Slurm Burst Buffer Support

Tim Wickberg (SchedMD LLC)

SC15
Burst Buffer Overview

- A cluster-wide high-performance storage resource
- Burst buffer (BB) support added Slurm version 15.08
- Two types of BB allocations:
  - **Persistent allocations** used by multiple jobs or
  - **Job allocations** associated with a specific job
- BB allocations can exist before, during and/or after a job allocation of compute resources
  - Used to stage-in data, scratch storage, and/or stage-out data
Workflow

- Job submission specifies burst buffer requirements, validated at submit time
- Slurm allocates available burst buffer resources to the jobs expected to start soonest (allocations may later be revoked for higher priority job)
- Stage-in of files begins
- Compute nodes may be allocated after stage-in of files completes
- Stage-out of files begins upon completion of computation
- Job record may be purged after stage-out of files completes
Workflow

**Slurmctld Daemon**

1. Job submission
2. Set job expected start time
3. Allocate compute resources as available after stage-in completes
4. Job execution complete
5. Job record be purged after stage-out completes

**Burst Buffer Plugin**

1. Validate burst buffer specification
2. Allocate available BB resources to jobs expected to start soonest
3. Stage-in data
4. Stage-out data
5. Release BB resources after stage-out completes
6. Revoke and release BB

---

Copyright 2015 SchedMD LLC
http://www.schedmd.com
Workflow

- If burst buffer operation fails after job submission (e.g. file staging error) the job will be HELD and it's "Reason" field set to the error message from the API underlying the plugin

- New wait reasons
  - BurstBufferResources – Waiting for allocation of burst buffer
  - BurstBufferStageIn – Waiting for stage-in to complete

- New job error codes
  - ESLURM_BURST_BUFFER_PERMISSION - Burst Buffer permission denied
  - ESLURM_BURST_BUFFER_LIMIT - Burst Buffer resource limit exceeded
  - ESLURM_INVALID_BURST_BUFFER_REQUEST - Burst Buffer request invalid
Slurm Plugins

- Implemented using Slurm plugins to support various infrastructures
  - **Cray** – Uses Cray-specific APIs
    - BB is known as DataWarp on Cray systems
  - **Generic** – Uses generic scripts to manage burst buffers

- Slurm development stages
  - Generic plugin developed first
  - Cray plugin developed later, based upon generic plugin and required substantial changes to infrastructure
  - Generic plugin currently requires some updates
Burst Buffer Directives

- For batch jobs, specified as comments in the script
  - "#BB" prefix for generic plugin directives
  - "#DW" prefix for Cray DataWarp directives
    - Exception: Use "#BB" prefix to create/delete persistent burst buffers, operation not supported by Cray "#DW" directives

- For interactive jobs
  - Use "--bbf" option to specify path of file with same directives as batch job
  - Use "-bb" option to specify simple inline directives
Batch Job Examples

#!/bin/bash
# Allocate 100GB, stage-in one file, then stage-out one file
#DW jobdw capacity=100GiB type=scratch access_mode=striped,private
#DW stage_in type=file source=/home/my/input destination=/ss/input
#DW stage_out type=file source=/ss/output destination=/home/my/output
a.out

#!/bin/bash
# Allocate 2GB per compute node
#DW jobdw swap 2GiB
a.out
Interactive Job Examples

$ srun -bbf=bb.spec -N2 a.out
$ cat bb.spec
#DW jobdw capacity=100GiB type=scratch access_mode=striped,private
#DW stage_in type=file source=/home/my/input destination=/ss/input
#DW stage_out type=file source=/ss/output destination=/home/my/output

$ salloc --bb="capacity=100g" -N2 a.out

$ salloc -bb="swap=2g" -N2 a.out

NOTE: Slurm supports file staging for both batch and interactive jobs
Persistent Buffer Examples

#!/bin/bash
# Allocate 48GB persistent burst buffer
#BB create_persistent name=my_database capacity=48GB access=striped type=scratch

#!/bin/bash
# Use existing persistent burst buffer
#DW persistentdw name=my_database
a.out

#!/bin/bash
# Destroy the persistent burst buffer
#BB destroy_persistent name=my_database
Persistent Buffers

• The ability for normal user's to create persistent buffers through Slurm is configurable, disabled by default
  • See “Flags=EnablePersistent” in burst_buffer.conf
• Slurm polls system state and recognizes when persistent buffers are created or deleted
  • Changes to buffers made outside of Slurm (e.g. using DataWarp commands directly) are reflected in the accounting based upon the user's default account and QOS
  • Changes in the total burst buffer space are also accounted for (e.g. in case of failures)
Failure Management

- Persistent burst buffers will be created prior to a job beginning execution
  - Persistent burst buffers will not be destroyed even if a job is canceled prior to initiation (other jobs could have started to use them)
- Job-specific burst buffers will be staged-out only if a job starts running
  - Job-specific burst buffers will be staged-out even if the job failed, timed-out, was canceled from a running state, etc.
  - Job-specific burst buffers will be deleted without file staging if a job is canceled prior to initiation (no new data is generated if the job never starts)
Status Commands

- Current BB status visible using the sview and scontrol commands

```
$ scontrol show burst
Name=cray DefaultPool=dwcache Granularity=1GB TotalSpace=50GB UsedSpace=42GB
StageInTimeout=60 StageOutTimeout=60 Flags=EnablePersistent
AllowUsers=alan,brenda,charles
GetSysState=/home/jette/Desktop/SLURM/install.linux/sbin/dw_wlm_cli
Allocated Buffers:
  JobID=1234 CreateTime=2015-08-28T11:46:20 Size=4G State=allocated UserID=alan(1000)
  Name=my_db CreateTime=2015-08-28T11:46:20 Size=30G State=allocated UserID=alan(1000)
Per User Buffer Use:
  UserID=alan(1000) Used=34G
  UserID=brenda(1001) Used=8G
```

By Slurm convention: Job buffers have numeric name
Persistent buffer names start with a letter
Slurm Configuration: slurm.conf

- New *slurm.conf* options:
  - BurstBufferType – Defines the plugins to use, multiple BB plugins can be used on the same cluster
  - DebugFlags=BurstBuffer – Generates detailed logging of BB actions
Slurm Configuration: burst_buffer.conf

- New `burst_buffer.conf` file
  - May alternately be named with a specific BB plugin name (e.g. "burst_buffer_cray.conf")
- In the same directory as the `slurm.conf` file
- Contains BB-specific options
  - Script/API location
  - Timeouts
  - Access controls (Allow/Deny users)
- See "man burst_buffer.conf" for details
Advanced reservations

- Burst buffer resources can be reserved at specific times for specific users/groups
  - Ensure that burst buffers are available for critical uses
  - Optionally associated with specific compute nodes
Trackable Resource (TRes)

- Burst buffers are fully supported as a trackable resource
  - A job's burst buffer space requirements can be a factor in computing its priority
  - A job's burst buffer space consumption can be a factor in its fair-share usage calculation or changes
  - Burst buffer space consumption can be limited by association
Cray DataWarp Note

- Job requesting burst buffers will be allocated whole nodes only, not individual CPUs
  - Slurm implicitly sets job's –exclusive option
- DataWarp plugin finished
  - NERSC testing now on new Cori (XC40) system
Generic Implementation

- “Burstbuffer/Generic” plugin in progress
  - Minimal version in 15.08 now
  - Full version hopefully ready in 16.05 release
- Hooks to site-provided scripts to setup, stage-in/out data, mount filesystems, and tear-down allocated BB resources
Questions?