LSF (How to Run Jobs)

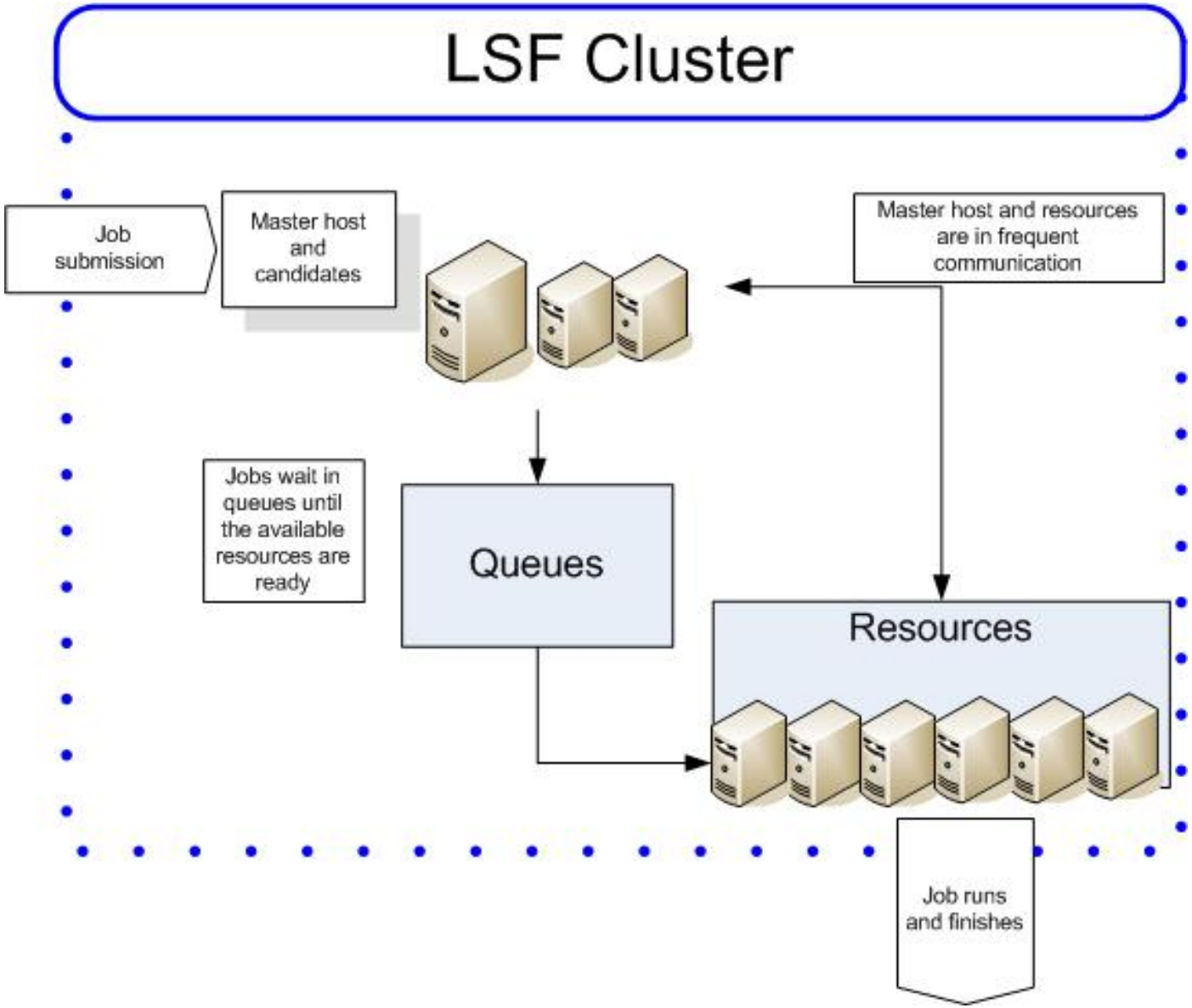
Ram Mohan Shrivastava

Sep 5, 2013

Introduction to Platform LSF

The Platform LSF ("LSF", short for load sharing facility) software is leading enterprise-class software that distributes work across existing heterogeneous IT resources creating a shared, scalable, and fault-tolerant infrastructure, delivering faster, more reliable workload performance while reducing cost. LSF balances load and allocates resources, while providing access to those resources.

LSF provides a resource management framework that takes your job requirements, finds the best resources to run the job, and monitors its progress. Jobs always run according to host load and site policies.



What is LSF Cluster

A group of computers (hosts) running LSF that work together as a single unit, combining computing power, workload, and resources. A cluster provides a single-system image for a network of computing resources.

Hosts can be grouped into a cluster in a number of ways. A cluster could contain:

All the hosts in a single administrative group

All the hosts on a sub-network

Hosts that have required hardware

Type of Hosts

Your cluster's hosts perform different functions.

Master host: An LSF server host that acts as the overall coordinator for the cluster, doing all job scheduling and dispatch.

Server host: A host that submits and executes jobs.

Client host: A host that only submits jobs and tasks.

Execution host: A host that executes jobs and tasks.

Submission host: A host from which jobs and tasks are submitted.

Some LSF Terms

Job

A unit of work run in the LSF system. A job is a command submitted to LSF for execution. LSF schedules, controls, and tracks the job according to configured policies.

Jobs can be complex problems, simulation scenarios, extensive calculations, or anything that needs compute power.

Job slot

A job slot is a bucket into which a single unit of work is assigned in the LSF system.

Hosts can be configured with multiple job slots and you can dispatch jobs from queues until all the job slots are filled. You can correlate job slots with the total number of CPUs in the cluster.

Some LSF Terms (Cont..)

Queue

A cluster-wide container for jobs. All jobs wait in queues until they are scheduled and dispatched to hosts.

Queues do not correspond to individual hosts; each queue can use all server hosts in the cluster, or a configured subset of the server hosts.

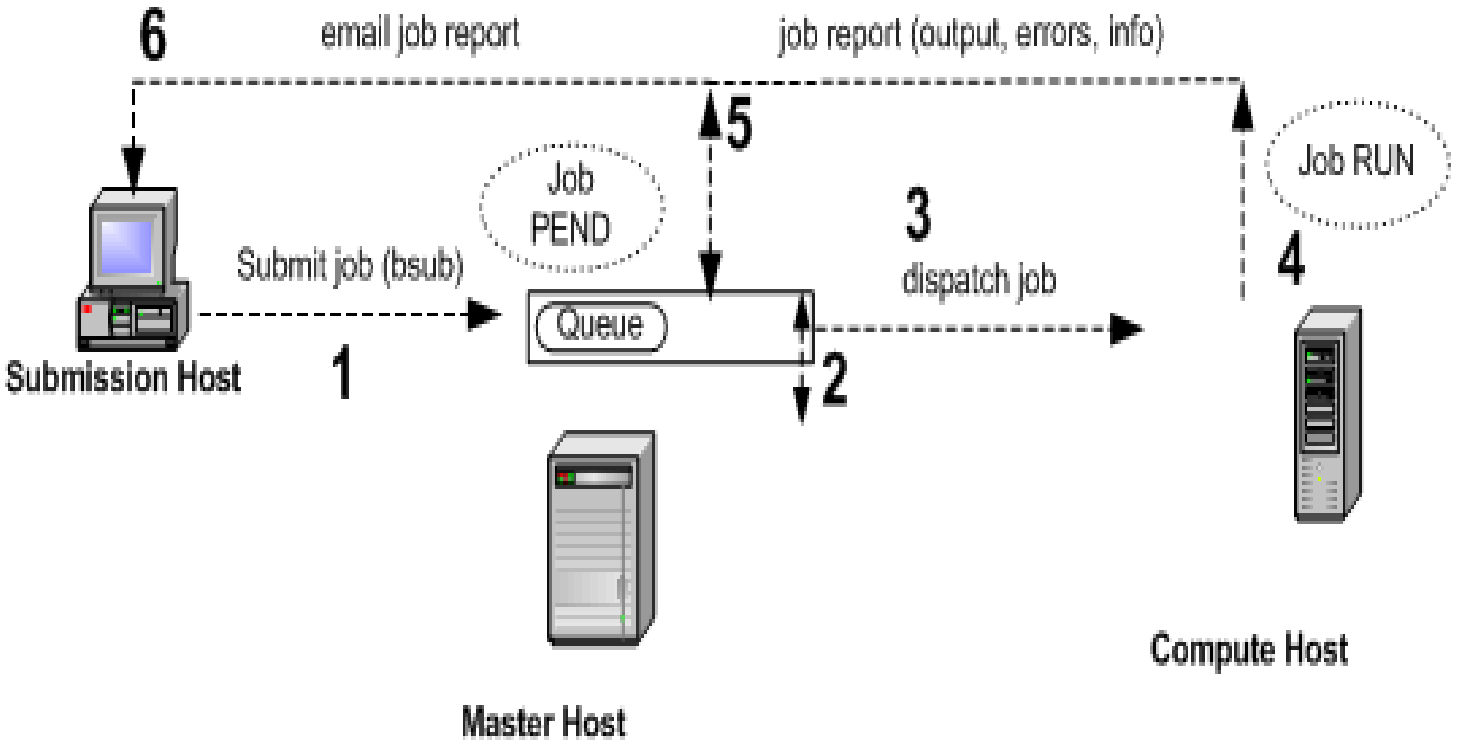
When you submit a job to a queue, you do not need to specify an execution host. LSF dispatches the job to the best available execution host in the cluster to run that job.

Queues implement different job scheduling and control policies.

Resources

Resources are the objects in your cluster that are available to run work. For example, resources include but are not limited to machines, CPU slots, and licenses.

Job life cycle



Working with Hosts

bhosts

Displays the current status of the host:

STATUS	Description
ok	Host is available to accept and run new batch jobs.
unavail	Host is down, or LIM and sbatchd are unreachable.
unreach	LIM is running but sbatchd is unreachable.
closed	Host will not accept new jobs. Use bhosts -l to display the reasons.
unlicensed	Host does not have a valid license.

bhosts -l

Displays the closed reasons (for details, see the bhosts command reference). A closed host does not accept new batch jobs:

```
bhosts
HOST_NAME          STATUS       JL/U    MAX  NJOBS   RUN  SSUSP  USUSP   RSV
hostA              ok            -      55    2      2    0      0       0
hostB              closed        -      20    16     16   0      0       0
...
```


Working with Hosts

lsload
Displays the current state of the host:

```
lsload
HOST_NAME status r15s r1m r15m ut pg ls it tmp swp mem
hostA      ok 0.0 0.0 0.0 4% 0.4 0 4316 10G 302M 252M
hostB      ok 1.0 0.0 0.0 4% 8.2 2 14 4231M 698M 232M
...
```

Close a host
Run badmin hclose:

```
badmin hclose hostB
Close <hostB> ...... done
```

If the command fails, it may be because the host is unreachable through network problems, or because the daemons on the host are not running.

Open a host
Run badmin hopen:

```
badmin hopen hostB
Open <hostB> ...... done
```

Paths of MPIs

Path of MPIs

Intel Cluster Studio

Installation path of Intel cluster studio is /opt/software/intel

How to set path of Intel Cluster Studio:

User should add below command in his/her .bash_profile file.

source /opt/software/intel/icsxe/2013.0.028/ictvars.sh intel64
(For 64 Bit)

source /opt/software/intel/icsxe/2013.0.028/ictvars.sh ia32
(For 32 Bit)

Or below script can be used to set path:

source /opt/software/intel/SetIntelPath.sh intel64
(For 64 Bit)

source /opt/software/intel/SetIntelPath.sh ia32
(For 32 Bit)

Platform MPI

Installation path of Platform MPI is /opt/platform_mpi

How to set path of Platform MPI:

User should add below command line in his/her .bash_profile file.

source /opt/platform_mpi/SetPMPIPath.sh

How to submit job using LSF （Intel MPI)

User need to create a job submission script like below:

```
#!/bin/bash
#BSUB -L /bin/bash
#BSUB -J HelloWorld    ## Specify Job Name
#BSUB -q normal        ## Specify Queue Name
#BSUB -o %J.out        ## Specify Output File
#BSUB -e %J.err        ## Specify Error File
#BSUB -n 32            ## Specify Number of Cores
#BSUB –N

mpirun -genv I_MPI_FABRICS shm:dapl -genv
I_MPI_DAPL_PROVIDER
ofa-v2-mlx4_0-1 –envall –genvall -np 32 ./Hello_Intel
```

Then Submit Job

```
$ bsub < Submit.lsf
```

How to submit job using LSF (Platform MPI)

User need to create a job submission script like below:

```
#!/bin/bash
#BSUB -L /bin/bash
#BSUB -J HelloWorld    ## Specify Job Name
#BSUB -q normal        ## Specify Queue Name
#BSUB -o %J.out        ## Specify Output File
#BSUB -e %J.err        ## Specify Error File
#BSUB -n 32            ## Specify Number of Cores
#BSUB -N

mpirun -np 32 -lsb_mcpu_hosts -IBV ./Hello_Intel
```

Then Submit Job

```
$ bsub < Submit.lsf
```

Managing Jobs

View all jobs for all users
Run bjobs -u all to display all jobs for all users. Job information is displayed in the following order:

Running jobs
Pending jobs in the order in which they are scheduled
Jobs in high-priority queues are listed before those in lower-priority queues

For example:

```
bjobs -u all
JOBID   USER    STAT    QUEUE      FROM_HOST   EXEC_HOST   JOB_NAME   SUBMIT_TIME
1004    user1   RUN     short      hostA       hostA       job0       Dec 16 09:23
1235    user3   PEND    priority   hostM                   job1       Dec 11 13:55
1234    user2   SSUSP   normal     hostD       hostM       job3       Dec 11 10:09
1250    user1   PEND    short      hostA                   job4       Dec 11 13:59
```

View jobs for specific users
Run bjobs -u *user_name* to display jobs for a specific user:

```
bjobs -u user1
JOBID   USER    STAT    QUEUE      FROM_HOST   EXEC_HOST   JOB_NAME   SUBMIT_TIME
2225    user1   USUSP   normal     hostA                   job1       Nov 16 11:55
2226    user1   PSUSP   normal     hostA                   job2       Nov 16 12:30
2227    user1   PSUSP   normal     hostA                   job3       Nov 16 12:31
```


Managing Jobs

Suspend a job
Run bstop *job_ID*.
Your job goes into USUSP state if the job is already started, or into PSUSP state if it is pending.

```
bstop 3421
Job <3421> is being stopped
```

The above example suspends job 3421.

UNIX bstop sends the following signals to the job:

SIGTSTP for parallel or interactive jobs—SIGTSTP is caught by the master process and passed to all the slave processes running on other hosts.

SIGSTOP for sequential jobs—SIGSTOP cannot be caught by user programs. The SIGSTOP signal can be configured with the LSB_SIGSTOP parameter in lsf.conf.

Windows bstop causes the job to be suspended.

Managing Jobs

Resume a job
Run bresume *job_ID*:

```
bresume 3421
Job <3421> is being resumed

resumes job 3421.
```

Resuming a user-suspended job does not put your job into RUN state immediately. If your job was running before the suspension, bresume first puts your job into SSUSP state and then waits for sbatchd to schedule it according to the load conditions.
.

Kill a job
Run bkill *job_ID*. For example, the following command kills job 3421:

```
bkill 3421
Job <3421> is being terminated
```


Working with Queues

Queue states

Queue states, displayed by bqueues, describe the ability of a queue to accept and start batch jobs using a combination of the following states:

Open: queues accept new jobs
Closed: queues do not accept new jobs
Active: queues start jobs on available hosts
Inactive: queues hold all jobs

Queue Information

The bqueues command displays information about queues. The bqueues -l option also gives current statistics about the jobs in a particular queue, such as the total number of jobs in the queue, the number of jobs running, suspended, and so on.

```
bqueues
QUEUE_NAME      PRIO STATUS          MAX JL/U JL/P JL/H NJOBS  PEND   RUN  SUSP
owners           43  Open:Active       -    -    -    -     0     0     0     0
priority         43  Open:Active       -    -    -    -     0     0     0     0
night            40  Open:Active       -    -    -    -     0     0     0     0
chkpnt_rerun_qu  40  Open:Active       -    -    -    -     0     0     0     0
short            35  Open:Active       -    -    -    -     0     0     0     0
license          33  Open:Active       -    -    -    -     0     0     0     0
normal           30  Open:Active       -    -    -    -     8     0     8     0
idle             20  Open:Active       -    -    -    -     0     0     0     0
```


Thank you