a key that is used to encrypt all sensitive information. The password is stored in a file readable only by the SAS install account. The password is not retrievable through the REST API but can be updated by the administrator through the REST API.

Authentication

REST APIs and web services have some predefined methods for authentication. For the REST API between a client and the master, SAS Workload Orchestrator currently supports username/password through either basic authentication or form authentication as well as negotiate authentication using Kerberos.

For the REST API between daemons in the grid and the master, SAS Workload Orchestrator uses OAuth authentication where the grid password serves as the key and the OAuth token contains a claim representing the HMAC of the data being sent. The HMAC contains the grid passphrase, which ensures that the data has not changed and that the data is indeed from a daemon that is part of the grid.

INSTALLATION AND CONFIGURATION

The installation and configuration of the grid provider for SAS Grid Manager is part of the SAS Deployment Wizard. This eliminates the need for an initial and separate install of the grid provider. The SAS Deployment Wizard installs and configures the grid daemon with a basic configuration. After the SAS Deployment Wizard is finished, the grid administrator must update the grid configuration by pointing the browser at the master to bring up the embedded web interface. All parts of the configuration can be updated dynamically except for web service settings.

OPERATION

HOSTS

There are two ways a host can participate in the grid - as a master candidate or as a server. Master candidates and servers can run jobs or high-availability services. If the master candidate is the current master, it also manages job scheduling, host status, and high-availability service scheduling.

The first task of a host after it initializes is to determine the current master. Each host traverses the list of master candidates as defined in the configuration file until one responds. The host then considers that master candidate the master and sends information about itself to the master. The master candidate host that considers itself the master must

create a master file lock on the shared directory to prevent multiple master candidates from simultaneously assuming the role of the master (in other words, the split-brain problem).

Once a master is determined, the hosts announce themselves to the master and provide static host information, static resource information, and alias host names. The static information contains information such as the host name, operating system, and version. The static resource information includes predefined static resources such as total memory and user-defined host static resource values. The alias host names are names that the host can be known by to clients who use the grid.

During normal operation, the hosts periodically provide dynamic resource information and license information to the master so that the master knows the current state of the host for scheduling purposes. The master saves this information for the next host evaluation cycle, where it uses the hosts' information to determine the state of the host. If the master does not hear from the host within a certain amount of time, it will mark the host INACTIVE. If the host remains inactive for an extended period, it is removed from the grid. A host can also be in several states including UNLICENSED, OVERLOADED (resource value exceeds a predefined threshold), FULL (maximum job limit for host is met), NO_SLOTS (if maximum job limit is 0), or OK. Master candidate hosts never get removed from the grid, so if the master candidate host has not contacted the master, it could also be in a state of UNKNOWN.

QUEUES

Periodically, and for certain events, the current master evaluates queued jobs to see if they can be scheduled. The events that trigger a scheduling evaluation include things such as job submission, job completion, or host addition. When the master performs a scheduling evaluation, it finds the queue with the highest priority, tries to resume any preempted jobs, and then tries to start any pending jobs. If multiple queues have the same priority, the jobs are scheduled on a first-in-first-out (FIFO) basis (numerical order).

If a job can be scheduled, the list of hosts that can run the job is created by using the static resource information. The static resource specification can be defined both on the job request and the queue. It can include tags, required resources, and consumed resources that the host must have to run the job. Once the list is created, the hosts are ranked using the dynamic resource information according to the compare order defined in the configuration.

If a job is pending because the hosts it can run on are full and if the queue can preempt other queues, the scheduler tries to find a host the pending job can use that has running jobs from a preemptible queue. If a job from a preemptible queue is found, the master suspends that job and start the pending job from the preempting queue on that host.

JOBS (JOB CHANGE CMD, ROUND-ROBIN TIME)

Because every host in the grid can run jobs, every daemon has a job subsystem whose purpose is to run jobs and report status to the master. When a daemon receives a job to run, it sets the job state to STARTING and adds the job to the list of jobs managed by this daemon. Periodically, and on demand, the daemon traverses the list of jobs assigned to it, starts any job that is in the STARTING state, removes any job that is in a completed state, and sends the status to the master.

When a host goes through its dynamic resource update cycle, it also updates the dynamic resources for all jobs managed on the host. If a job has limits defined and a limit is exceeded, the job is terminated. If a suspend/resume threshold is set for the host, the dynamic resource defined for the host is evaluated to see if the lowest priority job needs to

be suspended or resumed. After all jobs have been updated, their status is sent to the master.

SERVICES

Periodically, and when hosts enter or leave the grid, the current master goes through all services defined in the configuration, and see if any service does not have as many instances running as the service requires. If that is the case, the services subsystem goes through all available hosts, matches them with the hosts defined in the service, and determines if there is one that is available and that is not currently running an instance. If so, a service request is sent to the daemon on the host so that it can be run. If the service fails to start on the specified host, the host is marked as not available for the specified service and the master looks for another host to start the service on.

Because every host in the grid can run services, every daemon has a service subsystem whose purpose is to run services and report status to the master. When a daemon is requested to run a service, it uses the user name and password defined in the service to run the prescribed script with the parameter '*start*'. If the service fails to start, an error is returned to the master. After the service instance is started, the service subsystem on the daemon periodically goes through all the instances it is managing and runs the prescribed script with the '*status*' parameter to determine the status of the service. If the status is a negative value, the daemon retries the status check up to five times before restarting the service. If the status is a positive value, the daemon restarts the service by calling the script with the '*restart*' parameter. If the service has been restarted five times, the daemon notifies the host that the service is dead, and the host is marked as being not available for the specified service so that the master looks for another host to start the service on.

RESOURCES

User-defined resources help extend the grid's scheduling by adding static and dynamic resources that are important to the user into the job scheduling process. User-defined resources come in several types - static global values that can be consumed, host static values that can be consumed, host static values that can be used for host selection, or host dynamic values that can be used for host ordering. Global static values are determined when a host becomes the master, either by running the script defined in the service configuration or by using the value defined in the service configuration. Host static values are determined by the host when it reports its presence to the master, either by running the script defined in the service configuration or by using the value defined in the service configuration. Host dynamic values are determined every time the host determines its dynamic resource values by running the script defined in the service configuration.

CONFIGURATION

The grid configuration is kept in a file in JSON format. The path to the file and the daemon's aliases are passed to the daemon when it is launched. All daemons parse the file to get the necessary information to work as part of the grid.

The grid configuration can be changed at any time through the REST API or through the web interface. The configuration takes effect immediately without the need for any additional commands if the change does not affect the web service parameters (port, authentication, or TLS parameters). The master daemon will send the configuration to all other daemons in the grid via the REST API so that all daemons use the same configuration.

GENERAL

There are several configuration options that specify how the SAS Workload Orchestrator daemon works and where it finds the things it needs to function. These options include path information for these files and directories:

- license file
- password file
- shared directory (for master candidates)
- GUI directory
- directory containing the GUI ZIP file
- script that is run to change the job request information

The options also specify the time (in seconds) for these events:

- time before jobs are purged
- time between host updates
- time before a job is determined to be inactive and unknown
- time to wait for a service script to complete
- time to use for round-robin scheduling

The configuration information includes web service information such as the following:

- port the web service uses
- whether TLS/SSL is used
- whether authentication is used

Finally, the configuration includes the list of grid administrators and the resources used in the host compare order.

HOSTS

A host configuration specifies these options:

- role a host performs (either master candidate or server)
- number of jobs the hosts can process simultaneously
- tags associated with the host (that is, Boolean resources)
- job scheduling threshold to apply to the host
- job suspend threshold to apply to the host
- time-based overrides to apply to the previous values

To enable the same configuration to be used against multiple hosts, SAS Workload Orchestrator configures host types, where all options for a host are applied to these hosts:

- hosts whose name matches the names defined
- hosts whose name matches the host name pattern
- hosts whose IP address matches an IP pattern
- hosts whose IP address falls within an IP range

Hosts are configured somewhat differently depending on whether they are master candidates or just servers. Master candidates must be specified by name, preferably using their fully qualified domain name. Servers can use their name or be matched against a host name pattern, IP pattern, or IP range. The order that master hosts are defined in the configuration is the order that the master candidates are checked by all daemons to determine the current master.

QUEUES

A queue configuration specifies the following:

- name of the queue
- description
- whether the queue is the default queue
- priority of the queue
- whether jobs in the queue can be restarted
- maximum number of running jobs from the queue
- maximum number of jobs running on any one host
- maximum number of jobs running by any one user
- users that can use the queue
- hosts, host groups, or host types that the queue can use
- administrators of the queue
- tags required by jobs from this queue
- resources required by jobs from the queue
- consumable resources required by jobs from the queue
- job limits applied to jobs from the queue
- queues that this queue can preempt
- time-based overrides for most of the default values

SERVICES

The definition of a high-availability service includes these items:

- service name
- description
- user account under which the service runs
- password for the user account (encoded in SAS00* or grid encoding)
- script file to run
- hosts, host groups, or host types on which the service runs
- number of active instances required

RESOURCES

The definition of a user-defined resource includes the following:

- resource name
- description
- resource type (global static consumable, host static consumable, host static, or host dynamic)
- script file path
- whether a larger value is better than a smaller value

HOST GROUPS

Host groups allow grouping of hosts together to make specifying hosts on queues and services easier. The definition includes these items:

- host group name
- description
- names of all hosts in the group

USER GROUPS

User groups allow grouping of users together to make specifying users on queues easier. The definition includes the following:

- user group name
- description
- names of all users in the group
- names of administrators of the group that can view and manage jobs for the group

COMPARISON TO LSF AND YARN

SAS Workload Orchestrator does not have all the features that can be found in Platform LSF. SAS Workload Orchestrator is customized for SAS workloads and therefore focuses on just the features needed by SAS Grid Manager.

Feature	SAS Workload Orchestrator	Platform LSF	Hadoop YARN
Preemptible priority queue-based scheduling	Yes	Yes	Yes
Fairshare based scheduling	No	Yes	Yes
Host selection by sorting on dynamic resource values	Yes	Yes	No
Host selection by static accounting	Yes	No	Yes
Job preemption by suspension	Yes	Yes	No
Pluggable scheduling logic	No	No	Yes
Time-based queue configuration	Yes	Yes	No
Time-based host configuration	Yes	Yes	No
Time-based host group configuration	No	Yes	No

Feature	SAS Workload Orchestrator	Platform LSF	Hadoop YARN
Time-based user group configuration	No	Yes	No
Time-based configuration time definition	Cron expression	Day of week, time of day	N/A
Scheduling based on dynamic resources	Yes	Yes	No
User-defined resources	Yes	Yes	No
Scheduling thresholds based on resources	Yes	Yes	No
Suspension thresholds based on resources	Yes	Yes	No
Scheduling dispatch windows	Yes	Yes	No
Queue active job limits	Yes	Yes	No
Terminate jobs when limit exceeded	Yes	Yes	No
Ability to change job information before it gets into the queue	Yes	Yes	No
Job request specifies minimum resource requirements	Yes	Yes	No
Job request specifies consumable resource requirements	Yes	Yes	Yes
Job request specifies Boolean resources	Yes	Yes	Yes
Job owner	Authenticated user on job submission	Process owner of job submission	Authenticated user on job submission
Authentication	Username/password, Kerberos	Process owner	Kerberos
Data-in-motion security	SSL	Proprietary	SSL
Data-at-rest security	File-based permissions. Sensitive data encrypted using AES128 with key derived from site-defined password (SHA256, 10000 iterations)	File-based permissions. Sensitive data encrypted with internally defined key, AES128	File-based permissions
End-to-end Kerberos to run jobs	Yes	No	No
Type of jobs that can be run	Any	SAS only	Any
Embedded GUI	Yes	No	Yes
Dynamic configuration	Yes	No	No
REST-API based	Yes	No	Yes
Configuration files	One	Multiple	Multiple
Ability to start/stop VMs as needed	No	Yes	No
Support parallel jobs for parallel frameworks like MPI	No	Yes	Yes

Feature	SAS Workload Orchestrator	Platform LSF	Hadoop YARN
Supported operating systems	Windows X64, Linux X64	All SAS v9 Windows and UNIX server platforms	Linux X64

FUTURE

Going forward, SAS Workload Orchestrator will be extended to manage additional workloads, including jobs and components that are part of a SAS Viya deployment. In addition, SAS Workload Orchestrator might look to include bursting capabilities, containerized jobs, job process integration into OS resource management controls (such as Linux cgroups and Windows job objects), and other scheduling methodologies.

CONCLUSION

SAS Workload Orchestrator provides a great initial entry into the workload manager marketplace and lays the foundation for managing SAS workload into the future.

REFERENCES

- "IBM Spectrum LSF V10.1 documentation." IBM. Available https://www.ibm.com/support/knowledgecenter/SSWRJV_10.1.0/. Accessed April 10, 2019.
- "Apache Hadoop Yarn." Apache. Available https://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/YARN.html. Accessed on April 10, 2019.

RECOMMENDED READING

- SAS Institute Inc, 2018, *Grid Computing in SAS 9.4.* 5th ed. Cary NC: SAS Institute Inc. Available: https://go.documentation.sas.com/?docsetId=gridref&docsetTarget=titlepage.htm&docsetVersion=9.4 (accessed April 10, 2019).

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Doug Haigh
SAS Institute, Inc.
doug.haigh@sas.com