Controlling Host Selection

There are several way to control which hosts the supervisor will select when deciding where to start job instances on the farm. These are based on restrictions that can be defined for both jobs and Workers.

- Hosts: Send only to an explicit set of workers
- Host Groups: Send only to an explicit pre-defined group of workers
- Host Order: Select hosts in a certain preferential order

Job and Worker Restrictions

Restrictions are used to allow or restrict where jobs run, and are applied to both jobs and Workers. Restrictions are based on cluster names. A job has preferential priority on a Worker whose cluster matches the job's cluster, but the job is free to run on any Worker in any other cluster, subject to the restrictions defined here.

Restrictions defined for jobs

When a job has a restriction defined, it means *only run on hosts that satisfy the restriction expression*. Hosts that don't satisfy the restriction expression won't be considered as dispatch candidates (the job will never be sent to that Worker).

Restrictions defined for Workers

When a Worker has a restriction defined via its worker_restrictions value, it means *only run jobs whose cluster value matches one of the clusters in that worker's restriction expression.* The worker won't accept jobs whose cluster doesn't match one of the clusters in the worker's restriction expression.

Restrictions Syntax

A restriction is really defined as a "filter" for hosts based upon information in the queuing algorithm; the values are one or more cluster names. In the priority/cluster queuing system, a user specifies the restrictions with a directory structure format:

`/[<segment>/][<segment>/][+|*]`

- * means *only the first level below*.
- **+** means *all levels below that level, regardless of depth* in the hierarchy.

The restriction value is actually evaluated as an expression, and multiple clusters are specified in a "this cluster **OR** that cluster **OR** the other cluster" type of string, with the " **||** " symbol to mean **OR.**

Examples

Job Restrictions

Syntax	Meaning
`qbsub -cl /private -restr /private <cmd>`	Submit a job that will have highest priority in /private and run *only* in /private
`qbsub -cl /private/very -restr '/private || /private/*' <cmd>`	Submit a job that will have highest priority in /private/very, but could run in *any host* in /private or in the first level below /private
`qbsub -cl /private/very/deep -restr '/private || /private/+' <cmd>`	Submit a job that will have highest priority in /private/very/deep, but could run in *any* host at any level at /private or below

Worker Restrictions

Syntax	Meaning
`worker_cluster = "/private/very/deep"` `worker_restrictions = "/private/very/deep"`	Define a host that will *only* run jobs in /private/very/deep

`worker_cluster = "/private"` `worker_restrictions = "/private || /private/*"`	Define a host that will run jobs in any cluster at */private or 1 level below* - done with the *****
`worker_cluster = "/private/very"` `worker_restrictions = "/private/very || /private/very/+"`	Define a host that will only run jobs in /private/very or any level below - done with the **+**

Hosts

Qube! allows users to specify a list of hosts, for the job to run on. This is a comma-delimited list of hostnames. You can also specify a list of hosts on which the job is restricted from running.

Examples

Command	Meaning
`% qbsub --hosts "qb001,qb002" command`	Run 'command' on hosts called qb001 and qb002
`% qbsub --omithosts "qb001,qb002" command`	Prevent 'command' from running on qb001 or qb002

Host Groups

Qube! allows the administrator to organize the farm into clusters or host groups. These groups have no hierarchy, and hosts are allowed membership in multiple groups. In order to restrict a job to a specific set of hosts, the user may specify a set of groups to restrict the job to. This goes into the 'group' field of the job. Similarly to Hosts, it is also possible to omit groups from consideration.

Examples

Command	Meaning
`% qbsub --groups "vfx,character" Render my/file.ma`	Render the Maya file called file.ma on any host in the 'vfx' or 'character' groups
`% qbsub --omitgroups "vfx,character" command`	Run 'command' on any host *not* in the 'vfx' or 'character' groups

Host Order

By default Qube! chooses any host (Worker) in the list of hosts which qualify. If given a choice, a job is allowed to prefer a particular host based upon its attributes. This is established using the Qube! resources and priorities defined earlier in the Requirements section of this document.

Any worker resource or property can be specified, but the most commonly used are:

- host.processors
- host.memory
- host.processor_speed

Syntax

[+|-]host.*property*
[+|-]host.*resource*.[total|used|avail]

The **+** or **-** in the expression is used to determine if the job would prefer the largest or smallest value possible. If neither is used, **+** is assumed.

Examples

Command	Meaning
`% qbsub --hostorder "host.processor_speed" Render myscene.ma`	Choose the fastest host
`% qbsub --hostorder "-host.processors.used" Render myscene.ma`	Choose the host with the least number of worker slots in use
`% qbsub --hostorder "host.processor_speed,host.processors.avail" Render myscene.ma`	Choose the fastest host with the most available worker_slots

Notes

The system will use the hostorder specification only when initially choosing the most preferable Worker for the job itself. Once it has chosen a host, it will try to fill it up with instances from the job until the host is full. In other words, the system will *not* attempt to apply the hostorder to select a host for each individual instance.

See Also

supervisor_default_hostorder
worker_cluster