LA-UR-20-27701
Approved for public release; distribution is unlimited.

Title: BEE - FY20 P6-2: Support for HPC resource managers 2.3.6.01 – LANL ATDM ST / STNS01-6 P6 Milestone Completion Documentation

Author(s): Randles, Timothy C.

Intended for: ECP Milestone Completion Documentation

Issued: 2020-09-29

BEE - FY20 P6-2: Support for HPC resource managers

2.3.6.01 – LANL ATDM ST / STNS01-6 P6 Milestone Completion Documentation

Tim Randles
for the BEE Team

September 29, 2020

Managed by Triad National Security, LLC for the U.S. Department of Energy's NNSA

LA-UR-20-XXXXX

Activity Description

Support for HPC resource managers

BEE uses abstract interfaces between core functionality and specific driver code. An example of this is the worker_interface. The worker_interface is used by the BEETaskManager to communicate with the HPC system-specific resource manager (e.g. Slurm) and container runtime (e.g. Charliecloud). BEE initially only supported the Slurm HPC resource manager. This activity will make HPC resource manager support a configurable option. When this activity is complete BEE will support both the Slurm and LSF resource managers.

Execution Plan

To complete this activity we will:

- extend the BEETaskManager to use the BeeConfig class to determine which HPC resource manager (Slurm or LSF) is to be used according to the bee.conf file
- produce properly formatted LSF job scripts
- submit, query, and cancel jobs using the LSF resource manager

Implementation Highlight

Refactoring BEE's architecture is paying dividends

The same modular design highlighted by the completion of milestone 2.3.6.01/STNS01-5 (support for multiple container runtimes) continues to be validated with the completion of this milestone. The core BEE code uses a consistent API to interact with an HPC resource manager. The methods necessary to communicate with Slurm or LSF are implemented in a resource manager-specific driver that in turns implements the generic API used by BEE.

```
def submit_task(self, task):
    """Worker builds script and submits task as job returns job_id, job_state.

    :param task: instance of Task
    :rtype tuple (int, string)
    """
    return self._worker.submit_task(task)
```

`submit_task()` definition in generic API

```
def submit_job(self, script):
    """Worker submits job-returns (job_id, job_state), or (-1, error)."""
    job_st = subprocess.check_output(['bsub', script], stderr=subprocess.STDOUT)
    job_id = int(job_st.decode().split()[1][1:-1])
    job_state = self.query_job(job_id)
    return job_id, job_state

def submit_task(self, task):
    """Worker builds & submits script."""
    task_script = self.write_script(task)
    job_id, job_state = self.submit_job(task_script)
    return job_id, job_state
```

LSF driver

```
def submit_job(self, script, session, slurm_url):
    """Worker submits job-returns (job_id, job_state), or (-1, error)."""
    job_id = -1
    resp = session.post(f'{slurm_url}/job/submit', json={"job": {"environment": { "LD_LIBRARY_PATH": ""} }, 'script': sc
ript})
    if resp.status_code != 200:
        print(f'Error job submision failed with {resp.text}')
    else:
        status = json.loads(resp.text)
        job_id = status['job_id']
    _, job_state = self.query_job(job_id, session, slurm_url)
    return job_id, job_state

def submit_task(self, task):
    """Worker builds & submits script."""
    build_success, task_script, task_text = self.write_script(task)
    if build_success:
        job_id, job_state = self.submit_job(task_text,
                                            self.session, self.slurm_url)
    else:
        job_id = build_success
        job_state = task_script
    return job_id, job_state
```

Slurm driver

Note that the `submit_task` API is the same in each driver, but the `submit_job()` methods are very different for Slurm (RESTful API for job submission) and LSF (wrapped `bsub` command line call). The interface to the resource manager driver is consistent. Additional resource managers (e.g., PBS) can be supported simply by writing a new driver.

Completion Criteria

When this activity is complete the BEETaskManager will:

1. determine which HPC resource manager (Slurm or LSF) to use based on configuration data in the bee.conf file
2. format the HPC job script with the commands necessary for execution on the configured HPC resource manager
3. submit, query, and cancel jobs using the LSF resource manager

Completion Criteria

1. Determine which HPC resource manager (Slurm or LSF) to use based on configuration data in the bee.conf file

```
(beeflow-PKQVKgGh-py3.7) [trandles@login4.summit BEE_Private]$ grep -2 workload_scheduler ~/.config/beeflow/bee.conf
[DEFAULT]
bee_workdir = /autofs/nccs-svm1_home1/trandles/.beeflow
workload_scheduler = LSF

[graphdb]
```

bee.conf

LSF configuration and commands

```
(beeflow-PKQVKgGh-py3.7) [trandles@login2.summit client]$ cat ~/.beeflow/worker/clamr.sh
#!/bin/bash
#BSUB -P CSC420
#BSUB -W 0:30
#BSUB -nnodes 1
#BSUB -J clamr
#BSUB -o clamr.%J
```

job script

```
[trandles@fg-fey1 workflow-20200709-133043]$ cat ~/.config/beeflow/bee.conf
# BEE CONFIGURATION FILE #
[DEFAULT]
bee_workdir = /yellow/users/trandles/.config/beeflow
workload_scheduler = Slurm
```

bee.conf

Slurm configuration and commands

```
[trandles@fg-fey1 workflow-20200709-133043]$ cat clamr-6141516729113719703.sh
#! /bin/bash
#SBATCH
```

job script

2. Format the HPC job script with the commands necessary for execution on the configured HPC resource manager

```
(beeflow-PKQVKgGh-py3.7) [trandles@login4.summit BEE_Private]$ grep -2 workload_scheduler ~/.config/beeflow/bee.conf
[DEFAULT]
bee_workdir = /autofs/nccs-svm1_home1/trandles/.beeflow
workload_scheduler = LSF

[graphdb]
```

bee.conf

LSF configuration and commands

```
(beeflow-PKQVKgGh-py3.7) [trandles@login2.summit client]$ cat ~/.beeflow/worker/clamr.sh
#!/bin/bash
#BSUB -P CSC420
#BSUB -W 0:30
#BSUB -nnodes 1
#BSUB -J clamr
#BSUB -o clamr.%J
```

job script

LSF jobs scripts use “#BSUB” to denote submission options, while Slurm uses “#SBATCH”

```
[trandles@fg-fey1 workflow-20200709-133043]$ cat ~/.config/beeflow/bee.conf
# BEE CONFIGURATION FILE #
[DEFAULT]
bee_workdir = /yellow/users/trandles/.config/beeflow
workload_scheduler = Slurm
```

bee.conf

Slurm configuration and commands

```
[trandles@fg-fey1 workflow-20200709-133043]$ cat clamr-6141516729113719703.sh
#! /bin/bash
#SBATCH
```

job script

3. Submit, query, and cancel jobs using the LSF resource manager

```
(beeflow-PKQVKgGh-py3.7) [trandles@login2.summit client]$ python client.py
Welcome to BEE Client!
0) Submit Workflow
1) Start Workflow
2) Query Workflow
3) Pause Workflow
4) Resume Workflow
5) Cancel Workflow
6) Exit
$ 0
What will be the name of the job?
$ testjob
What is the workflow path?
$ cf-summit.cwl
Job submitted! Your workflow id is 42.
```

1. Workflow submitted using BEE client

```
Retrieved a title "testjob". What to do with it?
==== <class 'cwl_utils.parser_v1_0.Workflow'> ====
ins:  {'infile'}
outs: {'ffmpeg/outfile'}
gdbi passing kwargs
task:  clamr
  ins:      {'infile'}
  outs:     {'clamr/outfile'}
  command:  /CLAMR/clamr_cpuonly -n 32 -l 3 -t 5000 -i 10 -g 25 -G png
  hints:    {Requirement(req_class='DockerRequirement', key='dockerImageId', val
ue='/ccs/proj/csc420/BEE/clamr-ppc64le.tar.gz')}
task:  ffmpeg
  ins:      {'clamr/outfile'}
  outs:     {'ffmpeg/outfile'}
  command:  ffmpeg -f image2 -i /home/$USER/graphics_output/graph%05d.png -r 12
-s 800x800 -pix_fmt yuv420p /home/$USER/CLAMR_movie.mp4
  hints:    {Requirement(req_class='DockerRequirement', key='dockerImageId', val
ue='/ccs/proj/csc420/BEE/clamr-ppc64le.tar.gz')}
```

2. BEEWorkflowManager parses workflow and builds tasks

```
Submitted clamr to Task Manager
Task_id: 8487498901104158969 State PENDING
```

3. First task sent to BEETaskManager

```
Added clamr task to the submit queue
Job Submitted clamr: job_id: 375458 job_state: PENDING
```

4. BEETaskManager submits job to LSF

```
(beeflow-PKQVKgGh-py3.7) [trandles@login2.summit client]$ bjobs
JOBID   USER      STAT  SLOTS  QUEUE      START_TIME    FINISH_TIME   JOB_NAME
375458  trandles  PEND  -      batch      -             -             clamr
```

5. LSF shows pending job with same JOBID

3. Submit, query, and cancel jobs using the LSF resource manager

```
Added clamr task to the submit queue
Job Submitted clamr: job_id: 375458 ob_state: PENDING
Updated task state!
clamr PENDING -> RUNNING
```

```
Submitted clamr to Task Manager
Task_id: 8487498901104158969 State PENDING
Task_id: 8487498901104158969 State RUNNING
```

BEETaskManager updates BEEWorkflowManager with task state, which is reflected in the BEE client output when the workflow status is queried.

```
(beeflow-PKQVKgGh-py3.7) [trandles@login2.summit client]$ python client.py
Welcome to BEE Client!
0) Submit Workflow
1) Start Workflow
2) Query Workflow
3) Pause Workflow
4) Resume Workflow
5) Cancel Workflow
6) Exit
$ 2
What is the workflow id?
$ 42
STATUS
clamr--RUNNING
ffmpeg--WAITING

(beeflow-PKQVKgGh-py3.7) [trandles@login2.summit client]$ bjobs
JOBID   USER    STAT  SLOTS  QUEUE    START_TIME     FINISH_TIME    JOB_NAME
375458  trandles RUN   43     batch    Sep 29 10:03   Sep 29 10:33   clamr
```

Status is verified using the LSF `bjobs` command.

3. Submit, query, and cancel jobs using the LSF resource manager

```
container_runtime Charliecloud
Added clamr task to the submit queue
Job Submitted clamr: job_id: 374082 job_state: PENDING
Updated task state!
```

1. JOBID 374082, submitted via BEE, is pending.

```
(beeflow-PKQVKgGh-py3.7) [trandles@login1.summit client]$ python client.py
Welcome to BEE Client!
0) Submit Workflow
1) Start Workflow
2) Query Workflow
3) Pause Workflow
4) Resume Workflow
5) Cancel Workflow
6) Exit
$ 5
What is the workflow id?
$ 42
```

2. The workflow is canceled using the BEE client.

```
JobActions object at 0x7fff76d92128>
Workflow cancelled
```

3. The BEEWorkflowManager receives the command to cancel the workflow.

```
Updated task state!
Cancelling clamr with job_id: 374082
Added clamr task to the submit queue
```

4. The BEETaskManager is notified to cancel pending JOBID 374082.

```
(beeflow-PKQVKgGh-py3.7) [trandles@login1.summit client]$ bjobs -a -l

Job <374082>  Job Name <clamr>, User <trandles>, Project <CSC420>, Application
                    <small>, User Group <CSC420>, Status <EXIT>, Queue <batch>
                    , Job Priority <4>, Command <#!/bin/bash;#BSUB -P CSC420;#
                    BSUB -W 0:30;#BSUB -nnodes 1;#BSUB -J clamr;#BSUB -o clamr
                    .%J;module load charliecloud;mkdir -p /tmp;ch-tar2dir /ccs
                    /proj/csc420/BEE/clamr-ppc64le.tar.gz /tmp;ch-run /tmp/cla
                    mr-ppc64le --cd /home/$USER -- /CLAMR/clamr_cpuonly -n 32
                    -l 3 -t 5000 -i 10 -g 25 -G png;rm -rf /tmp/clamr-ppc64le>
Mon Sep 28 16:29:17: Submitted from host <login1>, CWD <$HOME/BEE_Private/beefl
                    ow/task_manager>, Output File <clamr.374082>, Re-runnable,
                     Requested Resources <1*{select[LN]span[hosts=1]} + 42*{se
                    lect[CN]span[ptile=42]}>;
Mon Sep 28 16:33:05 Exited
Mon Sep 28 16:33:05 Completed <exit>; TERM_OWNER: job killed by owner.
```

5. LSF JOBID 374082 is canceled as shown.