Torque

How do I use a submit filter to enforce submission rules?

Issue: Need to force users to follow site rules.
Affected Version: All versions of Moab (some variations in releases)
Symptom: A site has job submission requirements that need to be enforced. A simple
example is to force users to specify a walltime. More complicated might be to reject jobs
requesting a specific queue unless certain requirements are met.
Solution: Most requirements of this nature can be enforced using a submit filter. This will
give users immediate feedback if they attempt to submit a job incorrectly, and can avoid
having jobs permanently block in the queue.
Both Moab and Torque have submit filters. If using Moab with Torque, all submitted jobs will
end up going through the torque submit fitler, if it’s defined, whether the user submits with
msub or qsub. Only jobs submitted with msub, however, will go through a Moab submit filter.
For both types of submit filters, the script can be accepted by printing the STDIN stream
received to STDOUT (modified if desired), and exiting with a 0 value. To abort the job, simple
exit with a “1” value.
The two biggest differences between Moab and Torque filters is the way the job is passed to
the filter. A Moab submit filter will receive job XML, and the job ID will not be defined. When
Moab submits it’s job to Torque, the job ID is available to the submit filter, and the job and the
msub command-line arguments are converted to a job script format.
Moab filter
Moab supports two types of submit filters. The client-side server filter is the one
recommended for validating user’s submissions. Note that a savvy user can bypass this
submit filter by redefining the “MOABHOMEDIR” environment variable, and creating a
configuration with no filter.
The filter receives it’s information in XML, which must be parsed to be useful. There are
libraries available that handle this, depending on the scripting language, or for simple cases,
the XML string can be searched.
The command-line options will be converted into XML and passed in the
<SubmitString>
element. The format is a string, with spaces and a few other characters converted to escaped

strings (eg. A space will be \20). The only command-line argument the script gets is the
name of the submit filter.
The script specified on the job submission, either on STDIN or on the command line, is
passed as the XML element <Executable>. If it was passed in STDIN, it’s pretty simple, but
if msub was given a script, there a sub-element (<SubmitExecutable>) that define the script
name and another sub-element (<Request>) that contains it’s own sub-elements for msub
options. It gets complicated pretty fast.
Torque Submit Filter (also referred to as qsub submission filter in the Torque documentation)
A Torque submit filter does not have to deal with XML. Everything is in text format. When the
job is submitted through qsub, the command line options are passed on to the submit filter.
Those options need to be parsed, along with any options defined in the script’s #PBS
directives, must be parsed. The order of this parsing is important, and will be discussed
below.
If a job is submitted via msub, the command-line options will have been converted into #PBS
directives, within the script provided on STDIN, and the only command-line argument is the

submit filter path. Another advantage is the job number will be known. The job number will
be provided in a line similar to this in the input: #PBS -W
x=SID:Moab;SJID:Moab.88;SRMJID:Moab.88. Note the job ID in both the “SJID” and
“SRMJID” fields. The SRMJID is the resource manager’s job ID, which at the time the submit
filter runs, will match the SJID, which is Moab’s job ID.
Best Practices
In general, if a complicated submit filter is going to be used, it’s best to restrict users to only
use one submit method. If qsub is the desired submission tool, then a simple msub submit
filter can be written to reject all jobs with a polite “please use qsub” message. If msub is the
desired submission tool, then a slightly more complicated Torque submit filter is required
since Moab will call qsub to submit jobs. This filter will have to lok for the line containing the
SJID (see previous section), reject jobs that don’t contain that line, and allow jobs that do
contain those strings to pass through. This allows a site to maintain a single filter.

Torque filters are usually easier to write, but there are some situations where a Moab filter is
required. For example, a Moab filter is require if options only understood by Moab’s msub are
to be added or altered during the processing, or if Moab actions need to be taken prior to the
job’s submission (for example, creating additional jobs with dependencies linked to the job
being submitted).

Parsing Directives Within the Job Script in a Torque Submit Filter

When a job is submitted with msub, the directives in the script and command line are parsed
and combined into job script #PBS directives, which is what a qsub filter will see. If using
qsub, however, all of the script options, along with all of the command-line options, must be
considered. This is simple in some cases, but more complicated in others.

In general, command-line options override any options defined in #PBS directives within the
script. If a #PBS directive is defined multiple times within a script, the last definition of the
option is used by Torque. If writing a filter, however, a good practice would be to at least warn
the user if this happens.

It gets a little more tricky when defining node resources, though. For example, if a script
defines a node request in a #PBS directive, a node request on the command-line will totally
redefine the node resource request, and any node details defined within the script are lost.

For this reason, it may be best to reject jobs with node definitions in both places, and politely
request the user choose one place or the other.

Feedback To Users

Any information writted to STDERR from within the submit filter will be displayed to the user if
the script aborts the job.

Providing Job IDs in Messages From a Moab Submit Filter

Since the job ID is not known when a Moab submit filter runs, the script can’t simply print out
a message for the user that contains the job ID. Moab does have a way to format a message
that will be displayed instead of the standard job ID. This is accomplished by inserting a
specific XML element into the XML returned to Moab. Moab also will look for the first
occurrence of the string “%s”, and insert the Moab job ID in it’s place. The format element
can be inserted right in front of the </job> element-end tag.

For example, for job ID “Moab.1234”, the string:

<OutputFormat>To check on your job status, run “checkjob -v %s”</OutputFormat>
would display this after the submit filter has returned and Moab has finished submitting the
job:

To check on your job status, run “checkjob -v Moab.1234”
This feature is not available to Torque submit filters.

Unique solution ID: #1204
Author: Shawn Hoopes
Last update: 2018-03-30 18:33