README: LSF 11.1 (Multi-Zone) Beta Release

Introduction

This README is for the beta release of LSF 11.1. This release introduces capabilities for running large LSF clusters (approximately 10k - 20k hosts). It also introduces a new approach to installation. Management daemons are deployed as containers, while LSF worker agents are deployed as RPMs.

In an LSF Multi-Zone cluster, hosts are partitioned into zones. Each zone is governed by its own set of management daemons (e.g. mbatchd, mbschd) that run on dedicated hosts of that zone. Management daemons coordinate across zones to provide capabilities that are similar to that of a standard LSF cluster, while allowing a for much greater scalability.

Each zone has one of two roles:

- **Submission.** Submission zones are dedicated to user interaction. There will be one or more submission zones. All jobs will be submitted via a submission zone.
- **Execution.** Execution zones are responsible for the scheduling and execution of jobs. Jobs cannot be submitted directly by users to execution zones.

In some cases, it may suffice to deploy a single submission zone. In cases where teams that use the cluster are distributed geographically, or where the expected job throughput is very high, it may be preferable to use multiple submission zones.

Submission zones are dedicated to user interaction. They do not necessarily need compute hosts, though it is allowed.

When migrating from a standard LSF cluster to a Multi-Zone cluster, you might initially convert the existing cluster to Multi-Zone with all the compute hosts in the submission zone. Over time, these hosts can be migrated to execution zones.

When deciding on the number of execution zones to deploy, consider the number of compute hosts, the expected job throughput, and the locality of the compute hosts. You might put hosts from different data centers (or cloud providers) in different zones. Even within a single data center, you may partition hosts into multiple execution zones to reduce the load on any one set of LSF management servers.

The following capabilities have been introduced to support LSF Multi-Zone:

1. **Pull-based job forward algorithm.** Prior to forwarding jobs, submission zones advertise their pending workload to all execution zones. A job is only forwarded once an

allocation has been made for it in an execution zone. This ensures more predictable job scheduling behavior. For efficiency, jobs with common resource requirements are aggregated into "buckets" in the submission zone. The protocol between submission and execution zones operates at the bucket level.

2. **Global job IDs.** This feature allows the LSF administrator to statically partition the job ID space among submission zones. When enabled, jobs to maintain a consistent ID on both the submission and execution zones.
3. **Query service.** A query service instance can be deployed in each submission zone to respond to user queries. The service helps to provide users with a global view of the cluster. For example, without the query service enabled, a bhosts query will talk to its local mbatchd, which will report on the hosts that are within the zone. With the query service enabled, bhosts can report on hosts in all zones.
4. **Global scheduling policies.** Most of the resource management policies that are offered in a standard LSF cluster can also be applied independently in each zone (e.g., fairshare, limits, guarantees, etc.). For cross-zone policies you can use existing cross-cluster capabilities such as global fairshare and License Scheduler. LSF Multi-Zone expands on these capabilities to support global resources and global limits.
5. **Centralized Configuration.** To simplify configuration management for multiple zones, all zones share a common set of configuration files. Macros can be used within the configuration files themselves to set per-zone configurations when needed.

Aside from the new LSF Multi-Zone architecture, the beta release also introduces a new mechanism for installing LSF. Management daemons are installed as containers. Compute daemons are installed as RPM packages. This will be the installation method for future releases of LSF.

Comparing LSF Multi-Zone and LSF Multi-Cluster

Multi-Zone and Multi-Cluster are two alternative approaches for aggregating the compute resources across an enterprise.

In a Multi-Cluster configuration, the component clusters are loosely coupled. Different clusters may have different sets of administrators, users, queues, etc. Users have little visibility into the resources and policies in remote clusters. Job forwarding decisions are driven by the submission scheduler based on high level information about the state of the remote cluster. This information may include, e.g., aggregate resource availability and pending workload. Since not all details of the remote cluster can be considered at the submission side, job forward decisions may be suboptimal.

In contrast, LSF Multi-Zone assumes there to be a common set of administrators, queues, users, etc. across the zones. All zones share a common configuration, for simplified management. Users have visibility into resources and policies across zones. As with Multi-Cluster jobs are forwarded between zones. However, ther than have forward decisions driven by the zone in

which a job is submitted, they are driven by the zone in which the job executes. This makes job scheduling more predictable.

Installation

Download LSF Images

1. From the technical preview web page, download file: `lsf-download-11.1.0-2021102615-beta.sh`

2. Register a free account on https://cloud.ibm.com/registration

3. Request download permission by sending email to:
   bill.mcmillan@uk.ibm.com , georgeg@ca.ibm.com

4. After receiving the invitation email from IBM, confirm to accept the invitation

5. Login `https://cloud.ibm.com` with the new account
   In ibm cloud web console, click on 'Manage' on the top menu, select 'Access(IAM)' , select 'API keys' on the left menu, click on 'Create an IBM Cloud API key'

6. Copy and paste the API key string for environment variable: API_KEY

Prepare the Environment

1. **Select a shared directory for the LSF configuration.** The file server for the shared directory must have at least 30 GB of free space available. All hosts must be able to access this directory as the same file path. This shared directory must have *no_root_squash* set for the installer host, GUI host, and the management hosts.

2. **Select a host for image download.** This host must have Docker (Docker CE 19 or above) installed, and have Internet access.

   Redhat 7 example:

   ```
   # yum install docker
   # systemctl enable docker (Optional, to automatically start the Docker service at boot time)
   # systemctl start docker
   ```

   SUSE 12 example:

   ```
   # zypper install docker
   ```

```
# systemctl enable docker (Optional, to automatically start the Docker
service at boot time)
# systemctl start docker
```

For more details, refer to SUSE documentation:

https://documentation.suse.com/sles/12-SP4/html/SLES-all/cha-docker-installation.html

3. **Select an installer host.** You can use the same host as the download host as your installer host. Install and start docker service on this host.

4. **Select management hosts.** Select hosts for the following roles:

   - LSF scheduler host
   - LSF GUI host
   - LSF server hosts
   - Database hosts
   - Cognitive host
   - RTM host

5. **Enable passwordless ssh for root on the the installer host.** The root user on the installer host must be able to access all to all other hosts.

   ```
   # ssh-copy-id -i ~/.ssh/id_rsa.pub HOST1
   # ssh HOST1
   ```

   When connecting for the first time, specify the root password for ssh to run properly. Repeat these commands for all hosts.

6. **Install container runtime for the hosts selected in step 4.** Docker and Singularity are supported. All hosts must use the same runtime.

   Docker: use CE 19 or above
   Singularity: use 3.7 or above

   For Singularity build and installation, refer to:

   https://sylabs.io/guides/3.7/admin-guide/installation.html#installation-on-linux

7. **Log into Docker Hub.** If you do not have a Docker hub account, first register an account:

   https://hub.docker.com/

   Then, then log in to the Docker hub as root on the installation host.

```
# docker login -u user -p password
```

8. **Get the IBM Cloud API key and lsf-download.sh script.** The API_KEY is included with the product offering.

9. **Install Kafka for LSF multi-zone.** Kafka is required when LSF cluster is running with multi-zone, For beta testing, you can choose to have the deployment create and start a container running Kafka server. This is for testing purpose, and there are some known issues. To install Kafka outside the deployment, follow the link: https://kafka.apache.org/quickstart

10. **Install network services library.** If the LSF server hosts are running RHEL 8, ensure that they have the library libnsl.so.1 installed.

Deploy LSF Suite 11.1 without LSF Multi-Zone

In this section, we describe how to perform a fresh install of an LSF Suite 11.1 cluster, without enabling LSF Multi-Zone.

1. **Download LSF.** Log in to the download host as root and navigate to the shared directory. Set the API_KEY environment variable and run the lsf-download-11.1.0-xxxxx.sh script.

```
# cd /SHARED_DIRECTORY_PATH
# export API_KEY=jBRd8OUr4lERBjIM8H2VzYohM4gmtEy5ZKvz_ArVyH6L
# ./lsf-download-11.1.0-xxxxx.sh
```

2. **Configure the installer**. Log into the installer host as root and navigate to the shared directory.

```
# cd /SHARED_DIRECTORY_PATH
```

   Edit the file named *lsf-inventory* to define the hosts that you selected for each LSF role. Edit *lsf-config.yml* to define the global parameters.

3. **Integrate with LDAP.** Edit and test the auth-config.sh script to define how LDAP is integrated into containers for sharing the same user database.

   a. Select a host from the management host candidates list. Log in to that host as root.
   b. Navigate to the /SHARED_DIRECTORY_PATH directory

```
# cd /SHARED_DIRECTORY_PATH
```

   c. Run the LSF Docker container.

```
# docker load -i image_repo/lsf-scheduler-amd64_11.1.0-2021102515.tgz
# docker run -it --network host -v `pwd`:/test --entrypoint=  \
cp.icr.io/cp/ibm-spectrum-lsf/lsf-scheduler:11.1.0-2021102515-amd64
bash
```

d. Edit the auth-config.sh script with the LDAP user settings.

```
# vi /test/auth-config.sh
```

e. Run the auth-config.sh script.

```
# /test/auth-config.sh
```

f. Test the password settings for the LDAP user. For example:

```
# getent passwd userA
```

g. Continue editing the auth-config.sh script (steps d, e, and f) until you have added all LDAP users and complete the setup.

4. **Run the deployment scripts.** These scripts deploy LSF Suite according to the role definitions in the lsf-inventory file.

```
# ./lsf-predeploy-test.sh
# ./lsf-deploy.sh
```

Deploy LSF Suite 11.1 with LSF Multi-Zone

In this section, we describe how to perform a fresh install of an LSF Suite 11.1 cluster, with LSF Multi-Zone enabled. We assume that there is shared storage that can be accessed by all management hosts. In a subsequent section, we describe how to set up a Multi-Zone cluster where the management hosts without shared storage.

1. **Download LSF.** Log in to the download host as root and navigate to the shared directory. Set the API_KEY environment variable and run the lsf-download-11.1.0-xxxxx.sh script.

```
# cd /SHARED_DIRECTORY_PATH
# export API_KEY=jBRd8OUr4lERBjIM8H2VzYohM4gmtEy5ZKvz_ArVyH6L
# ./lsf-download-11.1.0-xxxxx.sh
```

2. **Configure the hosts in each zone.** Copy the lsf-zones/zone_example file, using zone name as the file name and edit the zone inventory file. Create one inventory file per zone.

   - The LSF_Schedulers parameter is required for each zone.
   - Define LSF_Query_Servers for the submission zone to enable the query service (recommended). The query service collects information from across zones to answer user queries (e.g. bjobs, bhosts).
   - You cannot define the LSF_Query_Servers role in execution zones.
   - You can define the LSF_Lic_Schedulers role in same host with LSF_Schedulers of submission zone.

- You can define the LSF_Lic_Schedulers role in alone host of submission zone, which will be add as a LSF client automatically.
- The zone name cannot contain the period character.

```
# cp lsf-zones/zone-example lsf-zones/submit-zone01
# vi lsf-zones/submit-zone01
# cp lsf-zones/zone-example lsf-zones/execute-zone01
# vi lsf-zones/execute-zone01
```

Repeat the above commands per zone.

3. **Configure the installer.** Set the following parameters in *lsf-config.yml*:

```
CONTAINER_ENGINE: docker or singularity
convert2multizones: True
LSF_ZONE_FILE_PATH: /etc/lsf.zone
Kafka_Host: "Hostname"
SUBMISSION_ZONES: [ "submit-zone01", "submit-zone02" ]
```

- You must specify submission zones in lsf-config.yml file.
- Other zones that are defined in lsf-zones/ directory are treated as execution zones.
- The zone name cannot contain a dot.
- If you want to define hosts in the lsf-inventory file, the hosts will belong to an unnamed zone. Unnamed zones are intended to facilitate the migration from a standard LSF cluster to a Multi-Zone cluster.

4. **Integrate with LDAP.** Edit and test the auth-config.sh script to define how LDAP is integrated into containers for sharing the same user database.

h. Select a host from the management host candidates list. Log in to that host as root.
i. Navigate to the /SHARED_DIRECTORY_PATH directory

```
# cd /SHARED_DIRECTORY_PATH
```

j. Run the LSF Docker container.

```
# docker load -i image_repo/lsf-scheduler-amd64_11.1.0-2021102515.tgz
# docker run -it --network host -v `pwd`:/test --entrypoint=  \
cp.icr.io/cp/ibm-spectrum-lsf/lsf-scheduler:11.1.0-2021102515-amd64
bash
```

k. Edit the auth-config.sh script with the LDAP user settings.

```
# vi /test/auth-config.sh
```

l. Run the auth-config.sh script.

```
# /test/auth-config.sh
```

m. Test the password settings for the LDAP user. For example:

```
# getent passwd userA
```

n. Continue editing the auth-config.sh script (steps d, e, and f) until you have added all LDAP users and complete the setup.

5. **Run the deployment scripts.** These scripts deploy LSF Suite according to the role definitions in the lsf-inventory file.

```
# ./lsf-predeploy-test.sh
# ./lsf-deploy.sh
```

Deploy License Scheduler

In this section we describe how to deploy LSF License Scheduler.

1. **Log in to the installer host as root.** Navigate to the shared directory.

```
# cd /SHARED_DIRECTORY_PATH
```

2. **Mount license server binaries in containers.** Edit lsf-config.yml to define LMSTAT_PATH as SHARE_DIRS to point to the directory that contains lmutil/lmstat. For example:

```
SHARE_DIRS: [ "/home", "/license_app/flexlm_share_dir/bin" ]
```

3. **Identify License Scheduler management hosts.** If LSF License Scheduler has not been deployed yet, edit file lsf-inventory (multi-zone disabled) or lsf-zones/xxxx (multi-zone) to add LSF_Lic_Scheduler host, then run ./lsf-deploy.sh.

4. **Update License Scheduler configuration.** Edit lsf.licensesheduler, modifying the below parameters.

```
LMSTAT_PATH # the full path to the directory where the lmutil (or lmstat)\
command is installed
LIC_SERVERS # replace "port@server" with your own license server
```

5. Restart LSF. Restart LSF services on all management hosts.

```
# ./lsf-service.sh start -l host1,host2,host3,...
```

Convert a Standard LSF Cluster to Multi-Zone

If you have deployed a standard LSF cluster, and want to convert to a Multi-Zone deployment, follow the steps below. With these steps, hosts that are part of the original cluster will become part of the unnamed zone in the Multi-Zone cluster.

1. **Update the installer configuration.** Configure the following parameters in lsf-config.yml:

   ```
   convert2multizones: True
   LSF_ZONE_FILE_PATH: /etc/lsf.zone
   Kafka_Host: "Hostname"
   ```

2. **Enable the query service (recommended).** Set the LSF_Query_Servers parameter in the lsf-inventory.

3. **Add zones.** If you want to add new zones in addition to a single unnamed zone, copy the lsf-zones/zone-example file to include the zone name and configure the file with your settings. The LSF_Schedulers parameter is required. Note that you cannot define hosts in the LSF_Query_Servers role in execution zones. The zone name cannot contain a dot.

   For example, for the zone named submit-zone01 run the following commands:

   ```
   # cp lsf-zones/zone-example lsf-zones/submit-zone01
   # vi lsf-zones/submit-zone01
   ```

   Repeat these commands for each zone.

4. **Run the install script.**

   ```
   # ./lsf-deploy.sh
   ```

   The script converts the default installation to multi-zone mode according to the role defintions in the lsf-inventory file and all zones in lsf-zones.

Add hosts/zones to LSF Suite

1. **Prepare the hosts.** Prepare hosts to be added as described in the "Prepare environment" section.

2. **Log in to the installer host as root.**

3. **Update host inventory.** Edit lsf-inventory file to add hosts in proper role area or edit/create lsf-zones/<ZONE_INVENTORY> to add the hosts.

   ```
   # cd /SHARED_DIRECTORY_PATH
   # vi lsf-inventory
   # vi lsf-zones/execute-zone01
   ```

4. **Run the deployment script.** This script deploys LSF Suite according to the role definitions in the lsf-inventory file.

   # ./lsf-deploy.sh

Remove hosts/zones

In this section, we describe how to remove hosts or zones from an LSF cluster.

1. **Log in to installer host as root.**

2. **Run undeploy script.** The script removes either a zone or a host from the cluster.

```
# cd /SHARED_DIRECTORY_PATH
# ./lsf-undeploy.sh  -i lsf-zones/execute-zone01
# ./lsf-undeploy.sh  -l ib22b11
```

3. **Update the host inventory.** If you are only removing hosts, edit *lsf-inventory* and *lsf-zones/<ZONE_INVENTORY>* files to remove the hosts.

4. **Remove zone inventory.** If you are deleting whole zones, remove the lsf-zones/<ZONE_INVENTORY> file. If the zone you are deleting is the submission zone, remove the zone from the lsf-config.yml file.

Manage (start/stop) LSF services on hosts/zones

In this section, we describe how to start and stop LSF services.

1. **Log in to the installer host as root.** Navigate to the deployment share directory.

```
# cd /SHARED_DIRECTORY_PATH
```

2. **Run the lsf-service.sh.** This script starts and stops LSF services.

   a. To manage services on hosts:

```
./lsf-service.sh start|stop -l [host1,host2 ...]
```

   For example:

```
# ./lsf-service.sh stop -l ib22b11,ib22b14
```

   b. To manage services on zones:

```
./lsf-service.sh start|stop -i [zone1-inventory,zone2-inventory ...]
```

   For example:

```
# ./lsf-service.sh stop -i lsf-zones/execute-zone01,lsf-
zones/execute-zone02
```

Note that spaces are not allowed between host names and zone-inventory file names.

Update the deployment configuration

The lsf-config.yml file includes global deployment settings. You can change this file and apply the configuration to a running LSF Suite 11.1 installation.

1. **Log in to the installer host as root.** Navigate to the deployment share directory.

```
# cd /SHARED_DIRECTORY_PATH
```

2. **Edit the configuration.** The configuration is in the lsf-config.yml file.

```
# vi lsf-config.yml
```

3. **Run the deployment script.**

```
# ./lsf-deploy.sh
```

Patch upgrade or rollback

The LSF Suite deployment allows you to update versions or roll back version updates.

1. **Run lsf-undeploy.sh script.** Login installer host as root, then goto the install directory.

```
# cd /SHARED_DIRECTORY_PATH
# ./lsf-undeploy.sh
```

2. **Download new package**. Log in to download host as root, verify the new or old images available on IBM Cloud by API key. Get the new or old lsf-download-11.1.0-xxxxx.sh script for the intended version.

```
# cd /SHARED_DIRECTORY_PATH
# ./lsf-download-11.1.0-xxxxx.sh
```

3. **Run the deploy script.** Go back to installer host, deploys the LSF Suite according to the role definitions in the lsf-inventory file.

```
 # ./lsf-deploy.sh
```

Remove LSF Suite

1. **Log in to the installer host as root.** Navigate to the shared directory.

```
# cd /SHARED_DIRECTORY_PATH
```

2. **Run the undeploy script.** Deploys the LSF Suite according to the role definitions in the lsf-inventory file.

```
# ./lsf-undeploy.sh
```

Upgrade LSF 10.x to LSF Suite 11.1

You can upgrade an LSF 10.x cluster to LSF Suite 11.1 under the following conditions:

1. The LSF 10.x conf/ and work/ directories are installed on a shared file system.

2. LSF add-on products are not upgraded automatically. Each LSF add-on product needs to be upgraded manually and separately.

3. You can only upgrade the LSF 10.x cluster to LSF 11.1 default mode with multi-zone disabled.

Perform the following procedure to upgrade LSF 10.x to LSF Suite 11.1:

1. Shut down LSF 10.x on all hosts, then prepare the hosts as described in the "Prepare environment" section.

2. Log in to the installer host as root. That is, navigate two levels above conf/ and work/.

```
# cd /SHARED_DIRECTORY_PATH
```

3. Set the API_KEY environment variable and run the lsf-download-11.1.0-xxxxx.sh script. For example:

```
# export API_KEY=jBRd8OUr4lERBjIM8H2VzYohM4gmtEy5ZKvz_ArVyH6L
# ./lsf-download-11.1.0-xxxxx.sh
```

4. Edit the lsf-inventory file to define the hosts that you selected for each LSF role. Edit the lsf-config.yml to define the global parameters, for example: CONTAINER_ENGINE.

5. Edit and test the auth-config.sh script to define how LDAP is integrated into containers for sharing the same user database.

   a. Select a host from the management host candidates list, and log in to that host as root.
   b. Navigate to the /SHARED_DIRECTORY_PATH directory

```
# cd /SHARED_DIRECTORY_PATH
```

   c. Run the LSF Docker container.

```
# docker load -i image_repo/lsf-scheduler-amd64_11.1-211015.tgz
# docker run -it --network host -v `pwd`:/test --entrypoint=
cp.icr.io/cp/ibm-spectrum-lsf/lsf-scheduler:11.1-211015-amd64 bash
```

d. Edit the auth-config.sh script with the LDAP user settings.

```
# vi /test/auth-config.sh
```

e. Run the auth-config.sh script.

```
# /test/auth-config.sh
```

f. Test the password settings for the LDAP user, using getent passwd <LDAP_USER>. For example:

```
# getent passwd userA
```

g. Continue editing the auth-config.sh script (steps d, e, and f) until you add all LDAP users and complete the setup.

6. Deploy the LSF Suite according to the role definitions in the lsf-inventory file.

```
# ./lsf-predeploy-test.sh
# ./lsf-deploy.sh
```

Roll back to LSF 10.x from LSF Suite 11.1

You can only perform a rollback if the current LSF 11.1 Suite deployment was upgraded from LSF 10.x.

Rollback steps:

1. Log in to the installer host as root and navigate to the parent directory of the original LSF conf/ and work/ directories.

```
# cd /SHARED_DIRECTORY_PATH
```

2. Run the rollback script.

```
# ./lsf-rollback.sh
```

3. Start LSF 10.x normally by following the LSF 10.x process.

Deploy LSF Multi-Zone without Shared Storage

In this section we discuss how to add a zone to an LSF Multi-Zone without relying on shared storage between zones. For example, the new zone to be added to the LSF cluster may be in a different data center such as on a public cloud.

Prerequisites:

1. You have deployed LSF 11.1 with Multi-Zone enabled, and LSF is running properly.
2. You have a shared file system for the new zone.
3. You have the same user database for the new region. For example, the users share the same global LDAP.
4. You are using the same LSF eauth key for the new region. Get the eauth key in /SHARED_DIRECTORY_PATH/lsf/conf/lsf.sudoers from the first deployment

Steps:

1. Log in to the download host for the new zone as root and navigate to the shared directory.

```
# cd /SHARED_DIRECTORY_PATH
```

2. Set the API_KEY environment and run the lsf-download-11.1.0-xxxxx.sh script. For example:

```
# export API_KEY=jBRd8OUr4lERBjIM8H2VzYohM4gmtEy5ZKvz_ArVyH6L
# ./lsf-download-11.1.0-xxxxx.sh
```

3. Log in to installer host as root and copy the zone inventory files from the existing zone. You can also create new zone inventory files for this new region by following step 3 described in the section on deploying a Multi-Zone cluster.

4. Copy lsf-config.yml from the existing region to the SHARED_DIRECTORY_PATH directory in the new region. Plan and fill in the new submission zones and zones index by following step 4 of the section on deploying a Multi-Zone cluster.

5. Copy the auth-config.sh file from the existing region to the SHARED_DIRECTORY_PATH directory in the new region.

6. Deploy zones to the new region by specifying the zone inventory files for the new region only.

```
#./lsf-predeploy-test.sh -i [lsf-zones/submit-zone02,lsf-zones/submit\
-zone02,…]
#./lsf-deploy.sh -i [lsf-zones/submit-zone02,lsf-zones/submit-zone02,…]
```

7. Update lsf.shared and lsf.conf for all existing regions. Copy the zone configuration part of the last region that you added from the lsf.conf and lsf.shared files to the corresponding lsf.conf and lsf.shared files in all existing regions.

   lsf.shared:

```
Begin Zones
CLUSTERNAME ZONE       SERVERS       ROLE       INDEX
myCluster   subzone01  ( oxidases1 ) submission 1
myCluster   subzone02  ( bricoles1 ) submission 2
myCluster   execzone01 ( justify1 )  execution  -
myCluster   execzone02 ( afar1 )     execution  -
```

```
End Zones
```

lsf.conf:

```
#if zone(execzone01)
LSF_QUERY_SERVICE_HOSTS="oxidases1 bricoles1"
LSF_MGMT_LIST="justify1"
#elif zone(execzone02)
LSF_QUERY_SERVICE_HOSTS="oxidases1 bricoles1"
LSF_MGMT_LIST="afar1"
#elif zone(subzone01)
LSF_MGMT_LIST="oxidases1"
LSF_QUERY_SERVICE_HOSTS="oxidases1"
LSF_QUERY_SERVICE_PORT="6894"
#elif zone(subzone02)
LSF_QUERY_SERVICE_PORT="6894"
LSF_QUERY_SERVICE_HOSTS="bricoles1"
LSF_MGMT_LIST="bricoles1"
#endif
```

8. Plan and update the global policy, host groups, queues, and other configurations for all regions.

9. Stop and start all regions by using the lsf-service.sh script from each individual deployment directory.

Check the LSF cluster

After the LSF Suite deployment, the LSF management services are running inside the following containers: LSF Scheduler, LSF License Scheduler, LSF Data Manager, LSF GUI, LSF Explorer, LSF RTM, LSF Simulator, LSF Predictor.

The LSF server and client hosts are running in physical hosts.

Perform the following steps to check whether the LSF cluster is deployed properly.

1. If the LSF GUI is deployed, open the LSF GUI with the following URL:

```
https://lsf-gui-host:8443
```

   - Log in as the LSF administrator that is defined in the lsf-config.yml file.
   - Submit a job using the generic template and check that bthe job is running properly.

2. If the LSF GUI is not deployed, use the LSF CLI to check the LSF cluster. Log in to an LSF server host or LSF client host as a normal LDAP user and run the following commands:

```
$ lsid
$ lsclusters
$ lsload
$ bsub sleep 100
```

Log in to an LSF management host as root and log in to the container, for example, the LSF scheduler container:

```
# docker ps
# docker exec -it lsf-scheduler-01 bash
# ps -ef  # --- Check that all LSF scheduler process are running
```

Run LSF commands as the LSF administrator.

```
# su -l lsfadmin
$ bhosts
$ bsub sleep 111
```

Install additional packages in LSF management containers

LSF management services are run in containers, which only have bare minimum packages installed by default. In some cases, you might need to install additional packages in some these containers. The script auth-config.sh is executed when container start. This script can be customized to perform such installations.

For example, LSF Resource Connector requires Java to be installed in the scheduler container. Follow the steps below:

1. Assume you have run lsf-download-xxx.sh, you will have file auth-config.sh available in current directory

2. Add the following lines to the end of this file:

```
if [[ "$LSF_ROLE" == "scheduler" ]]; then
    yum install -y java-11-openjdk
fi
```

Note: environment variable LSF_ROLE is available for all LSF containers, the values list: scheduler, gui, datamanager, licensesched, explorer, rtm-server, rtm-client, rtm-licpoller, rtm-lsfpoller.

3. If the scheduler service is running, restart it.

```
# ./lsf-service.sh start -l <SCHEDULER_HOST1>,<SCHEDULER_HOST2>
```

If LSF is not deployed yet, continue to follow normal steps to deploy LSF

Configuring a Multi-Zone Cluster

In an LSF Multi-Zone cluster, all zones share a common configuration directory. When shared storage is not available across all zones, then the configuration directory may be copied between zones. It is the responsibility of the

Some of the configuration files in the configuration directory are specific to a single zone, while other files are used by all zones. Sharing as much of the configuration as possible across zones helps maintain a consistent set of policies across zones, simplifying management.

How zones are defined

Zones are defined in the Zones section of lsf.shared as in the following example.

```
Begin Clusters
CLUSTERNAME
bigclust
End Clusters

Begin Zones
CLUSTERAME  ZONE    SERVERS        ROLE         INDEX
bigclust       -    (host1)        submission   1
bigclust    sub1    (host2)        submission   2
bigclust    exec1   (host3)        execution    -
bigclust    exec2   (host4)        execution    -
End Zones
```

The fields of the Zones sections are:

- **CLUSTERNAME:** This must be the name of a cluster specified in the Clusters section.
- **ZONE:** The name of the zone. There can be a single zone with a dash in the ZONE field. This unnamed zone must have a submission role.
- **SERVERS:** The set of management hosts for the zone.
- **ROLE:** This can either be *submission* or *execution*. All jobs are submitted through a submission zone.
- **INDEX (optional):** Defining this field enables the global job ID feature. Each submission zone should be assigned a unique integer in the range 1 to 99. Execution zones should have a dash. The index partitions the job ID space among submission zones. The index of a zone determines the last two digits of all jobs submitted to the zone.

Configuration shared across zones

The following configurations are shared across zones.

- lsf.shared
- lsf.conf
- Batch configuration files (lsb.*)

For lsf.conf and the batch configuration files, we support macros in the configuration files that can be used to control which parts of the configuration are visible to a zone. This is useful for example in lsf.conf to specify the management hosts for each zone.

```
#if zone(-)
LSF_MGMT_LIST=”host1”
#elif zone(sub1)
LSF_MGMT_LIST="host2"
#elif zone(exec1)
LSF_MGMT_LIST=”host3”
#elif zone(exec2)
LSF_MGMT_LIST=”host4”
#endif
```

Whenever an LSF command or daemon starts up in a Multi-Zone environment, it will first try to determine its zone. This zone is used to evaluate the macros when preprocessing the configuration files. The zone is determined by:

1. **LSF_ZONE environment variable.** The binary will first check for the presence of this environment variable.
2. **Zone file on host.** A zone file is provisioned on each host of a Multi-Zone cluster. The contents of the file is simply the name of the zone. The location of the zone file is given by a macro in lsf.conf.

   ```
   #define LSF_ZONE_FILE "/etc/lsf.zone"
   ```

If neither the environment variable nor the zone file exists, then the process assumes it belongs to the unnamed zone.

Cluster file

Each zone will read its own cluster file with name of the form:

```
lsf.cluster.<clustername>.<zonename>
```

The contents of each cluster file is the same as a standard LSF cluster file.

Work directories

Each zone will have its own work directory with name of the form:

```
LSB_SHAREDIR/<clustername>.<zonename>
```

Validate the configuration

Use the lsclusters and bclusters commands to verify that the clusters are connected and ready to forward jobs.

```
mspriggs@ib01x06-63: lsclusters
CLUSTER_NAME   STATUS   MANAGEMENT_HOST ADMIN    HOSTS  SERVERS    INDEX ROLE
bigclust       ok       host1           lsfadmin     1        1        1 submission
bigclust.sub1  ok       host2           lsfadmin     1        1        2 submission
bigclust.exec1 ok       host3           lsfadmin     1        1        - execution
bigclust.exec2 ok       host4           lsfadmin     1        1        - execution


> bclusters
[Job Forwarding Information ]
LOCAL_QUEUE     JOB_FLOW   REMOTE     CLUSTER    STATUS
short           send       short      exec1      ok
normal          send       normal     exec1      ok
short           send       short      exec2      ok
normal          send       normal     exec2      ok

[Resource Lease Information ]
No resources have been exported or borrowed
```

If either lsclusters or bclusters shows that the clusters are not connected:

- Check the logs for the lim and mbatchd daemons on the management hosts.
- Ensure that all zones are using the same ports in lsf.conf.
- Check that for every queue that is defined in the lsb.queues file in the submission cluster, there is a queue with the same name that is defined in the lsb.queues file in every execution cluster.
- Check the status of the Kafka service in badmin showstatus.

Submit a job to a submission cluster. Assuming that there are sufficient resources to run the job in an execution cluster, ensure that the job is forwarded and dispatched.

```
> bsub sleep 10000
Job <52801> is submitted to default queue <normal>.

> bhist -l 52801

Job <52801>, User <lsfuser>, Project <default>, Command <sleep 10000>
Wed Mar 17 14:21:14: Submitted from host <host1>, to Queue <normal>, CWD </sc
                     ratch/lsfuser/sub1/log>;
Wed Mar 17 14:21:20: Forwarded job to cluster exec1;
Wed Mar 17 14:21:20: Job 52801 forwarded to cluster exec1 as remote job 52801;
Wed Mar 17 14:21:20: Dispatched 1 Task(s) on Host(s) <ns01x03@exec1>, Allocated
                      1 Slot(s) on Host(s) <host2@exec1>, Effective RES_REQ <
                     select[type == local] order[r15s:pg] >;
Wed Mar 17 14:21:21: Starting (Pid 2181);
Wed Mar 17 14:21:21: Running with execution home </home/lsfuser>, Execution CW
                     D </scratch/lsfuser/sub1/log>, Execution Pid <2181>;
Wed Mar 17 14:21:21: offer ID 1792309082@1616005278@exec1;


Summary of time in seconds spent in various states by Wed Mar 17 14:23:32
  PEND     PSUSP    RUN      USUSP    SSUSP    UNKWN    TOTAL
  6        0        132      0        0        0        138
```

Multi-Zone Job Forward Algorithm

While the Skyward architecture introduces a new algorithm for deciding job forwarding between zones, it uses the existing Multi-Cluster protocol for actually forwarding jobs and for managing the lifecycle of forwarded jobs.

Multi-Zone job scheduling

In this section we describe how job scheduling works in Multi-Zone.

Jobs submitted to a submission zone are organised into *buckets*. All the jobs in a bucket have identical scheduling attributes. For example, all the jobs in a single bucket have the same resource requirement, user, queue, etc. Typically, there will be far fewer buckets than jobs, so the bucket abstraction can be leveraged for more efficient scheduling.

Submission zones broadcast their workload requirements at the bucket level to the execution zones. Execution zones then allocate resources for the buckets. The execution zones report the number of allocations that are made for a bucket. Once an execution zone has allocated resources for a bucket, jobs in that bucket are forwarded. The messages of this protocol are passed via a Kafka message queue that is deployed automatically by the installer.

Skyward introduces the *badmin bucket view* command to help users understand the pull-based job forward protocol.

```
> badmin bucket help
Usage:    bucket view [-u user] [-l] [bucket_id] | [bucket_key]
```

In general, you will see a different output from this command from submission and execution sides. From the submission side, the output looks like the following:

```
> badmin bucket view
BUCKET     USER       QUEUE      ALLOCATED  NJOBS      PEND       FWD_PEND   STARTED
1682633140 lsfuser1   normal     7          9          2          0          7
1124913885 lsfuser1   normal     0          3          3          0          0
1169399896 lsfuser2   normal     6          7          1          1          5
```

The columns are:

- BUCKET: The bucket ID.
- USER: The user that corresponds to the bucket.
- QUEUE: The queue that corresponds to the bucket.
- ALLOCATED: The allocations that are made for this bucket. This is calculated as the number of running jobs from the bucket, plus the number of additional jobs that can be run with the reserved capacity.

- NJOBS: The number of jobs in the bucket.
- PEND: The number of jobs in the bucket in PEND state.
- FWD_PEND: The number of jobs in the bucket that are forwarded to an execution cluster but are still in the PEND state.
- STARTED: The number of jobs from the bucket that are currently running.

The output of this command is slightly different when you run it on the execution side:

```
> badmin bucket view
BUCKET     USER     SUB_CLUSTER   QUEUE      ALLOCATED  NJOBS      PEND     STARTED
1682633140 lsfuser1 sub1          normal     7          7          1        6
1124913885 lsfuser1 sub1          normal     1          1          0        1
1169399896 lsfuser2 sub1          normal     4          4          0        4
```

This output adds a new SUB_CLUSTER column, which gives the submission zone for the bucket. The output also omits the FWD_PEND column that is shown when the command is called from the submission side.

The output is given from the perspective of the execution side, which knows only about the jobs that have been forwarded to it. The NJOBS, PEND, and STARTED fields reflect only the jobs that are forwarded.

Use the badmin bucket view command with the -l option to get additional information about buckets, including:

- Additional scheduling attributes for jobs in the bucket.
- The quotas set by the submission zone in each of each execution cluster. Execution zones will stop trying to allocate additional resources for a bucket once it reaches its quota for the bucket.
- Pending reasons (on the execution side only).

Feature interactions

Limits

You can configure limits independently within each of the execution zones, for example, in the Limit section of the lsb.resources file. These limits apply on a single execution zone.

To configure limits that apply globally, use the LSF global limit policy feature:

https://www.ibm.com/support/knowledgecenter/SSWRJV_10.1.0/lsf_multicluster/global_limits_mc_lsf.html

Fairshare

As with limits, you can configure user Fairshare policies independently within each execution zone. Configure these policies using the FAIRSHARE parameter for queues that are defined in the lsb.queues file. Each execution zone independently manages its own fairshare policies.

Optionally, you can use the LSF global fairshare feature to deploy an enterprise-wide set of fairshare policies:

https://www.ibm.com/support/knowledgecenter/SSWRJV_10.1.0/lsf_admin/chap_global_fairshare_lsf_admin.html

Short job scheduling

The RELAX_JOB_DISPATCH_ORDER parameter in the lsb.queues and lsb.params files enables LSF to better handle short jobs. With this parameter enabled, LSF can run multiple jobs from the same bucket consecutively on the same resource allocation. This reduces average scheduling overhead per job, boosting cluster utilization.

To enable this in your Multi-Zone cluster set the following parameter in lsb.params.

```
RELAX_JOB_DISPATCH_ORDER = ALLOC_REUSE_DURATION[10 30] FORWARD_RATIO[2]
```

FORWARD_RATIO is a new keyword that is introduced for Multi-Zone forward, and defines the ratio of forwarded jobs from a bucket to allocations. A ratio of 2 means that if an execution zone allocates for 1 job, then the submission zone will forward up to 2 jobs. The idea is that we maintain a small backlog of jobs on the execution zone, so that when one job finishes another job will immediately start on the same allocation.

Guaranteed resources

Ensure that all submission and execution zones share a common set of service classes that are defined in the lsb.serviceclasses file. That is, the lsb.serviceclasses configuration file within each zone must be identical. When a job is submitted, the RBAC and auto-attach policies are evaluated on the submission side.

GuaranteedResourcePools are handled independently within each execution zone. Each execution zone has its own set of guarantee policies that are independently configured in the lsb.resources file.

Preemption

The LSF queue-based Preemption is supported as expected. Ensure that Preemption is configured in lsb.queues in one or more execution zones.

Note that when a high priority job is first submitted, LSF automatically delays considering Preemption for the job for 2 minutes. The purpose of this delay is to give time to see whether the job can start without Preemption in another cluster and avoid Preemption entirely. Adjust this delay time with the PREEMPT_DELAY parameter in the lsb.params file on the execution side.

Query service

The Skyward technology preview introduces a new query service that responds to LSF client queries such as bjobs, bqueues, and bhosts. The goals of this service are to:

1. **Provide a global viewpoint.** Without the query service, you need to query the zones of a Skyward deployment individually to get a complete picture. The query service aims to consolidate information from across clusters. For example, bhosts normally reports the hosts within a single zone. When using the query service, bhosts reports on hosts in all LSF execution zones.
2. **Reduce load on the LSF master host.** The query service acts as a proxy to the LSF management hosts. It can help reduce load on the management hosts by throttling queries and caching the results. It can also track events published by the clusters, such as job status events, and use this information to answer some queries without having to delegate the query to an LSF management host.

For this technology preview, the focus is on proving a global viewpoint. Currently, the query service acts as a proxy without throttling or caching. Skyward uses this technology preview to demonstrate what the consolidated output looks like. Enhancements to reduce load on the management hosts will be considered for general release.

Configuring the query service

Each instance of the query service is associated with a single submission zone. If there are multiple submission zones, enable a separate query service for each submission zone. A query service instance reports information from the perspective of a single submission cluster. For example, the query service that is associated with one submission cluster only reports on the jobs submitted to its own submission cluster.

To enable the query service for a submission zone, set the following parameters in the lsf.conf file:

```
LSF_QUERY_SERVICE_HOSTS="hostA hostB"
LSF_QUERY_SERVICE_PORT=6884
```

The LSF_QUERY_SERVICE_HOSTS parameter must define the hosts that belong to the submission zone. If there is only a single management host, define the management host as

that host. LSF automatically starts the query service daemons on the specified hosts. The LSF_QUERY_SERVICE_PORT parameter defines the listening port for the query service daemons.

When multiple query service daemons are deployed for a single submission zone, client commands select a random one from the list.

In all execution zones you must also set the following parameter in the lsf.conf file:

```
LSF_QUERY_SERVICE_HOSTS="hostA hostB"
```

In the execution zones, the LSF_QUERY_SERVICE_HOSTS parameter lists the query service hosts from all submission zones. This configuration is used by the execution clusters to validate communications from query service daemons in the submission clusters.
NOTE: LSF_QUERY_SERVICE_PORT is not allowed to be configured in execution zones.

After reconfiguring, restart the LIM daemon on hosts that are configured in the LSF_QUERY_SERVICE_HOSTS parameter and restart the management host LIM on each execution cluster by running the lsadmin reconfig command.

Using the query service

After enabling the query service, run a command to test it. For example:

```
# lsid
IBM Spectrum LSF Standard Edition 11.1.0.0, Oct 18 2021
Copyright International Business Machines Corp. 1992, 2021.
US Government Users Restricted Rights - Use, duplication or disclosure restricted by
GSA ADP Schedule Contract with IBM Corp.

My cluster name is cc.sub1
My management host is host01

# lsid -zone all
IBM Spectrum LSF Standard Edition 11.1.0.0, Oct 18 2021
Copyright International Business Machines Corp. 1992, 2021.
US Government Users Restricted Rights - Use, duplication or disclosure restricted by
GSA ADP Schedule Contract with IBM Corp.

My cluster name is cc.ex1
My management host is node01
------------------------------------------------------------------------------
IBM Spectrum LSF Standard Edition 11.1.0.0, Oct 18 2021
Copyright International Business Machines Corp. 1992, 2021.
US Government Users Restricted Rights - Use, duplication or disclosure restricted by
GSA ADP Schedule Contract with IBM Corp.

My cluster name is cc.ex2
My management host is node50
------------------------------------------------------------------------------
IBM Spectrum LSF Standard Edition 11.1.0.0, Oct 18 2021
Copyright International Business Machines Corp. 1992, 2021.
```

```
US Government Users Restricted Rights - Use, duplication or disclosure restricted by
GSA ADP Schedule Contract with IBM Corp.

My cluster name is cc.sub1
My management host is host01
```

For the technology preview, the query service supports the following commands:

- badmin
- bapp
- battr
- bhosts
- bimages
- bjobs
- blimits
- bmgroup
- bparams
- bqueues
- bresources
- brsvs
- bsla
- bslots
- bugroup
- busers
- lsadmin
- lsclusters
- lshosts
- lsid
- lsinfo
- lsload

For each query command, you can specify '-zone' to filter data out.

```
[-zone "zone_name ..." | -zone all]
```

Keyword 'all' means all information you can get from the large-scale LSF cluster.

The default display for query commands will show no information for zones. If '-zone' is specified, the display will have 'zone' displayed:

- In table output, new 'ZONE' column is added.

```
# bapp -zone all
APP_NAME         ZONE              NJOBS      PEND       RUN      SUSP
app1             ex1                   0         0         0         0
app1             ex2                   0         0         0         0
app1             sub1                  0         0         0         0
```

- In section output, zone description is showed up

```
# bparams  -zone all
Parameters configuration of zone <ex1>
Default Queues:  normal interactive
```

```
MBD_SLEEP_TIME used for calculations:  10 seconds
Job Checking Interval:  7 seconds
Job Accepting Interval:  0 seconds

Parameters configuration of zone <ex2>
Default Queues:  normal interactive
MBD_SLEEP_TIME used for calculations:  10 seconds
Job Checking Interval:  7 seconds
Job Accepting Interval:  0 seconds

Parameters configuration of zone <sub1>
Default Queues:  normal interactive
MBD_SLEEP_TIME used for calculations:  10 seconds
Job Checking Interval:  7 seconds
Job Accepting Interval:  0 seconds
```

Further, 'LSF_QUERY_SERVICE_THREADS=number' can be used to optimize query service on demand. It defines how many worker threads that query service created to response requests. The default value is twice the number of cores that the query service host has.

Global policies

You can configure several scheduling policies to take effect within a single execution zone. Sometimes you might want to enable scheduling policies that take effect globally. LSF has some capabilities for this including the following:

1. Global fairshare
2. Global limits
3. License Scheduler
4. Global Resources

You can use all these features together with the Skyward architecture.

This technology preview introduces new functionality to support global resources, include both static and dynamic global resources.

Enable global policy daemon

Set the following parameters identically in the lsf.conf file for all zones:

```
LSB_GPD_PORT=<port>
LSB_GPD_CLUSTER=<submissoincluster.zone>
```

LSF uses these parameters to determine in which cluster the global policy service starts, and which listening port to use. All clusters in a Skyward deployment share a common global policy daemon, which is used to coordinate scheduling between the clusters.

LSF uses these parameters to enable the global policy daemon for all included global limits, global resources, and global fairshare. These parameters make the previous GLOBAL_LIMITS parameter in the lsb.params file obsolete.

Note:
Need restart both lim and sbatchd after configuring these two parameters.

Configure global resources

Configure global resources in the file $LSB_CONFDIR/cluster_name/configdir/lsb.globalpolicies in the cluster where the global policy daemon starts, as shown in the following example:

```
Begin Resource
RESOURCENAME   TYPE    INTERVAL  INCREASING  CONSUMABLE RELEASE DESCRIPTION
gres1          Numeric ()        N           Y          Y       (global static res)
gres2          Numeric ()        N           Y          Y       (global static res 2)
gres3          Numeric 60        N           Y          Y       (global dynamic res)
gres4          Numeric 60        N           Y          Y       (global dynamic res 2)
End Resource

Begin ResourceMap
RESOURCENAME         LOCATION
gres1                (100@[all])
gres2                (10@[all])
gres3               ([all])
gres4               ([all])

End ResourceMap
```

Here define 4 global resources, gres1 and gres2 are static, gres3 and gres4 are dynamic.

To use the global dynamic resource, need configure gres.xxx and put it to $LSF_SERVERDIR. It's the same format as elim.xxx. For example,

```
#!/bin/sh
# list the resources that the gres can report to gpolicyd
my_global_resource="myrsc"
# do the check when $LSF_RESOURCES is defined by gpolicyd
if [ -n "$LSF_RESOURCES" ]; then
# check if the resources gres can report are listed in $LSF_RESOURCES
res_ok=`echo " $LSF_RESOURCES " | /bin/grep " $my_global_resource " `
# exit with $GRES_ABORT_VALUE if the gres cannot report on at least
# one resource listed in $LSF_RESOURCES
      if [ "$res_ok" = "" ] ; then
            exit $GRES_ABORT_VALUE
      fi
fi
while [ 1 ];do
      # set the value for resource "myrsc"
      val="1"
      # create an output string in the format:
      # number_indices index1_name index1_value...
      reportStr="1 $my_global_resource $val"
```

```
      echo "$reportStr"
      # wait for 30 seconds before reporting again
      sleep 30
done
```

Configure the reservation method in lsb.globalpolicies for global resource, the default method is PER_JOB.

```
Begin ReservationUsage
RESOURCE                    METHOD        RESERVE
gres1                       PER_HOST       Y
gres3                       PER_TASK       Y
End ReservationUsage
```

After reconfiguring lsb.globalpolicies run:

1) badmin gpdrestart
to restart the global policy daemon.

2) badmin mbdrestart in all zones.
to restart all the mbatchd.

Use bgpinfo resource -c to check all the configuration for the global resources.

```
$ bgpinfo resource -c
RESOURCE_NAME  TYPE    ORDER  INTERVAL  RELEASE CONSUMABLE    METHOD RESERVE
gres1          Numeric   Dec         0      Yes        Yes  PER_HOST     Yes
gres2          Numeric   Dec         0      Yes        Yes         -      No
gres3          Numeric   Dec        60      Yes        Yes  PER_TASK     Yes
gres4          Numeric   Dec        60      Yes        Yes         -      No
```

Using global resources

When a global resource is configured, jobs in the rusage section can request the global resource as usual:

```
bsub -R "rusage[gres1=10]" ./a.out
```

To query the availability of these global resources, use the following command.

```
$ bgpinfo resource
RESOURCE                  TOTAL        RESERVED
gres1                     100.0             0.0
gres2                      10.0             0.0
gres3                     100.0             0.0
gres4                      10.0             0.0
```

To see detail for all zones, run bgpinfo resource -l

```
$ bgpinfo resource -l
RESOURCE             CLUSTER               TOTAL        RESERVED
gres1                <ALL>                 100.0             0.0
                     big1501.sub1501       100.0             0.0
                     big1501.exe1502       100.0             0.0
                     big1501.exe1901       100.0             0.0
                     big1501.exe1903       100.0             0.0
gres2                <ALL>                  10.0             0.0
                     big1501.sub1501        10.0             0.0
                     big1501.exe1502        10.0             0.0
                     big1501.exe1901        10.0             0.0
                     big1501.exe1903        10.0             0.0
gres3                <ALL>                  99.0             0.0
                     big1501.sub1501        99.0             0.0
                     big1501.exe1502        99.0             0.0
                     big1501.exe1901        99.0             0.0
                     big1501.exe1903        99.0             0.0
gres4                <ALL>                  10.0             0.0
                     big1501.sub1501        10.0             0.0
                     big1501.exe1502        10.0             0.0
                     big1501.exe1901        10.0             0.0
                     big1501.exe1903        10.0             0.0
```

To check the wide format, use -w option.
To check the specific resource, use bgpinfo resource [-w/-l] -s <global_resource_name>.

You can also check the availability of these global resources with respect to a single zone by using the bhosts -s command.

Limitations and known issues

1. If you run the bswitch or bmod command on a job that is in the RUN state in an execution zone, the execution side does not count toward any bucket that is shown in the badmin bucket view command. On the submission side, the job counts as part of a new bucket.