IBM Spectrum LSF 10.1

Release Notes

Tables of Contents

Release notes for IBM Spectrum LSF Version 10.1

Read this document to find out what's new in IBM Spectrum LSF Version 10.1. Learn about product updates, compatibility issues, limitations, known problems, and bugs fixed in the current release. Find LSF product documentation and other information about IBM Spectrum Computing products.

- **What's new in IBM Spectrum LSF**
  Review the new and changed behavior for each version of LSF.
- **System requirements and compatibility**
  The following sections describe the system requirements and compatibility information for version 10.1 of IBM Spectrum LSF.
- **IBM Spectrum LSF product packages**
  The IBM Spectrum LSF product consists of distribution packages for supported operating systems, installation packages, and entitlement files.
- **Bugs fixed**
  LSF 10.1 releases and Fix Packs contain bugs that were fixed since the general availability of LSF.
- **Known issues**
  LSF 10.1 has the following known issues.
- **Limitations**
  LSF 10.1 has the following limitations.
- **Detailed changes**
  The following topics describe new and changed commands, options, output, configuration parameters, environment variables, accounting, and job event fields in LSF 10.1.

What's new in IBM Spectrum LSF

Review the new and changed behavior for each version of LSF.

- **What's new in IBM Spectrum LSF Version 10.1 Fix Pack 14**
  The following topics summarize the new and changed behavior in LSF 10.1 Fix Pack 14.
- **What's new in IBM Spectrum LSF Version 10.1 Fix Pack 13**
  The following topics summarize the new and changed behavior in LSF 10.1 Fix Pack 13.
- **What's new in IBM Spectrum LSF Version 10.1 Fix Pack 12**
  The following topics summarize the new and changed behavior in LSF 10.1 Fix Pack 12.
- **What's new in IBM Spectrum LSF Version 10.1 Fix Pack 11**
  The following topics summarize the new and changed behavior in LSF 10.1 Fix Pack 11.
- **What's new in IBM Spectrum LSF Version 10.1 Fix Pack 10**
  The following topics summarize the new and changed behavior in LSF 10.1 Fix Pack 10.
- **What's new in IBM Spectrum LSF Version 10.1 Fix Pack 9**
  The following topics summarize the new and changed behavior in LSF 10.1 Fix Pack 9.
- **What's new in IBM Spectrum LSF Version 10.1 Fix Pack 8**
  The following topics summarize the new and changed behavior in LSF 10.1 Fix Pack 8.
- **What's new in IBM Spectrum LSF Version 10.1 Fix Pack 7**
  The following topics summarize the new and changed behavior in LSF 10.1 Fix Pack 7.
- **What's new in IBM Spectrum LSF Version 10.1 Fix Pack 6**
  The following topics summarize the new and changed behavior in LSF 10.1 Fix Pack 6.

- **What's new in IBM Spectrum LSF Version 10.1 Fix Pack 5**
  The following topics summarize the new and changed behavior in LSF 10.1 Fix Pack 5. This Fix Pack applies only to IBM POWER9 platforms.
- **What's new in IBM Spectrum LSF Version 10.1 Fix Pack 4**
  The following topics summarize the new and changed behavior in LSF 10.1 Fix Pack 4
- **What's new in IBM Spectrum LSF Version 10.1.0 Fix Pack 3**
  The following topics summarize the new and changed behavior in LSF 10.1.0 Fix Pack 3
- **What's new in IBM Spectrum LSF Version 10.1 Fix Pack 2**
  The following topics summarize the new and changed behavior in LSF 10.1 Fix Pack 2
- **What's new in IBM Spectrum LSF Version 10.1 Fix Pack 1**
  The following topics summarize the new and changed behavior in LSF 10.1 Fix Pack 1
- **What's new in IBM Spectrum LSF Version 10.1**
  The following topics summarize the new and changed behavior in LSF 10.1.

What's new in IBM Spectrum LSF Version 10.1 Fix Pack 14

The following topics summarize the new and changed behavior in LSF 10.1 Fix Pack 14.

Download this Fix Pack from IBM Fix Central. For more information, see Getting fixes from IBM Fix Central.

Release date: June 2023.

Operating system versions

When a specific release or version of an operating system reaches end of life (EOL) or its end of support date, the operating system vendor no longer supports and releases updates for the product, including automatic fixes, updates, and online technical assistance.

LSF, in turn, is not tested on EOL operating systems. If you have extended support to continue to use EOL operating systems, you can use these operating system versions with LSF. Any issue with using LSF will be supported only if it can be reproduced on a supported operating system.

Before applying Fix Pack 14, ensure that you use one of the supported operating systems. Here are the highlights of LSF operating system changes as of Fix Pack 14:

- Ubuntu 20.04 is newly certified for Linux kernel on IBM Power LE.
- Ubuntu 22.04 is newly certified for Linux x64 kernel and Linux kernel on IBM Power LE.
  Also note for Ubuntu 22.04 support, the LSF installer has been refreshed on IBM Passport Advantage. If you have not or cannot run the new LSF installer, update the **`profile.lsf`** and **`cshrc.lsf`** environment shell files to use LSF on Ubuntu 22.04, according to the steps in the Fix Pack 14 readme file.

New LSF rate limiter

LSF now supports a rate limiter, to limit the rate of excessive requests to the **`mbatchd`** daemon, preventing stress on the cluster, and improving performance. The rate limiter is supported on Linux.

Leverage the rate limiter to tune performance
: The rate limiter acts as a gatekeeper between commands and the **`mbatchd`** daemon.

Furthermore, to allow an administrator to temporarily block non-administrator and non-root users, hosts, or both, from performing **mbatchd** daemon operations when using the rate limiter, the **badmin** command has been extended to support **badmin lsfproxyd block**. Administrators can run this command to temporarily stop abusive or misbehaving users from interacting with the LSF cluster, and to avoid performance impact on other users.

Enable diagnostic logs to monitor the rate limiter

Similar to using diagnostic logs to enable **mbatchd** to write query source information to a log file, you can now do the same for the new LSFrate limiter: **lsfproxyd**. The new query_info.queryproxylog.*hostname* log file shows information about the source of **lsproxyd** requests, allowing you to troubleshoot problems. The log file shows who issued these requests, where the requests came from, the data size of the query, the batch operation code, whether the request was rejected or accepted, and the time that the **lsfproxyd** daemon receives and processes the requests. Configuration for the **lsfproxyd** daemon logging is also the same as for **mbatchd** diagnostic logs.

Mark dynamic hosts as expired and remove them from the cluster

LSF administrators can mark dynamic hosts that are no longer available, as expired, using the new **lsadmin expire** command. Furthermore, if you have configured the LSF_DYNAMIC_HOST_TIMEOUT parameter, in the lsf.conf configuration file, then the hosts marked as expired will be automatically removed from the LSF cluster, cleaning up the environment so extra hosts do not affect cluster performance. In Fix Pack 14, the LSF_DYNAMIC_HOST_TIMEOUT has been enhanced to support more cleanup timeout options.

Global job IDs for using an LSF multicluster environment

Global job IDs allow an LSF multicluster environment to use the same job IDs between the forwarding and forwarded clusters, keeping the IDs uniform. These global job IDs are unique. To guarantee unique job IDs, LSF introduces indexes for clusters, so that each job submitted from the cluster includes an index to the ending digits of the job ID. To configure global job IDs for your LSF multicluster environment, see Global job IDs for forwarding and forwarded clusters using.

IBM Spectrum LSF Data Manager supports cross-cluster user account mapping for a multicluster environment

LSF Data Manager now supports cross-cluster user account mapping for a multicluster environment, allowing cross-cluster job submission and execution for a multicluster environment which has different user accounts assigned to different hosts. If you have LSF Data Manager installed, and enabled it for multicluster, you can additionally enable this cross-cluster user account mapping, configure your lsb.users configuration file on the remote and local clusters in your multicluster environment.

Support MIG device isolation

LSF now leverages cgroups to enforce Nvidia Multi-Instance GPU (MIG) device isolation. Set this enforcement by configuring LSB_RESOURCE_ENFORCE="gpu" in the lsf.conf configuration file.

Support DCGM version 3.1.8

Fix Pack 14 supports Nvidia Data Center GPU Manager (DCGM) version 3.1.8. Note, however, that Nvidia Multi-Instance GPU (MIG) integration with DCGM does not work with LSF.

Higher priority GPU jobs can now preempt shared-mode GPU jobs if there are multiple jobs running on the GPU

Prior to Fix Pack 14, higher priority GPU jobs with **j_exclusive=yes** or **mode=exclusive_process** settings could not preempt shared-mode GPU jobs, if there were multiple jobs running on the GPU. With Fix Pack 14, this restriction is removed so that higher priority jobs can preempt other jobs in this scenario. For details about submitting jobs that require GPU resources, see the LSB_GPU_NEW_SYNTAX parameter for the lsf.conf configuration file.

Support for key LSF daemons to restart upon failure

The existing /usr/lib/systemd/system/lsfd.service file is used to start, stop, or restart key LSF daemons (**LIM**, **RES**, and **sbatchd**) as a group, not for each daemon. For example, if only the **LIM** daemon fails (core dumps or abnormally exits), the lsfd.service file cannot restart just the one daemon.

Fix Pack 14 introduces enhanced behavior for the lsfd.service file: there is now a child service file for each LSF daemon (lsfd-lim.service, lsfd-res.service, and lsfd-sbd.service), to handle each daemon separately. If any of the key LSF daemons fail, each daemon can now automatically restart.

Note that if LSF is already installed on a server host, to leverage this enhanced behavior for the lsfd.service file, apply Fix Pack 14 and follow the steps in the Fix Pack 14 readme file (SPCLSF-866 section).

The **systemctl** command manages Linux **systemd** services, including starting, stopping, and restarting the LSF daemons. For more information about the **systemctl** command, see system daemon commands. Note that once you use **systemctl** commands to control these daemons, do not use the existing **lsadmin limrestart**, **lsadmin resrestart**, or **badmin hrestart** control commands. Use the **systemctl** commands instead.

Automatically create cgroups for a job, without affinity[] requirements

Previously, LSF created a cgroup for a job only when you explicitly specify affinity[] requirements. With the new LSF_CGROUP_CORE_AUTO_CREATE parameter in the lsf.conf configuration file, you can now enable LSF to automatically create Linux cgroups for a job. With this setting, LSF automatically adds the **"affinity[core(1)]"** resource requirement string to the current job's resource requirement whenever jobs are submitted.

Assign resource values to be passed into dispatched jobs

Administrators can now assign a specific resource value when defining and mapping shared resources in a cluster. This way, when jobs are dispatched, they are associated with the assigned resource requirement. For example, consider an FPGA device as a shared hardware resource, where you have three types of FPGA devices: **card1**, **card2**, and **card3**; you can assign **card1**, **card2**, and **card3** to the **fpga** resource using the **ResourceMap** section of the lsf.cluster.*cluster_name* configuration file (see lsf.cluster for an example **ResourceMap** section). The file has now been extended to enable LSF to assign specific values to resources, rather than only accept the number of resources.

Note that in Docker or Podman, the characters `!@#$%^&*()+-.` are not allowed in environment variable names. To use this feature within a Docker or Podman environment, ensure that your host names and resource names do not contain these characters. Specifically, for hostnames, you can ensure that the LSF_STRIP_DOMAIN parameter in the lsf.conf configuration file is set to remove all periods (`.`) from the hostnames if a period exists, and manually ensure that all resource names do not include unsupported characters.

With Fix Pack 14, you can assign which resources to use, not only how many of each resource. LSF commands provide the following enhanced output and usage:

- Running the **bjobs -l** or **bhist -l** command now shows the names assigned to a resource, which dispatches with jobs. The information will be in the header of the job in this format. For example, `Assigned resource <fpga___hostname> with names <card1 card2 card3>`
- The **bhosts** and **lshosts** commands now support a new -sl option to show the resource names assignments.

Support to display all field names for -o option in bhosts, bjobs, bqueues, and busers commands

You can now specify a new `all` keyword for the -o custom output option when running the **bhosts**, **bjobs**, **bqueues**, or **busers** commands. Specifying `-o "all"` indicates for LSF to display all (versus specific) field names when showing the output for these commands.

New BSUB_QUIET values for bsub command job submission output

The **BSUB_QUIET** environment variable, which controls job submission printed output for the **bsub** command, now supports new values to print the job ID only, and to print both the job ID and queue type. See BSUB_QUIET for details on these new BSUB_QUIET values.

New host memory output fields for the bhosts command and the -o option

The **bhosts -o** command shows hosts and their static and dynamic resources, in a customized output format. With the -o option, you specify one or more field names to display in the command output, and Fix Pack 14 supports three new field names:

- `available_mem` displays the total currently available memory on the server (if processes outside of LSF uses memory on that server, this value decreases).
- `reserved_mem` displays the currently reserved memory on a host.
- `total_mem` displays the total memory available and reserved by LSF on a host.

More CPU and memory usage details added to bjobs, bhist, and bacct commands

Fix Pack 14 provides more detailed information for CPU and memory usage for running jobs:

- You can now review the average CPU efficiency, CPU peak usage, duration to get to the CPU peak for a job, and CPU peak efficiency values, and see this level of detail when you run the **bjobs -l**, **bhist -l**, or **bacct -l** commands. Refer to the to the `CPU USAGE` and `Accounting information about this job` sections in the output for these commands to see the new CPU usage details. Additionally, you can see memory usage efficiency values in the output.
- You can also see the average CPU efficiency, or the duration to reach the CPU peak, for a job by running the **bjobs -o** command with these new field name options:
  - **bjobs -o average_cpu_efficiency**
  - **bjobs -o cpu_peak_reached_duration**
- The `cpu_efficiency` field name for the **bjobs -o** command now has a more descriptive name: `cpu_peak_efficiency`, which differentiates it from the other CPU usage field names. To see the value, run **bjobs -o cpu_peak_efficiency** if you are using Fix Pack 14 or later.

Support for the LSB_CLEAN_PDONE_TIMEOUT parameter for the lsb.params file

LSF cleans finished jobs in PDONE state; however, some jobs can remain in DONE state and never move into PDONE state and remain in mbatchd core memory, increasing the lsb.event file size unnecessarily. Use the LSB_CLEAN_PDONE_TIMEOUT parameter to indicate the maximum amount of time to wait, before cleaning out the finished jobs that remain in DONE state that cannot move to PDONE state. LSF multiplies the value of LSB_CLEAN_PDONE_TIMEOUT by the number of CLEAN_PERIOD seconds (defined in the lsb.params configuration file), to determine the number of seconds to wait before the mbatchd daemon cleans those jobs from core memory.

You can run the **bparams -a** or **bparams -l** command to view the LSB_CLEAN_PDONE_TIMEOUT parameter as a configurable parameter in the lsb.params file.

New MAX_PREEXEC_FORWARDING_RETRY parameter for the lsb.params file

Fix Pack 14 introduces a new MAX_PREEXEC_FORWARDING_RETRY parameter to the lsb.params configuration file, which at the submission cluster, controls the maximum number of times to attempt forwarding a job to a remote cluster, if the job has been rejected due to the pre-execution command of the job.

If the job's pre-execution command fails all attempts, the job is recalled by the submission cluster.

If the job fails all forwarding attempts, the job is suspended or terminated based on the LOCAL_MAX_PREEXEC_RETRY_ACTION setting. If the suspended job resumes, the counter will be reset

You can run the **bparams -a** or **bparams -l** command to view the MAX_PREEXEC_FORWARDING_RETRY parameter as a configurable parameter in the lsb.params file.

New MAX_NUM_JOBS_KILL_OVER_RUNLIMIT parameter for the lsb.params configuration file

If many (such as thousands of) jobs reach their run limit, it takes time for the **mbatchd** daemon to kill those jobs using the KILL_JOBS_OVER_RUNLIMIT parameter. Fix Pack 14 introduces a new parameter to the lsb.params file, called MAX_NUM_JOBS_KILL_OVER_RUNLIMIT, which controls the maximum number of jobs that can be killed by KILL_JOBS_OVER_RUNLIMIT in one cycle. This prevents the **mbatchd** daemon from potentially hanging when there are too many jobs to handle.

New default values for the LSF_DATA_SCP_CMD and LSF_DATA_SSH_CMD parameters within the lsf.conf configuration file

As of Fix Pack 14, the default values for the LSF_DATA_SCP_CMD and LSF_DATA_SSH_CMD parameters have changed as follows:

- `LSF_DATA_SCP_CMD="scp -q -o UserKnownHostsFile=/dev/null -o StrictHostKeyChecking=no -o PasswordAuthentication=no -o BatchMode=yes -r -p"`
- `LSF_DATA_SSH_CMD="ssh -q -o UserKnownHostsFile=/dev/null -o StrictHostKeyChecking=no -o PasswordAuthentication=no -o BatchMode=yes"`

Previously, the LSF_DATA_SCP_CMD and LSF_DATA_SSH_CMD parameters affected job transfer scripts. As of Fix Pack 14, however, these parameters no longer affect job transfer scripts; they affect the **bstage** command when the staging area is not mounted on the execution host of the user's job instead. For an LSF administrator to change how job transfer scripts use `scp` or `ssh`, they must modify the job transfer scripts directly.

New RC_ACCOUNT and VM_TYPE fields for the `JOB_FINISH2` LSF event

The `JOB_FINISH2` LSF event contains details about LSF resource connector jobs. The LSF lsb.stream file can then capture and stream actions about the `JOB_FINISH2` event. To provide more details in the `JOB_FINISH2` event logs, LSF now includes the RC_ACCOUNT and VM_TYPE fields with the `JOB_FINISH2` event. For details on how to enable these new fields, see Viewing resource connector job events.

LSF resource connector supports Amazon Web Services (AWS) EC2 Fleet

The LSF resource connector for Amazon Web Services (AWS) now uses an Amazon EC2 Fleet API to create groups (or a fleet) of instances. EC2 Fleet is an AWS feature that extends the existing spot fleet, which gives you a unique ability to create fleets of EC2 instances composed of a combination of EC2 on-demand, reserved, and spot instances, by using a single API. For configuration details, see Using Amazon EC2 Fleet.

IBM Cloud Gen 2 provider capacity error codes added to the LSB_RC_TEMPLATE_REQUEST_DELAY parameter in the lsf.conf file

The LSB_RC_TEMPLATE_REQUEST_DELAY parameter in the lsf.conf configuration file allows you to define the number of minutes that LSF waits before repeating a request for a template, if the **ebrokerd** daemon encounters provider errors in a previous request. The parameter has been enhanced for the VPC Gen 2 (IBM Cloud Gen 2) API, to include several IBM Cloud Gen 2 capacity error codes.

What's new in IBM Spectrum LSF Version 10.1 Fix Pack 13

The following topics summarize the new and changed behavior in LSF 10.1 Fix Pack 13.

Download this Fix Pack from IBM Fix Central. For more information, see Getting fixes from IBM Fix Central.

Release date: June 2022.

- **Job scheduling and execution**
  The following new features affect job scheduling and execution.
- **Resource connector enhancements**
  The following enhancements affect LSF resource connector.
- **Resource management**
  The following new features affect resource management and allocation.
- **Container support**
  The following new feature affects LSF support for containers.
- **Command output formatting**
  The following enhancements affect LSF command output formatting.
- **Other changes to IBM Spectrum LSF**
  The following changes affect other aspects of LSF behavior.

Job scheduling and execution

The following new features affect job scheduling and execution.

Kill jobs and record jobs as DONE

LSF now allows you to use the new `bkill -d` command option to kill jobs, then records the jobs as `DONE` after the jobs exit. The -d option only takes effect for started jobs that are in the `RUN`, `USUSP`, or `SSUSP` state. Otherwise, the option is ignored.

For example,

```
bkill -d 2983 2986
Job <2983> is being terminated
Job <2986> is being terminated
```

Use the -d option when working with remote clusters.

The -d option is ignored if used with the -r or -s options.

Related tasks

- Kill and record jobs as DONE

Related reference

- **bkill** -d command option

Kill jobs by status

LSF now allows you to use the new `bkill -stat` command option to kill large number of jobs with the specified status as soon as possible.

The new option kills all applicable jobs and silently skips the jobs that LSF cannot kill. LSF kills local pending jobs immediately and cleans up the jobs as soon as possible, ignoring the time interval that is specified by CLEAN_PERIOD in lsb.params. Jobs that are killed in this manner are not logged to the lsb.acct file. LSF kills other jobs, such as running jobs, as soon as possible and cleans up these jobs normally (after the CLEAN_PERIOD time interval).

When running the `bkill -stat` command option, you do not need the job ID, nor do you need one of the -m, -q, -J, -g, -sla, or -app options.

The `bkill -stat run` command option kills all running jobs that you can kill.

The `bkill -stat pend` command option only works with three signals that are specified by the -s option: `INT`, `KILL`, or `TERM`.

The -stat option cannot be used with the -b option.

Related tasks

- Kill jobs by status

Related reference

- **bkill** -stat command option

Job count based fair share scheduling

LSF can now use the number of jobs in the fair share scheduling algorithm instead of the number of job slots.

You can potentially use job count based fair share scheduling if you set up a queue to run one job per host. In this situation, a job can use the entire job slots, memory, or other resources that are available in the host. Therefore, only the number of jobs that a user is running is under consideration for the fair share scheduling algorithm instead of job slots because each job can use all the job slots on a host.

To enable job count based fairs hare scheduling, define `FAIRSHARE_JOB_COUNT=Y` in the lsb.params file. If this parameter is set to `Y` or `y`, LSF uses the number of jobs in the fair share scheduling algorithm instead of job slots. This setting only affects the value that is multiplied by the job slots weighting factor in the fair share algorithm (RUN_JOB_FACTOR parameter in the lsb.params file).

Related concepts

- Job count based fair share scheduling

Related reference

- FAIRSHARE_JOB_COUNT parameter in the lsb.params file.

Delete job groups using idle times

LSF now allows you to delete job groups that reaches a specific idle time.

The new `bgdel -d idle_time` command option deletes job groups with idle times larger than the specified *idle_time*.

The command **bjgroup** now displays a new default output field for **bjgroup** that shows the idle time for each job group.

LSF now allows you to use the new parameter JOB_GROUP_IDLE_TTL to define the job group's time-to-live (TTL) when all jobs leave the job group. The **mbatchd** daemon removes the job group once the TTL reaches zero.

Related reference

- **bgdel** -d command option
- **bjgroup** IDLE_TIME
- JOB_GROUP_IDLE_TTL

Modify cgroup memory and swap limits for running jobs

LSF now supports modifying a running job's cgroup memory and swap limits. Continue to use **bmod -M** and **bmod -v** command to change these limits for a job. When you request to change a job's memory or swap limits, LSF modifies the job's requirement to accommodate your request, and now also modifies the job's cgroup limits setting for running jobs.

For details on LSF handles memory and swap limits, and configuration settings, and how LSF handles OS behavior for cgroup limit changes, see Memory and swap limit enforcement based on Linux cgroup memory subsystem.

Resource connector enhancements

The following enhancements affect LSF resource connector.

Automatic selection of the cheapest AWS template

Spot instances can provide a significant saving compared to other instance types. You can use spot instances to bid on spare Amazon EC2 computing capacity. Since spot instances are often available at a discount compared to the pricing of On-Demand instances, you can significantly reduce the cost of running your applications, grow application's compute capacity and throughput for the same budget, and enable new types of cloud computing applications.

You can set a threshold on the price for when to use spot instances. However, when the spot price it too high and you specified the lowest price as your allocation strategy, starting in IBM Spectrum LSF Version 10.1 Fix Pack 13, LSF automatically redirects to the next available template. The next available template is based on template priority, which can be the next cheapest Spot or On-Demand instance when the marketed spot price

is higher than the bid price. To check whether you set the lowest price as your allocation strategy, verify the allocationStrategy parameter is set to lowestPrice in the awsprov_templates.json file. If another allocation strategy is set, such as diversified, LSF does not automatically switch to the cheapest template.

To view the current market price, run the command:

```
badmin rc view -c templates -p aws
```

An example of the output, when the template spot price is equal or greater than the market price, which enables the template:

```
aws
    Templates
        templateId: aws2
            maxNumber: 2
            spotPrice: 0.004000     >>> This is the spot price set that is higher
than the current market price.
            fleetRole: arn:aws:iam::700071821657:role/EC2-Spot-Fleet-role
            allocationStrategy: lowestPrice
            imageId: ami-0dbddce684d30c81d
            subnetId: subnet-0dfee843e19bfeb52
            vmType: t3a.micro
            keyName: ib19b07
            userData: pricing=spot;zone=us_west_2b
            marketSpotPrice: 0.003200      >>> This is the real current market
price, which is lower than the spot price.
            securityGroupIds: [ "sg-08f1a36be62fe02a4" ]
            ebsOptimized: FALSE
            priority: 5
            attributes
                mem: ["Numeric", "700"]
                ncpus: ["Numeric", "2"]
                zone: ["String", "us_west_2b"]
                awshost: ["Boolean", "1"]
                ncores: ["Numeric", "1"]
                type: ["String", "X86_64"]
                pricing: ["String", "spot"]
```

An example output, when the template spot price is less than market price, which disables the template:

```
aws
templateId: aws-vm-3
            spotPrice: 0.003800
            marketSpotPrice: 0.031500
        Template disabled as the spot bid price is lesser than the market spot
price
```

Starting in IBM Spectrum LSF Version 10.1 Fix Pack 13, LSF automatically disables the template as the spot bid price is lesser than the market spot price and redirects to the next available template. The next available template is based on template priority, which might be the next cheapest Spot or On-Demand instance template. This template is temporarily disabled until the market price drops or the price is manually increased in the template. To manually increase the spot price, specify a greater bid price for the spotPrice parameter in the awsprov_templates.json file.

Related tasks

- Configuring AWS Spot instances

Related reference

- awsprov_templates.json

Selecting templates for minimum number of servers

In the resource connector policy (policy_config.json) file, a new Optimizations category is available to configure a list of allocation rules entries that specify how many hosts from a certain template are worth considering over another template. For more information about configuring the new allocRules parameter and its attributes, see policy_config.json.

To view the policies and optimizations that are configured for your resource connector policy (policy_config.json) file, run the command:

```
badmin rc view -c policies
```

An example of the output:

```
Policies
    Name: Policy1
        Consumer
            rcAccount: ["all"]
            templateName: ["all"]
            provider: ["all"]
        MaxNumber: 100
        StepValue: 5:10
    Name: Policy2
        Consumer
            rcAccount: ["default", "project1"]
            templateName: ["aws_template1"]
            provider: ["aws"]
        MaxNumber: 50
        StepValue: 5:20
Optimizations
        4 hosts (aws:aws_template1) replaced by 1 hosts (aws:aws_template3)
        2 hosts (aws:aws_template1) replaced by 1 hosts (aws:aws_template2)
```

In addition, consider the RC_DEMAND_POLICY parameter in the lsf.queues file contained the following example configuration:

```
RC_DEMAND_POLICY = THRESHOLD[[2,10] [4,5]]
```

This configuration sets optimization to first apply four hosts with `aws_template1` VMs with one host with `aws_template3` VM. Then, considers applying two hosts with `aws_template1` VMs with one hosts `aws_template2` VM. The RC_DEMAND_POLICY defined the buffer time for four jobs is shorter than two jobs, so that when the demand trigger by four or more jobs, then the first optimization rule is applied; otherwise, the second optimization rule is applied.

Related reference

- RC_DEMAND_POLICY
- policy_config.json

Resource management

The following new features affect resource management and allocation.

Global resources

LSF now supports global resources, which are resources that are shared between all connected clusters.

LSF handles global resources in the same way as local resources. For resource requirement strings, specify global resources for simple, compound, and alternative resource requirement strings in the same way as local resources. In addition, specify global resources for the `rusage`, `select`, `span`, `same`, `cu`, `order`, `duration`, and `decay` keywords in the resource requirement in the same way as local resources. Global resources can be static or dynamic resources, but must be numeric, consumable, and releasable.

LSF also supports global limits to use global resources because local limits in the cluster cannot use global resources. Global limits now support the compete distribution policy in addition to the evenly distribution policy.

Define global resources in the lsb.globalpolicies file. The Resource section specifies the global resource, the ResourceMap section defines the mapping between the global resources and their sharing clusters, the DistributePolicy sections define the distribution policies for the global resources and global limits, and the ReservationUsage section specifies the method of reserving global resources.

The **bgpinfo** command now displays information on global resources with the introduction of two subcommands: The **bgpinfo resource** subcommand shows usage information on global resources and the **bgpinfo policy** subcommand shows the distribution policies for global resources and global limits. The **bhosts -s** command option, which displays information about shared numeric resources on the hosts, also displays information on global resources, since global resources are numeric and consumable.

Related concepts

- Global resources

Related reference

- Resource section in the lsb.globalpolicies file
- ResourceMap section in the lsb.globalpolicies file
- DistributePolicy section in the lsb.globalpolicies file
- ReservationUsage section in the lsb.globalpolicies file
- Clusters section in the lsf.licensescheduler file
- **bgpinfo resource** and **bgpinfo policy** subcommands
- **bhosts -s** command option
- **blstat -cl** command option

Prevent bwait from running within a job

LSF now allows you to prevent the **bwait** command from running within a job. This prevents running jobs from blocking slots with the **bwait** command, since blocking slots leads to low slot utilization.

To prevent LSF from running the **bwait** command within a job, define `LSB_BWAIT_IN_JOBS=N` in the lsf.conf file. If this new parameter is not defined, the default value is `Y` (LSF can run the **bwait** from within a job).

If the **bwait** command is allowed to run within a job, LSF posts the wait condition as an external status message, which means that running a query on the job shows the wait condition.

Related concepts

- Improve slot utilization by preventing bwait from running in jobs

Related reference

- LSB_BWAIT_IN_JOBS parameter in the lsf.conf file
- **bwait** command

GPU resource allocation for resizable jobs

Leverage the new LSB_RESIZE_GPUS and LSB_ RESIZE_TIME environment variables for the resize notification command environment (when using resizable jobs). Additionally, depending on the GPU requirement for your resizable jobs, the behavior for resize requirements can differ. For details, see How resizable jobs work with other LSF features.

New allocated GPU information display in the **JOB_RESIZE_NOTIFY_START** record for **lsb.events**:

- **GPU_ALLOC_COMPAT (%s)**: This new string describes resized portions of the GPU allocation.
- **GPU_MEM_RSV (%s)**: This new string describes GPU memory that is reserved by resized tasks on execution hosts.

Related reference

- lsb.events

Default values for GPU parameter are changed

Starting in IBM Spectrum LSF Version 10.1 Fix Pack 13, the default values of the following three parameters are changed to:

```
LSF_GPU_AUTOCONFIG=Y
LSB_GPU_NEW_SYNTAX=extend
LSF_GPU_RESOURCE_IGNORE=Y
```

If you have fix pack 13 installed, no further action is needed to set these parameters. If you have IBM Spectrum LSF Version 10.1 Fix Pack 12 or earlier, consider explicitly configuring the same values to these three parameters.

If you want to keep the former GPU behavior, and any one of the three parameters are missing in lsf.conf file, you must explicitly configure the following default settings that are defined in fix pack 12 or earlier:

```
LSF_GPU_AUTOCONFIG=N
LSB_GPU_NEW_SYNTAX=N
LSF_GPU_RESOURCE_IGNORE=N
```

Related information

- lsf.conf

Container support

The following new feature affects LSF support for containers.

Podman version 3.3.1 support

LSF now supports the use of Pod Manager (Podman) 3.3.1. The Podman integration allows LSF to run jobs in Podman OCI containers on demand. For details, see with Podman.

Apptainer for running LSF jobs

Apptainer is the rebranded product name for Singularity. To accommodate this name change, LSF supports running LSF jobs in either named containers, accepting either singular[] or the new apptainer[] keyword as a value for the CONTAINER parameter configuration in LSF application profiles and batch queues. See lsb.applications and lsb.queues for container configuration details.

Mount a temporary directory in the container of the Docker job

In previous releases of IBM Spectrum LSF, the temporary (/tmp) directory is always mounted to the temporary directory of the host (/tmp) in the container of the Docker job.

Starting in IBM Spectrum LSF Version 10.1 Fix Pack 13, you can specify to either mount or not mount the temporary (/tmp) directory to the temporary directory of the host (/tmp) in the container of the Docker job. By default, the temporary directory is mounted. For more information, see the LSF_DOCKER_MOUNT_TMPDIR parameter in the lsf.conf file.

Related reference

- LSF_DOCKER_MOUNT_TMPDIR

Command output formatting

The following enhancements affect LSF command output formatting.

New limit and resource output fields for bqueues -o

LSF now has new limit and resource output fields that you can specify for the **bqueues -o** command output. The following table describes the new output fields:

Table 1. New limit and resource output fields for bqueues

Field name	Width	Aliases
max_corelimit	8	corelimit
max_cpulimit	30	cpulimit

Field name	Width	Aliases
default_cpulimit	30	def_cpulimit
max_datalimit	8	datalimit
default_datalimit	8	def_datalimit
max_filelimit	8	filelimit
max_memlimit	8	memlimit
default_memlimit	8	def_memlimit
max_processlimit	8	processlimit
default_processlimit	8	def_processlimit
max_runlimit	12	runlimit
default_runlimit	12	def_runlimit
max_stacklimit	8	stacklimit
max_swaplimit	8	swaplimit
max_tasklimit	6	tasklimit
min_tasklimit	6	
default_tasklimit	6	def_tasklimit
max_threadlimit	6	threadlimit
default_threadlimit	6	def_threadlimit
res_req	20	
hosts	50	

Note: The following resource limit field names are supported, but show the same content as their corresponding maximum resource limit fields (that is, the following resource limit field names are aliases): `corelimit`, `cpulimit`, `datalimit`, `filelimit`, `memlimit`, `processlimit`, `runlimit`, `stacklimit`, `swaplimit`, `tasklimit`, `threadlimit`.
For example, `corelimit` is the same as `max_corelimit`.

You can also specify these new fields in default output formatting of the **bqueues** command by specifying the LSB_BQUEUES_FORMAT parameter in the lsf.conf file, or by specifying the LSB_QUEUES_FORMAT environment variable.

Related concepts

- Customize queue information output

Related reference

- **bqueues -o** command option
- LSB_BQUEUES_FORMAT parameter in the lsf.conf file

Customize resource allocation limits information output using blimits -o

Like the **bjobs -o** and **bqueues -o** options, you can now also customize specific fields that the **blmits** command displays by using the -o command option. This allows you to create a specific output format that shows all the required information, which allows you to easily parse the information by using custom scripts or to display the information in a predefined format. For details see on the new -o option usage, interactions with existing **blimits** command options, and affected fields, see blimits.

Related concepts

- Display resource allocation limits
- View information about resource allocation limits

Related reference

- **blimits -a** command option

Memory and CPU usage details added to bjobs, bhist, and bacct commands

Added memory and CPU usage information, such as CPU efficiency, CPU peak usage, and memory efficiency values to the following list of commands:

- **bjobs -l**
- **bhist -l**
- **bacct -l**
- **bjobs -o cpu_peak**
- **bjobs -o cpu_efficiency**
- **bjobs -o mem_efficiency**

Use the output details, such as the CPU peak usage, to configure the new CPU_PEAK_SAMPLE_DURATION parameter (in the lsb.params configuration file) to define the duration of the sample CPU time for running jobs. For more information, see the CPU_PEAK_SAMPLE_DURATION reference topic.

Sample output of **bjobs -l** command:

```
Job <108>, User , Project , Status , Queue , Command <blaunch ../scripts/eat 50
100 2>, Share group charged

Thu Dec 9 04:49:03: Submitted from host , CWD </scratch/qa4/lsf_user/v1011/clu1>,
4 Task(s), Requested Resources <span[ptile=2]>;
Thu Dec 9 04:49:03: Started 4 Task(s) on Host(s) <2ib22b11> <2ib16b14>,
Allocated 4 Slot(s) on Host(s) <2ib22b11> <2ib16b14>,
Execution Home </home/lsf_user>, Execution CWD </scratch/qa4/lsf_user/v1011/clu1>;
Thu Dec 9 04:49:39: Resource usage collected.
The CPU time used is 71 seconds.
MEM: 115 Mbytes; SWAP: 0 Mbytes; NTHREAD: 6

MEMORY USAGE:
MAX MEM: 115 Mbytes; AVG MEM: 114 Mbytes; MEM EFFICIENCY: 100%

CPU USAGE:
CPU EFFICIENCY: 98%; CPU PEAK USAGE: 2

SCHEDULING PARAMETERS:
r15s r1m r15m ut pg io ls it tmp swp mem
loadSched - - - - - - - - - - -
loadStop - - - - - - - - - - -
```

Sample output of **bhist -l** command:

```
Thu Dec 9 04:49:03: Starting (Pid 140972);
```

```
Thu Dec 9 04:49:03: Running with execution home </home/lsf_user>, Execution CWD
</scratch/qa4/lsf_user/v1011/clu1>, Execution Pid <140972>;
Thu Dec 9 04:51:04: Done successfully. The CPU time used is 480.0 seconds;
HOST: ib22b11; CPU_TIME: 240 seconds
HOST: ib16b14; CPU_TIME: 240 seconds
Thu Dec 9 04:51:05: Post job process done successfully;

MEMORY USAGE:
MAX MEM: 215 Mbytes; AVG MEM: 163 Mbytes; MEM EFFICIENCY: 100%

CPU USAGE:
CPU EFFICIENCY: 98%; CPU PEAK USAGE: 2

Summary of time in seconds spent in various states by Thu Dec 9 04:51:05
PEND PSUSP RUN USUSP SSUSP UNKWN TOTAL
0 0 121 0 0 0 121
```

Sample output of **bacct -l** command:

```
Accounting information about this job:
Share group charged
CPU_T WAIT TURNAROUND STATUS HOG_FACTOR MEM SWAP
480.00 0 121 done 3.9669 215M 0M
CPU_PEAK CPU_EFFICIENCY MEM_EFFICIENCY
2 98% 100%

Host based accounting information about this job:
HOST CPU_T MEM SWAP
ib22b11 240.00 115M 0M
ib16b14 240.00 100M 0M
```

Sample output of **bjobs -o cpu_peak** command:

```
CPU_PEAK
0.00
```

Sample output of **bjobs -o cpu_efficiency** command:

```
CPU_EFFICIENCY
0.00%
```

Sample output of **bjobs -o mem_efficiency** command:

```
MEM_EFFICIENCY
0.00%
```

Related reference

- **bjobs -l** command
- **bhist** command
- **bacct** command
- **bjobs -o** command
- CPU_PEAK_SAMPLE_DURATION parameter

Other changes to IBM Spectrum LSF

The following changes affect other aspects of LSF behavior.

Updated support for hardware locality (hwloc) library

LSF now uses version 2.6 of the hardware locality (**hwloc**) library.
After upgrading to **hwloc** 2.6, for hosts with one NUMA node per socket, the hardware topology layer changes from NUMA > socket > core > thread to socket > NUMA > core > thread. You need to adjust your job submission patterns if your jobs assume that the NUMA layer is always above the socket layer.

The **lim -T** and **lshosts -T** commands now display the physical index for the socket and core. This display change is consistent with the existing output for the **bhosts -aff** command.

Specify the number of MultiCluster starting jobs and tasks that can be configured at the receive-jobs queue

You can now specify how many MultiCluster jobs or tasks, from remote clusters, that can be configured at the receive-jobs queue by setting two new parameters, IMPT_JOBLIMIT and IMPT_TASKLIMIT, in the lsb.queues configuration file.

Honoring the preferred host for host group members

You can now specify this preferred host for dispatching a job to a certain host group. Set this value in the GROUP_MEMBER section of the lsb.hosts configuration file.

Host groups support for commands that support host names

Existing LSF commands that support a host name parameter, now also support host groups. See the affected commands: **battr**, **bresume**, **brvs**, **lshosts**, and **lsload**, for details on the additional parameter. Additionally, to use the host group parameter with the **lshosts** and **lsload** commands, you must first configure the new LSF_HOSTGROUP_INFO=Y setting in the lsf.conf file.

New platform support

LSF supports the following platforms (see Operating system support for details):

- RHEL 8.5 and 8.6 on x64 and Power, kernel 4.18.0, glibc 2.28
- RHEL 9.0 on x64 and Power, kernel 5.14.0, glibc 2.34
- RHEL 8.*x*, RHEL 9.0, and IBM AIX 7.*x* on IBM Power 10

What's new in IBM Spectrum LSF Version 10.1 Fix Pack 12

The following topics summarize the new and changed behavior in LSF 10.1 Fix Pack 12.

Release date: 17 August 2021.

- **Security**
  The following new features affect cluster security.
- **Resource connector enhancements**
  The following enhancements affect LSF resource connector.
- **GPU enhancements**
  The following enhancements affect LSF GPU support.
- **Deprecated features on IBM Spectrum LSF**
  The following features are deprecated in LSF 10.1 Fix Pack 12, and might be removed in a future version of LSF.
- **Other changes to IBM Spectrum LSF**
  The following changes affect other aspects of LSF behavior.

Security

The following new features affect cluster security.

New administration subcommand to check the LSF security configuration

LSF now has a new **badmin security view** subcommand to check the current configuration of the LSF security mechanism.

The **badmin security view** command displays a summary of the current configuration.

The **badmin security view -v** command option provides a detailed description of the current configuration and displays any changes that you need to make to the configuration to secure your cluster.

Related concepts

- View the cluster security settings

Related reference

- **badmin** command

Using the previous user authentication key

LSF now allows you to continue using the previous **eauth** key for encrypting and decrypting user authentication data. After defining a new **eauth** key, this gives LSF administrators time to update the **eauth** key on each host in the cluster without disrupting authentication operations.

To continue using the old **eauth** key when defining a new key, rename the current LSF_EAUTH_KEY parameter in the lsf.sudoers file to LSF_EAUTH_OLDKEY, then define the LSF_EAUTH_OLDKEY_EXPIRY parameter to specify an expiry date for the old key. Define a new LSF_EAUTH_KEY parameter with the new **eauth** key as the value. After the expiry date, the old key no longer works and only the new LSF_EAUTH_KEY parameter works.

The date is in the form of [*year–month–day*] where the number ranges are as follows: year after 1970, month 1-12, day 1-31.

To enable the previous key, you must define both LSF_EAUTH_OLDKEY and LSF_EAUTH_OLDKEY_EXPIRY in the lsf.sudoers file.

Related reference

- Configuration to modify external authentication
- lsf.sudoers file

Root privileges for LSF Application Center, LSF Explorer, LSF Process Manager, and LSF RTM

LSF now restricts root privileges on all hosts for LSF Application Center, LSF Explorer, LSF Process Manager, and LSF RTM by default.

If you are using LSF Application Center, LSF Explorer, LSF Process Manager, or LSF RTM, you must specify a space-separated list of hosts in the LSF_ADDON_HOSTS parameter in the lsf.conf file. This allows the root users on these specified hosts to remotely execute commands. You must also set `LSF_DISABLE_LSRUN=N` in the lsf.conf file to enable hosts that are running LSF Application Center to use the **lsrun** and **lsgrun** commands.

Related reference

- LSF_ADDON_HOSTS parameter in the lsf.conf file

Resource connector enhancements

The following enhancements affect LSF resource connector.

Support for additional Google Cloud Platform features

The LSF resource connector for Google Cloud Platform now includes support for the following Google Cloud Platform features:

Launch instance templates

You can now specify the launchTemplateId attribute to enable launch instance templates. You need to create the specified instance template in Google Cloud before using it. When using launch instance templates, you can define all the instance's properties within the template when you create it, then you only need to specify the zone or region in the googleprov_templates.json file. The same attributes that are specified in the googleprov_templates.json file override the values that are specified in the template. For more information on the override behavior of instance templates, see the Google Cloud documentation: https://cloud.google.com/compute/docs/instances/create-vm-from-instance-template#creating_a_vm_instance_from_an_instance_template_with_overrides.
Note: To export environment variables to the instances, you still must specify the userData attribute in the googleprov_templates.json file. To add labels to the instances, you still must specify the instanceTags attribute in the googleprov_templates.json file.
For more information on launch instance templates, see the Google Cloud documentation:https://cloud.google.com/compute/docs/instance-templates

Local SSDs

LSF now supports attached local SSDs through launch instance templates, but does not include an interface in the googleprov_templates.json file. In Google Cloud, attach the local SSD to the launch instance template in Disks > Add new disk by selecting Local SSD scratch disk in the Type field.
You must mount SSDs before using them. LSF includes an example code that illustrates how to mount multiple local SSDs in one logical volume in the <LSF_TOP>/<LSF_VERSION>/resource_connector/google/scripts/example_user_data.sh file.

For more information on local SSDs, see the Google Cloud documentation: https://cloud.google.com/compute/docs/disks/local-ssd.

Preemptible VM instances

Preemptible VM instances are instances that run at a lower cost than standard instances, with most of the same features as standard instances.
LSF now supports preemptible VM instances through launch instance templates, but does not include an interface in the googleprov_templates.json file. In Google Cloud, set the Preemptibility field to On when creating the launch instance template to enable preemptible VM instances.

When a VM instance is being preempted, the instance transitions into `TERMINATED` status and LSF automatically requeues the job that is running on the instance. LSF then deletes the preempted instance.

For more information on preemptible VM instances, see the Google Cloud documentation: https://cloud.google.com/compute/docs/instances/preemptible.

Bulk instance APIs (Bulk API endpoints)

LSF resource connector now automatically uses bulk API endpoints to create Google Cloud instances. Google Cloud Platform provides both zonal bulk API endpoint (**Instances.BulkInsert**) and regional bulk API endpoint (**RegionInstances.BulkInsert**) support.
If the zone in which you want to create your instances is not important, configure LSF resource connector to call the regional bulk API endpoint by specifying a value for the GCLOUD_REGION parameter in the googleprovconfig.json file or by defining a region in the googleprov_template.json file. The region that is defined in the googleprov_templates.json file overrides the region that is defined in the GCLOUD_REGION parameter. Google Cloud Platform automatically selects the zone in which to create your instances, considering the available hardware capacity in each zone.

If you want to specify a zone in which to create your instances, define the zone in the googleprov_templates.json file, and LSF resource connector calls the zonal bulk API endpoint.

Related tasks

- Configuring Google Cloud Platform for resource connector

Related reference

- googleprov_config.json
- googleprov_templates.json

GPU enhancements

The following enhancements affect LSF GPU support.

NVIDIA Data Center GPU Manager (DCGM) integration updates

LSF now integrates with Version 2.3.1 of the NVIDIA Data Center GPU Manager (DCGM) API.

Enable the DCGM integration by defining the LSF_DCGM_PORT parameter in the lsf.conf file.

Related reference

- LSF_DCGM_PORT parameter in the lsf.conf file

Support for Nvidia Multi-Instance GPU (MIG)

LSF now supports Nvidia Multi-Instance GPU (MIG) devices, such as the Nvidia A100 GPU. MIG allows a single supported GPU to be securely partitioned into up to seven independent GPU instances, providing multiple users with independent GPU resources.

Specify MIG device requirements in the GPU requirements strings (**bsub -gpu** option or the GPU_REQ parameter in the lsb.queues and lsb.applications files) or resource requirements strings (**bsub -R "rusage[]"** option GPU_REQ parameter in the lsb.queues and lsb.applications files) by using the new mig keyword. Specify the requested number of GPU instances for the MIG job. Valid GPU instance sizes are 1, 2, 3, 4, 7. Optionally, specify the requested number of compute instances after the specified GPU instance size and a slash character (`/`). The requested compute instance size must be less than or equal to the requested GPU instance size. In addition, Nvidia MIG does not support the following GPU/compute instance size combinations: 4/3, 7/5, 7/6. If this is not specified, the default compute instance size is the same as the GPU instance size. For example, the following job uses 2 Nvidia MIG devices with a GPU instance size of 3 and a compute instance size of 2:

```
bsub -gpu "num=2:mig=3/2" ./app
```

The new LSF_MANAGE_MIG parameter in the lsf.conf file enables dynamic MIG scheduling.

- If set to `Y` or `y`, LSF dynamically creates GPU instances (GI) and compute instances (CI) on each host, and LSF controls the MIG of each host. If you enable dynamic MIC scheduling, do not manually create or destroy MIG devices outside of LSF.
- If set to `N` or `n`, LSF uses static MIG scheduling. LSF allocates the GI and CI based on the configuration of each MIG host, and dispatches jobs to the MIG hosts. LSF does not create or destroy the GI and CI on the MIG hosts. If you use static MIG scheduling and want to change MIG devices, you must wait for the running MIG job to finish, then destroy the existing MIG device, create a new MIG device, and restart the LSF daemons.

The **lshosts -gpu** command option now shows whether the GPU supports MIG functions. To view detailed information on MIG instances, use the new -mig option together with the **lshosts -gpu** command option.

The **bhosts -gpu -l** and **bjobs -l** command options show additional MIG device information. The **bhosts -o** command option and the LSB_BHOSTS_FORMAT parameter in the lsf.conf file also has a new mig_alloc keyword to display the MIG allocation information in the **bhosts** customized output.

Related concepts

- Submitting and monitoring GPU jobs

Related reference

- GPU_REQ parameter in the lsb.applications file.
- GPU_REQ parameter in the lsb.queues file.
- **bsub -gpu** command option
- **bhosts -gpu** command option
- **lshosts -gpu** command option

Deprecated features on IBM Spectrum LSF

The following features are deprecated in LSF 10.1 Fix Pack 12, and might be removed in a future version of LSF.

Table 1. Deprecated features in LSF 10.1 Fix Pack 12

Deprecated feature	Corresponding deprecated parameters and commands	Alternative feature

LSF multicluster capability resource leasing model (See Platform LSF resource leasing model).	• lsb.resources file:
	  ◦ HostExport section
	  ◦ SharedResourceExport section
	• lsb.queues file:
	  ◦ HOSTS: `allremote` and `all@`*`cluster_name`* keywords
		Updated LSF default behavior.

LSF/XL feature and LSF Advanced Edition.	• **-cname** option for the following commands: **bacct**, **bhosts**, **bjobs**, **bmgroup**, **lshosts**, **lsload**.	Updated LSF default behavior.

Job slot pools for fair share scheduling	• lsb.queues file:
	  ◦ MAX_SLOTS_IN_POOL
	  ◦ SLOT_POOL
	  ◦ SLOT_SHARE
	  ◦ USE_PRIORITY_IN_POOL
		Guarantee SLA (guarantee service class).

Deprecated feature	Corresponding deprecated parameters and commands	Alternative feature

toplib integrations	• lsf.conf file:
	  ○ LSF_TOPD_PORT
	  ○ LSF_TOPD_TIMEOUT	

Chunk job scheduling	• lsf.conf file:
	  ○ LSB_CHUNK_RUSAGE
	• lsb.params file:
	  ○ CHUNK_JOB_DURATION
	• lsb.applications and lsb.queues files:
	  ○ CHUNK_JOB_SIZE
		Use the RELAX_JOB_DISPATCH_ORDER parameter in the lsb.params file to enable multiple jobs with common resource requirements to run consecutively on the same allocation.

HPC integrations for the following environments:
• Parallel job support using PAM
• IBM Parallel Environment (IBM PE) (See Running MPI workload through IBM Parallel Environment Runtime Edition)
• SGI CPUSET
	• lsf.conf file:
	  ○ LSF_PAM_APPL_CHKPNT
	  ○ LSF_PAM_CLEAN_JOB_DELAY
	  ○ LSF_PAM_HOSTLIST_USE
	  ○ LSF_PAM_PLUGINDIR
	  ○ LSF_PAM_USE_ASH
	  ○ LSF_PE_NETWORK_NUM
	  ○ LSF_PE_NETWORK_UPDATE_INTERVAL
	  ○ LSB_CPUSET_BESTCPUS
	  ○ LSB_CPUSET_DISPLAY_CPULIST
	  ○ LSF_CPUSETLIB
	• lsb.applications file:
		Use the **blaunch** command for parallel jobs.

Deprecated feature	Corresponding deprecated parameters and commands	Alternative feature
	◦ NETWORK_REQ • lsb.params file: ◦ MAX_PROTOCOL_INSTANCES ◦ NETWORK_REQ ◦ STRIPING_WITH_MINIMUM_NETWORK • lsb.queues file: ◦ MAX_PROTOCOL_INSTANCES ◦ NETWORK_REQ ◦ STRIPING_WITH_MINIMUM_NETWORK • **bsub -network** command option	
Automatic CPU frequency selection in energy aware scheduling (See Automatic CPU frequency selection)		
Relaxed syntax for resource requirement selection strings	• lsf.conf file: ◦ LSF_STRICT_RESREQ: Will be fixed to `Y`.	Updated LSF default behavior (strict syntax for resource requirement selection strings).
Task list files and related commands	• lsf.task file • lsf.conf file: ◦ LSF_SHELL_AT_USERS • **ch** command • **lsfmon** command • **lslogin** command • **lsmon** command • **lstcsh** command • **lsltasks** command • **lsrtasks** command • **lseligible** command	

Deprecated feature	Corresponding deprecated parameters and commands	Alternative feature

SLA scheduling except guarantee SLA (See Using goal-oriented SLA scheduling).	• lsb.params file:
	  ○ ENABLE_DEFAULT_EGO_SLA
	• lsb.serviceclasses file:
	  ○ GOALS: `THROUGHPUT`, `VELOCITY`, and `DEADLINE` keywords.	

Slot and package guarantee policy	• lsb.params file: ○ SIMPLIFIED_GUARANTEE: Will be fixed to `Y`.	Guarantee SLA (guarantee service class), which will be the default behavior.

GPU scheduling and features that use ELIM	• lsf.conf file:
	  ○ LSB_GPU_NEW_SYNTAX: Will be fixed to `extend`.
	  ○ LSF_GPU_RESOURCE_IGNORE: Will be fixed to `Y`.
	  ○ LSF_GPU_AUTOCONFIG: Will be fixed to `Y`.
	• All **elim.gpu.*** ELIMS.
		Updated LSF default behavior (GPU autoconfiguration).

Other changes to IBM Spectrum LSF

The following changes affect other aspects of LSF behavior.

New platform support

LSF supports the following platforms:

- RHEL 8.4, kernel 4.18.0, glibc 2.28
- SLES 15.3, kernel 5.3.18, glibc 2.26

What's new in IBM Spectrum LSF Version 10.1 Fix Pack 11

The following topics summarize the new and changed behavior in LSF 10.1 Fix Pack 11.

Release date: 21 December 2020

- **Terminology changes to IBM Spectrum LSF**
  LSF has the following changes to terminology.
- **GPU enhancements**
  The following enhancements affect LSF GPU support.
- **Job scheduling and execution**
  The following new features affect job scheduling and execution.
- **Resource connector enhancements**
  The following enhancements affect LSF resource connector.
- **Resource management**
  The following new features affect resource management and allocation.
- **Security**
  The following new features affect cluster security.
- **Container support**
  The following new feature affects LSF support for containers.
- **Command output formatting**
  The following enhancements affect LSF command output formatting.
- **Other changes to IBM Spectrum LSF**
  The following changes affect other aspects of LSF behavior.

Terminology changes to IBM Spectrum LSF

LSF has the following changes to terminology.

LSF now has the following terminology changes to the following components:

Table 1. Terminology changes

Old terminology	New terminology
master host	management host
master candidate host	management candidate host
master batch daemon (**mbatchd**)	management batch daemon
slave batch daemon (**sbatchd**)	server batch daemon
master LIM	management host LIM
slave LIM	server host LIM
master ELIM (MELIM)	management ELIM
master queue	parent queue
slave queue	child queue

Old terminology	New terminology
master **esub** (**mesub**)	management **esub**
slave.config file	server.config file

Some of these terminology changes are not reflected in command or script output.

GPU enhancements

The following enhancements affect LSF GPU support.

Improved performance for GPU resource collection

LSF has performance improvements for GPU metric collection by changing how the management host LIM and server host LIMs collect GPU resource information. These improvements include enabling the **mbatchd** daemon to no longer get GPU resource information from the management host LIM, and by completely removing the built-in GPU resources (`gpu_<num>n`) from the management host LIM.

To enable **mbatchd** to no longer get GPU resources from the management host LIM, set the LSF_GPU_RESOURCE_IGNORE parameter to `Y` in the lsf.conf file. This improves LSF response time because there are fewer LSF resources to manage and display.

In addition, if LSF_GPU_AUTOCONFIG is set to `Y` and LSB_GPU_NEW_SYNTAX is set to `Y` or `extend`, all built-in GPU resources (`gpu_<num>n`) are completely removed from the management host LIM. LSF uses a different method for the management host LIM and server host LIMs to collect GPU information. This further improves performance by having fewer built-in LSF resources.

The LSF_GPU_RESOURCE_IGNORE, LSF_GPU_AUTOCONFIG, and LSB_GPU_NEW_SYNTAX are all preexisting parameters in the lsf.conf file. To fully improve the performance of GPU resource collection, enable these parameters to completely remove the built-in GPU resources from the management host LIM:

```
LSF_GPU_AUTOCONFIG=Y
LSB_GPU_NEW_SYNTAX=extend
LSF_GPU_RESOURCE_IGNORE=Y
```

Note: If you are using LSF RTM, make sure that you are running LSF RTM, Version 10.2 Fix Pack 11, or later. If you cannot update LSF RTM to at least this version, do not set LSF_GPU_RESOURCE_IGNORE to `Y`, otherwise LSF RTM will not show host GPU information. This is because LSF RTM uses LSF resources to get host GPU information.

Related concepts

- Optimizing GPU resource metric collection

Related reference

- LSB_GPU_NEW_SYNTAX parameter in the lsf.conf file
- LSF_GPU_AUTOCONFIG parameter in the lsf.conf file
- LSF_GPU_RESOURCE_IGNORE parameter in the lsf.conf file

Displaying power utilization for each GPU

You can now enable the **lsload -gpuload** command to display the power utilization per GPU on the host.

The **lsload -gpuload** command displays the power utilization for each GPU in the new `gpu_power` column.

Related reference

- **lsload -gpuload** command option

Changes to Nvidia GPU resource requirements

LSF now has changes to the selection of Nvidia GPU resources in the GPU resource requirement string (**bsub -gpu** option or the GPU_REQ parameter in the lsb.queues and lsb.applications files).

The new gvendor keyword in the GPU requirements strings enables LSF to allocate GPUs with the specified vendor type. Specify `gvendor=nvidia` to request Nvidia GPUs, which is the default value.

The nvlink=yes keyword in the GPU requirements string is deprecated. Replace `nvlink=yes` in the GPU requirements string with `glink=yes` instead.

The new glink keyword in the GPU requirements strings specifies enables job enforcement for special connections among GPUs. If you specify `glink=yes` when using Nvidia GPUs, LSF must allocate GPUs that have the NVLink connection. By default, LSF can allocate GPUs that do not have special connections if there are an insufficient number of GPUs with these connections. Do not use glink with the nvlink keyword, which is now deprecated.

In addition, the **lshosts -gpu** and **bhosts -gpu** command options now show the GPU vendor type (AMD or Nvidia).

Related concepts

- Submitting and monitoring GPU jobs

Related reference

- GPU_REQ parameter in the lsb.applications file.
- GPU_REQ parameter in the lsb.queues file.
- **bsub -gpu** command option
- **bhosts -gpu** command option
- **lshosts -gpu** command option

Support for AMD GPUs

LSF now supports the use of AMD GPUs.

You can now specify AMD GPU models in the GPU requirements strings (**bsub -gpu** option or the GPU_REQ parameter in the lsb.queues and lsb.applications files).

The new gvendor keyword in the GPU requirements strings enables LSF to allocate GPUs with the specified vendor type. Specify `gvendor=amd` to request AMD GPUs. If this is not specified, the default is to request Nvidia GPUs.

The new glink keyword in the GPU requirements strings specifies enables job enforcement for special connections among GPUs. If you specify `glink=yes` when using AMD GPUs, LSF must allocate GPUs that have the xGMI connection. By default, LSF can allocate GPUs that do not have special connections if there are an insufficient number of GPUs with these connections. Do not use glink with the nvlink keyword, which is now deprecated.

In addition, the **lshosts -gpu** and **bhosts -gpu** command options now show the GPU vendor type (AMD or Nvidia).

To enable GPU requirements strings and use AMD GPUs in these requirements strings, `LSF_GPU_AUTOCONFIG=Y`, `LSB_GPU_NEW_SYNTAX=extend`, and `LSF_GPU_RESOURCE_IGNORE=Y` must be defined in the lsf.conf file.

Related concepts

- Submitting and monitoring GPU jobs

Related reference

- GPU_REQ parameter in the lsb.applications file.
- GPU_REQ parameter in the lsb.queues file.
- **bsub -gpu** command option
- **bhosts -gpu** command option
- **lshosts -gpu** command option

Improved GPU preemption

LSF Version 10.1 Fix Pack 7 introduced preemptive scheduling for GPU jobs so that a lower priority GPU job can release GPU resources for higher priority GPU jobs, with certain restrictions on the types of jobs that are involved in preemptive scheduling. LSF Version 10.1 Fix Pack 11 now introduces improvements to GPU preemption that removes or relaxes several of these previous restrictions on GPU jobs, including the following:

- Non-GPU jobs can now preempt lower priority GPU jobs.
- GPU jobs no longer have to be configured for automatic job migration and rerun to be preemptable by higher priority jobs. That is, the MIG parameter no longer has to be defined and the RERUNNABLE parameter no longer has to be set to `yes` in the lsb.queues or lsb.applications file. Ensure that you properly configure the MIG, RERUNNABLE, or REQUEUE parameters to ensure that GPU resources are properly released after the job is preempted.
- GPU jobs no longer have to have either `mode=exclusive_process` or `j_exclusive=yes` set to be preempted by other GPU jobs. GPU jobs can also use `mode=shared` if the GPU is used by only one shared-mode job.
  Higher priority GPU jobs cannot preempt shared-mode GPU jobs if there are multiple jobs running on the GPU.

Previously, to enable GPU preemption, you define the LSB_GPU_NEW_SYNTAX parameter in the lsf.conf file as either `Y` or `extend`, then configure the PREEMPTABLE_RESOURCES parameter in the lsb.params file to include the `ngpus_physical` resource. LSF then treats the GPU resources the same as other preemptable resources.

To enable the improved GPU preemption features introduced in LSF Version 10.1 Fix Pack 11, you must define the LSB_GPU_NEW_SYNTAX parameter in the lsf.conf file as `extend` (not as `Y`), then configure the

PREEMPTABLE_RESOURCES parameter in the lsb.params file to include the `ngpus_physical` resource. If you define the LSB_GPU_NEW_SYNTAX parameter in the lsf.conf file as `Y` instead of `extend`, GPU job preemption is enabled without these improvements and still has the restrictions from LSF Version 10.1 Fix Pack 7.

Related concepts

- Preemptive scheduling
- Enabling GPU features

Related reference

- LSB_GPU_NEW_SYNTAX parameter in the lsf.conf file.

Job scheduling and execution

The following new features affect job scheduling and execution.

Submit jobs as other users

LSF now allows a job submission user to submit jobs as another job execution user. This is useful if you want the job submission user to be a particular user, but to map the job execution user to other users.

To enable this feature, you must download and deploy the **bsubmit** executable file, which is a wrapper for the **bsub** command. For more details, refer to the following link: https://github.com/IBMSpectrumComputing/lsf-utils/tree/master/bsubmit

To define the execution users to which you can map submission users, create a configuration file named lsf.usermapping in the $LSF_ENVDIR directory to define the user mapping policy for the **bsubmit** command. The lsf.usermapping allows you to map several job execution users and user groups to a single submission user or user group. This file must be owned by the LSF administrator, with file permissions set to read/write for the owner and read-only for all other users.

For example,

```
#Submission user or group     # Execution users or groups
userA                         userB,userC,userD
groupA                        groupB
```

This lsf.usermapping configuration file means that the **userA** user can submit jobs as **userB**, **userC**, or **userD**. Users in the **groupA** group can submit jobs as any user in the **groupB** user group.

To submit jobs as other users, use the new bsubmit command. For example, run the following command if the job submission user **userA** is submitting a job as job execution user **userC**:

```
bsubmit --user userC myjob
```

Related concepts

- Submit jobs as other users

Related reference

- lsf.usermapping file
- **bsubmit** command

Dynamically add hosts to existing advance reservations

LSF now allows you to dynamically add new hosts to existing advance reservations.

To dynamically add new hosts to an existing advance reservation, use the new -f command option in the **brsvmod addhost** command. LSF selects these hosts based on the resource requirements as specified in the -R or -m command options.

Related concepts

- Changing an advance reservation

Related reference

- **brsvmod addhost -f** command option

Resource connector enhancements

The following enhancements affect LSF resource connector.

Launch IBM Cloud Gen 2 instances

LSF clusters can launch instances from IBM Cloud Virtual Servers for VPC Gen 2 (IBM Cloud Gen 2) to satisfy pending workload. The instances join the LSF cluster. If instances become idle, LSF resource connector automatically deletes them. Configure Cloud VPC Gen 2 as a resource provider with the ibmcloudgen2_config.json and ibmcloudgen2_templates.json files.

Related concepts

- Configuring IBM Cloud Virtual Servers for VPC Gen 2 for resource connector

Related reference

- ibmcloudgen2_config.json
- ibmcloudgen2_templates.json

Configure Spot VMs for CycleCloud

You can now enable and configure the use of spot VMs in the LSF resource connector template for Microsoft Azure CycleCloud. Spot VMs are instances that the Azure infrastructure can evict whenever Azure needs to

reclaim the capacity. There is no guarantee that a spot VM is available when requesting a new instance, but spot VMs cost less than regular on-demand VMs.

To use spot VMs, you must be using Microsoft Azure CycleCloud, Version 8 or newer.

To enable spot VMs, define the new interruptible parameter as `true` in the cyclecloudprov_templates.json file. If you enabled spot VMs, you can also specify the maximum allowed price before Azure evicts the instance. Define the new maxPrice parameter to the maximum price, in USD/hour per VM, in the cyclecloudprov_templates.json file.

Note: If the value of maxPrice is set to a lower price than the current offering price, the VM request remains pending.

Related reference

- cyclecloudprov_templates.json file

Specify job level account names for LSF resource connector

You can now specify account names for LSF resource connector at the job level. These account names are tagged on hosts that are borrowed through LSF resource connector. Previously, you could only specify account names at the queue or application level with the RC_ACCOUNT parameter.

To set the project name as the default account name, enable `DEFAULT_RC_ACCOUNT_PER_PROJECT=Y` in the lsb.params file. This account name overrides the value of the RC_ACCOUNT parameter at the application and queue levels (lsb.applications and lsb.queues files).

You can allow users to assign a specific account name at the job level by enabling `ENABLE_RC_ACCOUNT_REQUEST_BY_USER=Y` in the lsb.params file. This allows users to use the `bsub -rcacct "`*`rc_account_name`*`"` command option to assign an account name. This account name overrides all other values of the account name, including the setting of the project name. This also allows users to use an **esub** script to set the LSB_SUB6_RC_ACCOUNT parameter to change the job level account name. This parameter value overrides the value of the DEFAULT_RC_ACCOUNT_PER_PROJECT parameter in the lsb.params file and **bsub** or **bmod -rcacct** command options.

View accounting statistics for jobs that are billed to the specified account name by running the **bacct -rcacct** command option, or for jobs with the account name that actually ran on an LSF resource connector host by running the **bacct -rcalloc** command option.

Related tasks

- Updating configuration for resource connector

Related reference

- **bsub -rcacct** command option
- **bacct -rcacct** and **-rcalloc** command options
- ENABLE_RC_ACCOUNT_REQUEST_BY_USER parameter in the lsb.params file
- DEFAULT_RC_ACCOUNT_PER_PROJECT parameter in the lsb.params file

Resource management

The following new features affect resource management and allocation.

Per-consumer global limits for job resource allocations

When using global limit scheduling in the LSF multicluster capability, you can now configure the global limits to apply to each individual consumer instead of to all consumers.

To define the global per-consumer resource allocation limits, edit the lsb.globalpolicies file and specify the PER_APP, PER_LIC_PROJECT, PER_PROJECT, PER_QUEUE, or PER_USER parameters in the Limit section. You cannot specify the per-consumer limit together with the consumer limit (for example, you cannot configure PER_APP and APPS limits in the same Limit section).

All other features of global limit scheduling remain the same.

To enable global limit scheduling among all clusters, edit the lsb.params file and define GLOBAL_LIMITS=Y.

Related tasks

- Global limits for job resource allocations

Related reference

- Limit section in the lsb.globalpolicies file

Monitor scheduler efficiency

LSF now provides a scheduling efficiency metric to help you with monitoring cluster performance.

When the amount of time that LSF spent scheduling a job is large compared to the run times of jobs, you will observe a low resource utilization in your cluster. For instance, if the average run time of jobs equals the average time required to fill a slot after a job finishes, the slot usage in the cluster will be approximately 50% of what it would be if scheduling overhead is zero. It is not always clear whether low utilization is caused by scheduling performance or by configured policies (such as limits) that block jobs from accessing resources.

LSF now provides a scheduling efficiency metric in the **badmin perfmon** command that estimates how the slot and memory utilization of the cluster is affected by scheduling overhead. A value near 100% means that improving scheduler performance does not significantly improve resource utilization, while a lower percentage indicates how improving scheduler performance will improve resource utilization. For example, a value of 75% means that due to scheduling overhead, resource utilization is only 75% of what it could be if scheduling overhead were to be reduced to zero.

Run the **badmin perfmon view** command to view the scheduler efficiency for finished jobs within a sample period. This displays the scheduler efficiency numbers for both the set of finished jobs within the sample period and all finished jobs in the cluster.

Run the **bacct** command to view the scheduler efficiency for all finished jobs in a cluster.

Related concepts

- Monitor scheduler efficiency and overhead

Related reference

- **bacct** command
- **badmin** command

Security

The following new features affect cluster security.

New administration command to start and stop LSF daemons

LSF now has a new, separate administration command, **bctrld**, to start and stop LSF daemons (LIM, RES, and **sbatchd**).

The **bctrld** executable file, which is owned by root, is installed with the setuid flag turned off and this command can be used only by root. To allow non-root users (such as LSF administrators) to run the **bctrld** command to start the LSF daemons, you must either add the non-root users to the sudo Linux group or enable the setuid flag and configure the non-root users in the lsf.sudoers file.

This command moves all the LSF daemon start, stop, and restart administration subcommands into one separate command. The old administration subcommands are now obsolete and will be deprecated in a future release. The **bctrld** command syntax is identical to the old administration subcommands. The following table lists the old administration subcommands that the **bctrld** replaces:

Table 1. New LSF administration commands

Old LSF administration subcommand	New bctrld subcommand
badmin hrestart	**bctrld restart sbd**
badmin hshutdown	**bctrld stop sbd**
badmin hstartup	**bctrld start sbd**
lsadmin limrestart	**bctrld restart lim**
lsadmin limshutdown	**bctrld stop lim**
lsadmin limstartup	**bctrld start lim**
lsadmin resrestart	**bctrld restart res**
lsadmin resshutdown	**bctrld stop res**
lsadmin resstartup	**bctrld start res**

The **bctrld** command syntax follows the general format of `bctrld action daemon_name options`.

Moving the LSF daemon start and stop actions to a separate command means that LSF administration commands no longer require root privileges. Therefore, the LSF installer no longer enables the setuid bit for the LSF administration commands (**badmin**, **lsadmin**, **egosh**, **utmpreg**, **swtbl_api**, **ntbl_api**, **lstbl_nid**, and **swtbl_poe**), so the LSF commands no longer run with root privileges. This increases the security of your cluster by preventing unauthorized use of root privileges.

Related concepts

- If you install as a non-root user

Related tasks

- Allowing administrators to start daemons with lsf.sudoers

Related reference

- **badmin** command
- **bctrld** command
- **lsadmin** command
- lsf.sudoers file

Restrict user access to the lsmake command

LSF can now restrict users from using the **lsmake** command to run **make** tasks on remote hosts.

To enable LSF to restrict users from using the **lsmake** command, set the existing parameter LSF_DISABLE_LSRUN parameter to `Y` in the lsf.conf file. This parameter enables the LSF RES process to refuse remote connections from the **lsrun** and **lsgrun** commands unless the user is either an LSF administrator or root, but this parameter now also enables RES to refuse remote connections from the **lsmake** command.

This increases cluster security by preventing unauthorized users from accessing remote hosts.

Related tasks

- Restrict user access to remote hosts

Related reference

- LSF_DISABLE_LSRUN parameter in the lsf.conf file

Kerberos user impersonation

LSF can now use Kerberos user impersonation to submit jobs as the authenticated Kerberos TGT user.

To enable Kerberos user impersonation, first enable user eauth with krb5, then set the new LSB_KRB_IMPERSONATE parameter to `Y` in the lsf.conf file. For the **blaunch** command to work when user eauth with krb5 is enabled, you must also set LSF_EAUTH_DATA_REUSE=N in the lsf.conf file.

Run the Kerberos command `kinit -r` *`user_name`* to obtain a user TGT for the submission user to impersonate. When submitting a job, LSF changes the OS submission user on the job execution host to this TGT user.

Note: For security purposes, using root is not supported for Kerberos. Do not use root and do not add the root user to Kerberos.
If Kerberos user impersonation is enabled, the following LSF commands work differently:

- If the token user is not the OS submission user, commands that depend on OS file permissions (such as **bpeek** and **brestart**) do not work properly.
- The ACL feature for the **bacct** and **bhist** commands is disabled to prevent other users from getting the LSF administrator token. To ensure that the commands remain secure, do not enable the setuid bit for the **bacct** and **bhist** executable files, and disable them if they are already set.
- The **lsrun** command might behave inconsistently between running on local and remote hosts, because when an **lsrun** task is run on the local host, it does not go through **eauth** authorization.

Related reference

- LSB_KRB_IMPERSONATE parameter in the lsf.conf file
- LSF_AUTH parameter in the lsf.conf file
- Configuration to enable Kerberos authentication
- Configuration to modify Kerberos authentication

Authentication of query commands

LSF can now use authentication for query commands when external authentication is enabled.

To enable query command authentication, set the new LSF_AUTH_QUERY_COMMANDS parameter to `Y` in the lsf.conf file. Use the LSF_AUTH parameter in the lsf.conf file to specify the external client to server authentication method that is used.

After changing the value of the LSF_AUTH_QUERY_COMMANDS parameter, you must restart the **mbatchd** and **gpolicyd** daemons for this parameter to take effect.

Note:

- Before enabling query command authentication in the LSF cluster, you must ensure that all hosts in the cluster (including management, server, and client hosts), including all LSF command executable files, are updated to LSF, Version 10.1 Fix Pack 11, or newer. If you have any commands that are built using the APIs from previous versions of LSF, you must also rebuild these commands with the APIs from this latest version of LSF.
- If you are enabling query command authentication for the LSF multicluster capability, ensure that all LSF clusters use the same LSF_EAUTH_KEY value in the lsb.sudoers file.

Related reference

- LSF_AUTH_QUERY_COMMANDS parameter in the lsf.conf file
- LSF_AUTH parameter in the lsf.conf file
- Configuration to enable Kerberos authentication
- Configuration to enable external authentication

Container support

The following new feature affects LSF support for containers.

Running LSF jobs in Podman containers

LSF now supports the use of Pod Manager (Podman). The Podman integration allows LSF to run jobs in Podman OCI containers on demand.

The Podman installation packages must be installed on all the LSF server hosts where you intend to run Podman container jobs. The minimum Podman version is Podman, Version 1.6.4 on a RHEL 8.2 host. For optimal performance, use Podman, Version 1.9.3, or newer, on a RHEL 8.2.1 host.

Important: You cannot use LSF to run Docker jobs if you are using LSF to run Podman jobs.

Related concepts

- with Podman

Related reference

- CONTAINER and EXEC_DRIVER parameters in the lsb.applications file
- CONTAINER and EXEC_DRIVER parameters in the lsb.queues file

Running LSF jobs in Enroot containers

LSF now supports the use of Enroot. The Enroot integration allows LSF to run jobs in Enroot containers on demand.

The enroot or enroot-hardened installation package, Version 3.10, or later must be installed and configured correctly on the LSF server hosts where you intend to run Enroot jobs.

Related concepts

- with Enroot

Related reference

- CONTAINER and EXEC_DRIVER parameters in the lsb.applications file
- CONTAINER and EXEC_DRIVER parameters in the lsb.queues file

Command output formatting

The following enhancements affect LSF command output formatting.

Specify a unit prefix for bjobs -o resource fields

You can now specify a unit prefix for the following resource fields in the **bjobs -o** command option or the LSB_BJOBS_FORMAT parameter in the lsf.conf file: `mem`, `max_mem`, `avg_mem`, `memlimit`, `swap`, `swaplimit`, `corelimit`, `stacklimit`, and `hrusage` (for `hrusage`, the unit prefix is for `mem` and `swap` resources only).

To specify a unit prefix, specify a second colon (`:`) with a unit in the customized output format string in the **bjobs -o** command option or the LSB_BJOBS_FORMAT parameter in the lsf.conf file. This unit is `KB` (or `K`) for

kilobytes, MB (or M) for megabytes, GB (or G) for gigabytes, TB (or T) for terabytes, PB (or P) for petabytes, EB (or E) for exabytes, ZB (or Z) for zettabytes), or S to automatically adjust the value to a suitable unit prefix and remove the "bytes" suffix from the unit. The default is automatically adjust the value to a suitable unit prefix, but keep the "bytes" suffix in the unit.

The display value keeps two decimals but rounds up the third decimal. For example, if the unit prefix is set to G, 10M displays as 0.01G.

You can also specify the unit prefix by defining the LSB_UNIT_FOR_JOBS_DISPLAY environment variable or the LSB_UNIT_FOR_JOBS_DISPLAY parameter in the lsf.conf file. The value of the **bjobs -o unit** keyword setting overrides the value of the LSB_UNIT_FOR_JOBS_DISPLAY environment variable, which also overrides the value of the LSB_UNIT_FOR_JOBS_DISPLAY parameter in the lsf.conf file.

In addition, the default width for these resource fields except hrusage are increased from 10 to 15. That is, the following output fields now have a default width that is increased from 10 to 15:mem, max_mem, avg_mem, memlimit, swap, swaplimit, corelimit, and stacklimit.

Related reference

- **bjobs -o** command option
- LSB_BJOBS_FORMAT parameter in the lsf.conf file
- LSB_UNIT_FOR_JOBS_DISPLAY parameter in the lsf.conf file

Other changes to IBM Spectrum LSF

The following changes affect other aspects of LSF behavior.

Support for cgroup v2

LSF uses Linux **cgroup v1** to manage job process and resource limit accounting. LSF can now use **cgroup v2**, which is the newest version of Linux **cgroup**, for these functions. LSF automatically detects which version of **cgroup** is running on each host.
Important: **cgroup v2** support requires the lsf10.1_lnx310-lib217 package, not the lsf10.1_linux2.6-glibc2.3 package.
Different LSF hosts in the cluster can use different versions of **cgroup** as long as each individual LSF host is only running one version of **cgroup**. If you have both versions of **cgroup** enabled in a host, you must disable one of the versions. For example, hostA can use **cgroup v1** and hostB can use **cgroup v2** as long as each host is only running one version of **cgroup**.

The LSF configuration to enable **cgroup v2** accounting is the same as **cgroup v1**. That is, edit the lsf.conf file to enable LSF_PROCESS_TRACKING=Y and LSF_LINUX_CGROUP_ACCT=Y, then define the LSB_RESOURCE_ENFORCE parameter to specify the resources to enforce.

Related concepts

- Affinity binding based on Linux cgroup cpuset subsystem
- Job control actions
- Memory and swap limit enforcement based on Linux cgroup memory subsystem

Related reference

- LSB_RESOURCE_ENFORCE parameter in the lsf.conf file
- LSF_LINUX_CGROUP_ACCT parameter in the lsf.conf file
- LSF_PROCESS_TRACKING parameter in the lsf.conf file

Call back function for external scheduler plugin

The external scheduler plugin (**extsched** API) LSF can now register a job event callback function so that it can be notified when a job is created or destroyed within the scheduler context.

The **extsched** API now includes job callback functions for job events.

Related concepts

- API reference

New platform support

LSF supports the following platforms:

- RHEL 7.9, kernel 3.10, glibc 2.17
- RHEL 8.3, kernel 4.18.0, glibc 2.28
- SLES 15.2, kernel 5.3.18, glibc 2.26
- Ubuntu 20.04 LTS, kernel 5.4, glibc 2.31

What's new in IBM Spectrum LSF Version 10.1 Fix Pack 10

The following topics summarize the new and changed behavior in LSF 10.1 Fix Pack 10.

Release date: 16 June 2020

- **GPU enhancements**
  The following enhancements affect LSF GPU support.
- **Job scheduling and execution**
  The following new features affect job scheduling and execution.
- **Container support**
  The following new feature affects LSF support for containers.
- **Command output formatting**
  The following enhancements affect LSF command output formatting.
- **Performance enhancements**
  The following enhancements affect performance.
- **Resource connector enhancements**
  The following enhancements affect LSF resource connector.
- **Resource management**
  The following new features affect resource management and allocation.

- **Security**
  The following new features affect cluster security.
- **Other changes to IBM Spectrum LSF**
  The following changes affect other aspects of LSF behavior.

GPU enhancements

The following enhancements affect LSF GPU support.

Disable CUDA_VISIBLE_DEVICES for MPS jobs

LSF now allows you to disable the CUDA_VISIBLE_DEVICES environment variable for MPS jobs.

When the CUDA_VISIBLE_DEVICES environment variable is disabled, LSF does not set the CUDA_VISIBLE_DEVICES*<number>* environment variables for tasks, so LSF MPI does not set CUDA_VISIBLE_DEVICES for the tasks. LSF just sets the CUDA_VISIBLE_DEVICES*<number>* environment variables for tasks, not CUDA_VISIBLE_DEVICES. LSF MPI converts the CUDA_VISIBLE_DEVICES*<number>* environment variables into CUDA_VISIBLE_DEVICES and sets that for the tasks.

LSF uses the CUDA_VISIBLE_DEVICES environment variable to tell user applications which GPUs are allocated and to make only those GPUs visible to the application. To disable CUDA_VISIBLE_DEVICES for MPS jobs, add the "`,nocvd`" keyword to the existing `mps` value in the GPU resource requirements string (that is, the **bsub -gpu** command option, LSB_GPU_REQ parameter in the lsf.conf file, and the GPU_REQ parameter in the lsb.queues and lsb.applications files). Specify `mps=yes,nocvd` or `mps=yes,shared,nocvd` in the GPU requirements to disable CUDA_VISIBLE_DEVICES in MPS jobs.

`LSB_GPU_NEW_SYNTAX=extend` must be defined in the lsf.conf file to work with MPS jobs.

Related tasks

- Configuring GPU resource requirements
- Submitting jobs that require GPU resources

Related reference

- **bsub -gpu** command option
- LSB_GPU_NEW_SYNTAX parameter in the lsf.conf file
- LSB_GPU_REQ parameter in the lsf.conf file
- GPU_REQ parameter in the lsb.queues file
- GPU_REQ parameter in the lsb.applications file

Block distribution of allocated GPUs

LSF now allows you to enable block distribution of allocated GPUs, which means that LSF can distribute the allocated GPUs of a job as blocks when the number of tasks is greater than the requested number of GPUs.

For example, if a GPU job requests to run on a host with 4 GPUs and 40 tasks, block distribution assigns GPU0 for ranks 0-9, GPU1 for ranks 10-19, GPU2 for tanks 20-29, and GPU3 for ranks 30-39.

To enable block distribution of allocated GPUs, specify `block=yes` in the GPU resource requirements string (that is, the **bsub -gpu** command option, LSB_GPU_REQ parameter in the lsf.conf file, and the GPU_REQ parameter in the lsb.queues and lsb.applications files).

`LSB_GPU_NEW_SYNTAX=extend` must be defined in the lsf.conf file to enable block distribution of allocated GPUs.

Related tasks

- Configuring GPU resource requirements
- Submitting jobs that require GPU resources

Related reference

- **bsub -gpu** command option
- LSB_GPU_NEW_SYNTAX parameter in the lsf.conf file
- LSB_GPU_REQ parameter in the lsf.conf file
- GPU_REQ parameter in the lsb.queues file
- GPU_REQ parameter in the lsb.applications file

Pack scheduling for GPU shared mode jobs

LSF now allows you to enable pack scheduling for shared mode GPU jobs.

LSF packs multiple shared mode GPU jobs to allocated GPUs. When enabled, LSF packs multiple shared mode GPU jobs to allocated GPUs. LSF schedules shared mode GPUs as follows:

1. LSF sorts the candidate hosts (from largest to smallest) based on the number of shared GPUs that already have running jobs, then by the number of GPUs that are not exclusive.
   If the order[] keyword is defined in the resource requirements string, after sorting order[], LSF re-sorts the candidate hosts by the gpack policy (by shared GPUs that already have running jobs first, then by the number of GPUs that are not exclusive). The gpack policy sort priority is higher than the order[] sort.

2. LSF sorts the candidate GPUs on each host (from largest to smallest) based on the number of running jobs.

After scheduling, the shared mode GPU job packs to the allocated shared GPU that is sorted first, not to a new shared GPU.

Previously, LSF allocated additional GPUs to new shared mode GPU jobs. For example, a host has 4 GPUs and one GPU job is running on a shared mode GPU. When another shared mode GPU job is scheduled to the host, another GPU is allocated to the new job. Therefore, two shared mode GPUs are running two GPU shared mode jobs. With pack scheduling enabled, LSF allocates one shared mode GPU to run both GPU shared mode jobs if that GPU is sorted first.

To enable pack scheduling for shared mode GPU jobs, specify `gpack=yes` in the GPU resource requirements string (that is, the **bsub -gpu** command option, LSB_GPU_REQ parameter in the lsf.conf file, and the GPU_REQ parameter in the lsb.queues and lsb.applications files).

`LSB_GPU_NEW_SYNTAX=extend` must be defined in the lsf.conf file to enable pack scheduling of shared mode GPU jobs.

Related tasks

- Configuring GPU resource requirements
- Submitting jobs that require GPU resources

Related reference

- **bsub -gpu** command option
- LSB_GPU_NEW_SYNTAX parameter in the lsf.conf file
- LSB_GPU_REQ parameter in the lsf.conf file
- GPU_REQ parameter in the lsb.queues file
- GPU_REQ parameter in the lsb.applications file

NVIDIA Data Center GPU Manager (DCGM) integration updates

LSF, Version 10.1 Fix Pack 2 integrated with NVIDIA Data Center GPU Manager (DCGM) to work more effectively with GPUs in the LSF cluster. LSF now integrates with Version 1.7.2 of the NVIDIA Data Center GPU Manager (DCGM) API.

Enable the DCGM integration by defining the LSF_DCGM_PORT parameter in the lsf.conf file.

Related reference

- LSF_DCGM_PORT parameter in the lsf.conf file

Job scheduling and execution

The following new features affect job scheduling and execution.

Use host attributes to schedule jobs using attribute affinity

LSF now allows you to create attributes for hosts and to use these attributes for job affinity scheduling. Submit jobs and define host preferences based on which hosts have specific attributes.

Host attributes give you a flexible way of specifying host affinity preferences to give you more control over host selection for job scheduling. Use the **battr create** command to create attributes that are assigned to hosts. Use the **bsub -jobaff** command option when submitting jobs to define host preferences based on host attributes. If you specified a preferred attribute with **bsub -jobaff** and this attribute does not exist, LSF automatically creates the attribute on the execution hosts after dispatching the job to the hosts.

For example, you can specify that jobs can only run (or prefer to run) on hosts with a specific attribute or on hosts that do not possess a specific attribute. If you enabled same job affinity, you can also specify that a job can only run (or prefer to run) on hosts or compute units that are already running a specific job, or on hosts or compute units that are not running a specific job.

Specify the users who can create host attributes with the ATTR_CREATE_USERS parameter. Limit the maximum number of host attributes that can exist in the cluster (to limit the decrease in cluster performance) with the ATTR_MAX_NUM parameter (the default value is 100). Change the time-to-live for all host attributes

with the ATTR_TTL parameter (the default value is 1 hour). Enable the SAME_JOB_AFFINITY parameter to enable same job affinity. All host attribute and affinity parameters are in the lsb.params file.

Related concepts

- Job affinity scheduling with host attributes

Related reference

- ATTR_CREATE_USERS parameter in the lsb.params file
- ATTR_MAX_NUM parameter in the lsb.params file
- ATTR_TTL parameter in the lsb.params file
- SAME_JOB_AFFINITY parameter in the lsb.params file
- **battr** command
- **bhosts -attr** command option
- **bsub -jobaff** command option

Control job forwarding decisions

When using the LSF multicluster capability LSF now allows you to specify the job forwarding behavior and the amount of time after job submission and scheduling for LSF to revert to the default job forwarding behavior.

To specify the amount of time, in seconds, for LSF to revert to default job forwarding behavior, define the MC_FORWARD_DELAY parameter in the lsb.queues file.

Specify a positive integer for LSF to only attempt local hosts and not forward the job to a remote cluster for the specified amount of time . Specify a negative integer for LSF to only forward the job to a remote cluster and not attempt local hosts for the specified amount of time.

After this amount of time, this parameter no longer takes effect and LSF reverts to the default job forwarding behavior, which is to attempt the local hosts first, then forward the job to a remote cluster if this failed.

Related reference

- MC_FORWARD_DELAY parameter in the lsb.queues file

lsrun and lsgrun use UNIX group information on the client side

The **lsrun** and **lsgrun** commands now consider the LSF_UGROUP_TRANSFER parameter in the lsf.conf file to use the UNIX group information that is set by the user on the client side.

If the LSF_UGROUP_TRANSFER parameter is enabled in the lsf.conf file, tasks in the execution side that the **lsrun** or **lsgrun** commands run use the UNIX group information that is set by the user on the client side.

Related reference

- **lsgrun** command
- **lsrun** command
- LSF_UGROUP_TRANSFER parameter in the lsf.conf file

Reschedule IBM CSM jobs that fail with specific error codes

You can now enable LSF to reschedule IBM Cluster Systems Manager (CSM) jobs that are stage-in or non-transfer jobs that fail during CSM setup if they fail with certain CSM API error codes.

To enable this feature and to specify the CSM API error codes for which LSF will reschedule the jobs, specify the RESCHED_UPON_CSM_SETUP_ERROR parameter in the lsb.params file.

If CSM setup fails with one of the specified CSM API error codes, LSF repeats the attempt to schedule or dispatch the job.

Related tasks

- Configuring the CSM integration

Related reference

- RESCHED_UPON_CSM_SETUP_FAILURE parameter in the lsb.params file

Container support

The following new feature affects LSF support for containers.

Docker container names

You can now enable LSF to automatically assign a name to a Docker container when it creates the Docker container.

To enable this feature, set the ENABLE_CONTAINER_NAME parameter to True in the lsfdockerlib.py file.

When running a container job, specify the --name option in the options keyword configuration for LSF to assign a name to the container.

The container name uses the following naming convention:

- Normal jobs and **blaunch** parallel job containers: *<cluster_name>*.job.*<job_id>*
- Array jobs and array **blaunch** parallel job containers: *<cluster_name>*.job.*<job_id>*.*<job_index>*
- **blaunch** parallel job task containers: *<cluster_name>*.job.*<job_id>*.task.*<task_id>*
- Array **blaunch** parallel job task containers: *<cluster_name>*.job.*<job_id>*.*<job_index>*.task.*<task_id>*

Related tasks

- Configuring to run Docker jobs

Related reference

- CONTAINER in the lsb.queues file.
- CONTAINER in the lsb.applications file.

Command output formatting

The following enhancements affect LSF command output formatting.

bsub -K displays the job ID after the job is finished

You can now enable the **bsub -K** command option to display the job ID after the job is finished.

The **bsub -K** job submission option enables the command to wait for the job to complete and to send job status messages to the message. When the job is finished, this command option normally sends the message "`Job is finished`". You can now enable LSF to display the job ID with the job finish message by specifying `LSB_SUBK_SHOW_JOBID=Y` in the lsf.conf file or environment variable.

Related reference

- **bsub -K** command option
- LSB_SUBK_SHOW_JOBID parameter in the lsf.conf file

Add a reason when stopping or resuming a job

LSF now has a new -C option added to the bstop and bresume commands. This option gives the user the option to add a reason as to why the job is being stopped or resumed. These stop or resume reasons and the related hosts appear in the **bhist -l** command output. You can also use the **bjobs -o** command option or the LSB_BJOBS_FORMAT parameter in the lsf.conf file to add the `suspend_reason`, `resume_reason`, `suspend_issue_host`, `resume_issue_host`, and `kill_issue_host` fields (in addition to the existing `kill_reason` field) to customized output.

Related reference

- **bstop -C** command option
- **bresume -C** command option

Not displaying all individual hosts for the "all" shared resource

The **lsload -s**, **lshosts -s**, and **bhosts -s** commands, which that display hosts that share cluster resources, now display `ALL` instead of displaying each individual host in the cluster if the cluster is configured to allow all hosts to share a resource.

If the LOCATION parameter in the lsf.cluster.*clustername* file is set to `all` to indicate that the resource is shared by all hosts in the cluster, the `LOCATION` field in the **lsload -s**, **lshosts -s**, and **bhosts -s** commands also display `ALL`, instead of displaying each individual host in the cluster. This improves performance for clusters that define a large amount of resources and reduces clutter of the command output.

To display the individual names of all the hosts in the cluster in the command output, specify the new -loc option together with the -s option.

Related reference

- **bhosts -s** command option
- **lshosts -s** command option
- **lsload -s** command option

Displaying GPU resources with LSF resources

You can now enable the **lsload -s**, **lsload -l**, and **bhosts -l** commands, which display LSF resources, to no longer display information about GPU resources. That is, you can now enable these options to not display `gpu_<num>n` resources by enabling the **mbatchd** and **mbschd** daemons to ignore GPU resources.

This improves the performance of the **mbatchd** and **mbschd** daemons and reduces clutter of the command output.

To enable the daemons to ignore GPU resources, specify `LSF_GPU_RESOURCE_IGNORE=Y` in the lsf.conf file. Do not set to `Y` if you are using LSF RTM, otherwise LSF RTM will not show host GPU information. This is because LSF RTM uses LSF resources to get host GPU information.

Once the daemons ignore GPU resources, you can still view GPU-based information, including GPU resources, by using the existing -gpu option of the **lsload**, **lshosts**, and **bhosts** commands, or the existing -gpuload option of the **lsload** command.

Related reference

- **bhosts -gpu** command option
- **lshosts -gpu** command option
- **lsload -gpu** and **-gpuload** command options

Performance enhancements

The following enhancements affect performance.

Improved scheduling efficiency

LSF introduced a number of enhancements to job dispatch control to improve scheduling efficiency:

- LSF can now publish job decisions in a decision package before the end of the scheduling cycle. The maximum number of job decisions that can accumulate before LSF publishes the decisions in a decision package is 300, or you can change this by specifying the JOB_DISPATCH_PACK_SIZE parameter in the lsb.params file:
  `JOB_DISPATCH_PACK_SIZE=`*integer*
- The default value of the LSB_MAX_JOB_DISPATCH_PER_SESSION parameter in the lsf.conf file is changed to 15000. The previous default value was Min (MAX(300, Total CPUs), 3000). If the JOB_DISPATCH_PACK_SIZE parameter is set to `0` in the lsb.params, the default value reverts to this previous default value.
- You can now specify a job scheduling interval threshold time. **mbschd** checks the threshold during the regular job scheduling phase and the backfilling phase, and skips the rest of the jobs if the scheduling

cycle exceeds this time. To specify the threshold time, add a second number, in seconds, to the JOB_SCHEDULING_INTERVAL parameter in the lsb.params file:
`JOB_SCHEDULING_INTERVAL=`*min_time* [`ms`] [*max_time*]

Do not specify a time that is lower than your average scheduling time because this decreases scheduling efficiency. Run the **badmin perfmon view** command to determine the average value, which you can use to determine a reasonable value to set for the maximum scheduling cycle time. The default maximum is 45 seconds.

- The default value of the MAX_SBD_CONNS parameter in the lsb.params file is changed to 2 * numOfHosts + 300. The previous value was numOfHosts + (2 * LSB_MAX_JOB_DISPATCH_PER_SESSION)+200. This means that the maximum number of open file connections between **mbatchd** and **sbatchd** no longer depends on the LSB_MAX_JOB_DISPATCH_PER_SESSION parameter because LSF now persists **sbatchd** connections.

Related reference

- JOB_DISPATCH_PACK_SIZE parameter in the lsb.params file.
- JOB_SCHEDULING_INTERVAL parameter in the lsb.params file.
- MAX_SBD_CONNS parameter in the lsb.params file.
- LSB_MAX_JOB_DISPATCH_PER_SESSION parameter in the lsf.conf file.
- LSF_ACCEPT_NUMCLIENTS parameter in the lsf.conf file.

Improved LSF response time

LSF introduced a number of enhancements to improve LSF response time:

- The maximum of number new client connections to the **mbatchd** port that **mbatchd** accepts during each scheduling cycle is increased from the previous value of 1 to a default value of 6. You can now change this value by specifying the new LSF_ACCEPT_NUMCLIENTS parameter in the lsf.conf file:
`LSF_ACCEPT_NUMCLIENTS=`*integer*

- LSF has improved the response time of **mbatchd** client validation for jobs that are submitted from new floating clients.
- The **badmin hopen -i lock** and **badmin hclose -i lock** command options now use **mbatchd** to perform the tasks on a host list. Previously, these commands were issued by the client to each host one by one.
- The **lsload -s**, **lshosts -s**, and **bhosts -s** commands, which that display hosts that share cluster resources, now display `ALL` instead of displaying each individual host in the cluster if the cluster is configured to allow all hosts to share a resource. For more information, refer to Not displaying all individual hosts for the "all" shared resource.
- You can now enable the **mbatchd** and **mbschd** daemons to ignore GPU resources. This means that the **lsload -s**, **lsload -l**, and **bhosts -l** commands, which display LSF resources, no longer display information about GPU resources. That is, these options do not display `gpu_<`*num*`>n` resources, which improves LSF response time because there are fewer LSF resources to manage and display. To enable the daemons to ignore GPU resources, specify `LSF_GPU_RESOURCE_IGNORE=Y` in the lsf.conf file. Do not enable if you are using LSF RTM, otherwise LSF RTM will not show host GPU information. For more information, refer to Displaying GPU resources with LSF resources.

Related reference

- LSF_ACCEPT_NUMCLIENTS parameter in the lsf.conf file.

Relax additional constraints on reusing resource allocations for finished short jobs

LSF now allows you to relax additional constraints on the pending jobs that can reuse resource allocations for finished jobs.

RELAX_JOB_DISPATCH_ORDER is an existing parameter in the lsb.params and lsb.queues files that enables LSF to allow multiple jobs with common resource requirements to run consecutively on the same resource allocation for a finished job. To ensure that limits are not violated, LSF selects pending jobs that belong to the same user and have other attributes in common.

LSF now allows you to further relax the constraints on the pending jobs that can reuse the resource allocation for finished jobs. This allows more jobs to reuse the resource allocation, but might result in resource limits and policies being temporarily violated because these limits and policies are relaxed.

To relax the constraints, use the SHARE[] keyword in the RELAX_JOB_DISPATCH_ORDER parameter. The SHARE[] keyword specifies constraints that the **mbatchd** daemon no longer has to apply when determining which pending jobs can reuse the resource allocation of a finished job. If a job is finished and LSF does not find any pending jobs with the same user or other attributes in common, LSF considers the specifications in the SHARE[] keyword. Specify one or more of the following keywords within SHARE[] for LSF to also consider the following pending jobs:

SHARE[[user] [group] [project]]

user
: Pending jobs that do not have the same job owner as the finished job.

group
: Pending jobs that are not associated with the same fair share group (**bsub -G** command option) as the finished job.

project
: Pending jobs that are not assigned to the same project (**bsub -P** command option) as the finished job.

For example, if there are no pending jobs with the same common attributes, set the following to enable pending jobs that belong to different users and are associated with different fair share groups to also reuse the resource allocation:

```
RELAX_JOB_DISPATCH_ORDER=SHARE[user group]
```

If using the LSF multicluster capability, RELAX_JOB_DISPATCH_ORDER applies only to forwarded jobs.

Related reference

- RELAX_JOB_DISPATCH_ORDER parameter in the lsb.params file.
- RELAX_JOB_DISPATCH_ORDER parameter in the lsb.queues file.

Resource connector enhancements

The following enhancements affect LSF resource connector.

Specify host project ID for Google Cloud to generate subnet and VPN values

You can now specify a host project ID for the LSF resource connector for Google Cloud to generate subnet and VPN values.

To specify a host project ID, specify the new hostProject parameter in the googleprov_templates.json file. The LSF resource connector uses the specified host project ID to generate the VPN and subnet values instead of the Google Cloud Project ID (that is, the GCLOUD_PROJECT_ID parameter in the googleprov_config.json file). If not specified, LSF resource connector continues to use the Google Cloud Project ID to generate subnet and VPN values.

Related reference

- hostProject parameter in the googleprov_templates.json file

Configure launch templates for AWS

You can now specify a launch template in the LSF resource connector template for Amazon Web Services (AWS) and specify which version of the launch template to use. A launch template is an Amazon Elastic Compute Cloud (EC2) feature that reduces the number of steps that are required to create an AWS instance by capturing all launch parameters within one resource. You can launch both On-Demand and Spot Instances with launch templates.

To specify a launch template, define the launchTemplateId parameter in the awsprov_templates.json file. To select a specific launch template version, define the launchTemplateVersion parameter in the awsprov_templates.json file.

Related tasks

- Configure launch templates for AWS

Related reference

- awsprov_templates.json file

Configure EFA network interfaces for AWS

You can now attach an Elastic Fabric Adapter (EFA) network interface to the LSF resource connector template for AWS. EFA is a network interface for Amazon Elastic Compute Cloud (EC2) instances that allows you to run HPC applications with improved levels of communication between several different nodes. You can attach EFA network interfaces to both On-Demand and Spot Instances.

To attach an EFA network interface to an AWS instance, you can either set the interfaceType parameter value to `efa` in the awsprov_templates.json file or enable it from the network interface section of the launch templates in the AWS EC2 Console.

You can only specify an EFA network interface for supported AMI or instance types. For more details on supported AMI or instance types for EFA interfaces, refer to the Amazon Web Services website

(https://aws.amazon.com/).

Related tasks

- Attach EFA network interfaces to AWS templates

Related reference

- awsprov_templates.json file

Configure image names for CycleCloud

You can now specify the image name of the Azure Machine Image in the LSF resource connector template for Microsoft Azure CycleCloud. This also allows you to specify versions of the image from the Azure Shared Image Gallery.

To specify the image name of the Azure Machine Image, define the new imageName parameter in the cyclecloudprov_templates.json file.

Related reference

- cyclecloudprov_templates.json file

Resource management

The following new features affect resource management and allocation.

Improved resource utilization for guarantee policies

LSF now has a new scheduling algorithm for package guarantees to improve resource utilization, and includes changes to command output to show additional information for guaranteed resource pools and loaning information.

Previously, the LSF administrator chooses a package size for a guarantee policy on a set of hosts when configuring the package guarantee policy. To be effective, the package size must be at least the same as the requirements of a single job that is given a guarantee in the pool. When reserving the resources for jobs in the guarantee pool, LSF reserves multiples of the entire package size, so even if the job does not use the entire package size, the entire package is reserved. This means that smaller jobs might remain pending even if there are sufficient slots and memory available on a host to allow the job to run, simply because the entire package was reserved for a running job that does not necessarily require the entire package size.

The new scheduling algorithm relaxes requirements for LSF to reserve multiples of the entire package size to avoid problems with resource utilization. At the beginning of each scheduling session, LSF partitions the hosts within each guarantee package or slot pool into owner and shared hosts based on the configured guarantees. Owner hosts are only for owners who have not exceeded the guarantee, while shared hosts have no such restriction. LSF ensures that there are enough owner hosts to honor the configured guarantees so that the total number of packages on all owned hosts is at least the number of packages that are configured for owners.

The **bresources -g** command option shows additional information for package and slot type guarantees, including loaning information, which makes the guarantee policy easier to understand. In addition, the **bhist -l** and **bjobs -l** command options show the resources that are borrowed from the GSLA pools, and loaning information from these pools.

For jobs that use guarantees, LSF supports queue-based preemption in package and slot type guarantees. A higher priority job can preempt any job from the pool of shared hosts. For host and license pools, LSF supports queue-based preemption for owner jobs. A higher priority owner job whose guarantee has not been fulfilled can preempt any loaned owner job in the pool of owner hosts.

Enable these new behaviors by setting the new SIMPLIFIED_GURANTEE parameter to `Y` in the lsb.params file. These new behaviors are automatically enabled at the time of the installation.

In addition, the RETAIN keyword is now deprecated and replaced with IDLE_BUFFER in the GuaranteedResourcePool section of the lsb.resources file. Use IDLE_BUFFER instead of RETAIN when defining loan policies in the guarantee pool.

Related concepts

- Configuration overview of guaranteed resource pools

Related tasks

- Viewing configured guaranteed resource pools
- Viewing guarantee policy information

Related reference

- **bresources** command
- LOAN_POLICIES parameter in the lsb.resources file.
- SIMPLIFIED_GUARANTEE parameter in the lsb.params file.

Use multiple reasons to close hosts

LSF now allows you to specify multiple reasons for closing hosts by specifying lock IDs when closing the hosts.

This allows multiple users to keep a host closed for different reasons. For example, userA might be updating an application while userB is configuring the operating system. The host remains closed until both users complete their tasks and open the host using their specific lock IDs.

Use the new -i option with the **badmin hclose** command when closing a host to specify a lock ID to attach to the closed host. If the host is already closed, running **badmin hclose -i** *lock_id* stacks the new lock ID with any existing lock IDs on the closed host to ensure that the host remains closed if at least one lock ID is still attached to the host.

Each lock ID is a string that can contain up to 128 alphanumeric and underscore (_) characters. The keyword all is reserved and cannot be used as the lock ID.

Use -i together with the -C option to attach an administrator message to the lock ID, as shown in the following example:

```
badmin hclose -i "lock_update_app1" -C "Updating application1"
```

If another user closes the host with a different lock ID, as shown in the following example, the host remains closed even if the first user opens the host:

```
badmin hclose -i "lock_config_os" -C "Configuring OS"
```

Use the new -i option with the **badmin hopen** command to remove the specified lock ID from the closed host. You can specify a space-separated list of lock IDs, or use the all keyword to remove all lock IDs from the host. If there are no longer any lock IDs attached to the host, this command also opens the host. To remove the lock ID from the previous example (lock_config_os), run the following command:

```
badmin hopen -i "lock_config_os" -C "Finished OS configuration"
```

Since the first user is still updating application1, the host remains closed until the first user removes the lock_update_app1 lock ID.
You can specify a space-separated list of lock IDs to remove multiple lock IDs, or use the all keyword to remove all lock IDs from the closed host. Once all lock IDs are removed from the host, the command also opens the closed host.

The **bhosts -l** command option to shows all lock IDs and comments in tabular format if there are any lock IDs that are attached to the closed host.

Related tasks

- Use lock IDs to specify multiple reasons for closing a host

Related reference

- **badmin hclose** and **hopen** commands
- **bhosts -l** command option

Connect to a job execution host or container

LSF now allows you to connect to a job execution host or container for debugging or general connectivity purposes.

Use the new **battach** command to directly connect (attach) to an existing job execution host or container for a specified job. Once connected, you can run shell commands from the interactive command line. **battach** runs an interactive /bin/sh shell by default, but you can use the **battach -L** command option or the LSB_BATTACH_DEFAULT_SHELL environment variable to specify an alternate shell. If both are specified, the **battach -L** command option overrides the LSB_BATTACH_DEFAULT_SHELL environment variable.

For sequential jobs, the **battach** command connects to the job execution host for non-Docker jobs or the job container on the job execution host for Docker jobs. For parallel jobs, use the **battach -m** command option to connect to the specified job execution host for non-Docker jobs or the specified job container on the job execution host for Docker jobs.

Related tasks

- Connect to a job execution host or container

Related reference

- **battach** command

Submit job tasks into a PAM session

You can now enable LSF to open a PAM session when submitting jobs to Linux hosts using PAM.

When enabled, LSF opens a PAM session for the user and executes a RES process into that session. The RES process executes the job task, then LSF disowns the process. This means that other LSF integrations are automatically handled in the PAM session.

To enable the PAM session, set the USE_PAM_CREDS parameter to session in the lsb.queues or lsb.applications file. You can also use the new limits keyword together with the session keyword (that is, by defining USE_PAM_CREDS=limits session). Setting the limits keyword is functionally identical to enabling USE_PAM_CREDS=y except that you can define limits together with the session keyword.

In addition, if USE_PAM_CREDS is enabled, LSF can apply PAM limits to the specified application or queue when its job is dispatched to a Linux host using PAM LSF. If LSF limits are more restrictive than PAM limits, LSF limits are used, otherwise PAM limits are used. PAM limits are system resource limits defined in the limits.conf file.

Note: When configuring Linux PAM to be used with LSF, you must configure Linux PAM so that it does not ask users for their passwords because Jobs are not usually interactive.
Depending on the USE_PAM_CREDS parameter setting, LSF assumes that the following Linux PAM services are created:

- If USE_PAM_CREDS is set to `y` or `limits`, LSF assumes that the Linux PAM service "**lsf**" is created.
- If USE_PAM_CREDS is set to `session`, LSF assumes that the Linux PAM service "**lsf-***<clustername>*" is created.
- If USE_PAM_CREDS is set to `limits session`, LSF assumes that the Linux PAM services "**lsf**" and "**lsf-***<clustername>*" are created.

It is also assumed that the "**lsf**" service is used in conjunction with the /etc/security/limits.conf file.
The job **sbatchd** daemon checks the **lsf** service, and the job or task RES daemon checks the **lsf-***<clustername>* service

Related tasks

- Configure a PAM file

Related reference

- USE_PAM_CREDS parameter in the lsb.applications file
- USE_PAM_CREDS parameter in the lsb.queues file

Job group limits for forwarded jobs

When using the LSF multicluster capability, job group limits now apply to jobs that are forwarded to another cluster when using the job forward mode.

When using global limit scheduling, job group limits are still applied when the jobs are forwarded to another cluster.

To enable job group limits to apply to jobs forwarded to another cluster when using the job forward mode with the LSF multicluster capability, edit the lsb.params file and define GLOBAL_LIMITS=Y.

The global policy daemon (**gpolicyd**) is responsible for applying job group limits to forwarded jobs. To use job group limits on forwarded jobs, you must also ensure that the LSB_GPD_PORT and LSB_GPD_CLUSTER parameters are defined correctly in the lsf.conf file for **gpolicyd**.

Related tasks

- Global limits for job resource allocations

Related reference

- Limit section in the lsb.globalpolicies file
- **blimits -gl** command option

Global limits for job resource allocations

When using the LSF multicluster capability, you can now configure global limit scheduling to apply resource allocations to all clusters.

Batch features for the LSF multicluster capability, job forward mode, or lease mode are not required to use global limits. Therefore, you can use global limit scheduling if you have multiple LSF clusters without using the LSF multicluster capability.

When using global limit scheduling, job group limits are still applied when the jobs are forwarded to another cluster.

To enable global limit scheduling among all clusters, edit the lsb.params file and define GLOBAL_LIMITS=Y.

The global policy daemon (**gpolicyd**) is responsible for applying global limits. To use global limits, you must also ensure that the LSB_GPD_PORT and LSB_GPD_CLUSTER parameters are defined correctly in the lsf.conf file for **gpolicyd**.

To define the global resource allocation limits, edit the lsb.globalpolicies file and specify the global resource allocation limits in the Limit section. Specify global resource allocations the same way you would specify local resource allocation limits in the Limit sections of the lsb.resources file.

Run the **blimits -gl** command option to display the current usage of global limits.

Run the **blimits -gl -c** command option to display the global limits configurations in the lsb.globalpolicies file.

Note:

- There might be conflicts when allocating the available allocation limit to different clusters. Global limit scheduling might allow conflicts to occur temporarily between clusters. After an overcommitted job is done, the global limits can return. If the RELAX_JOB_DISPATCH_ORDER parameter in the lsb.params file is enabled in the cluster, the allocation can be reused by other jobs, and the resource conflicts can remain in place for a longer period of time.
- Global limits are evenly divided between different clusters, without any fairness considerations. Larger jobs that require a large share of the resource allocation might pend if that amount is larger than the

per-cluster limit.
For example, if the `res1` resource has a global limit of 12 units among three clusters (4 per cluster), a job that requires 5 units of the `res1` resource pends because this is larger than the per-cluster limit.

Related tasks

- Global limits for job resource allocations

Related reference

- Limit section in the lsb.globalpolicies file
- **blimits -gl** command option

Security

The following new features affect cluster security.

Root privileges

The LSF_ROOT_REX parameter is an existing parameter in the lsf.conf file that specifies root execution privileges for jobs from both local and remote hosts. While LSF_ROOT_REX=N (which disables root execution privileges) is the default setting since this parameter's introduction, there were a number of issues related to sites inadvertently leaving this parameter enabled. Due to these issues, the LSF_ROOT_REX parameter is now obsolete and no longer allows root execution privileges for jobs from local and remote hosts. Any actions that were performed with root privileges must instead be performed as the LSF administrator.

If you need to temporarily run LSF commands with root privileges, specify `LSF_ROOT_USER=Y` in the lsf.conf file. When you are done, you must disable this parameter to ensure that your cluster remains secure.

If you need to temporarily run LSF License Scheduler commands with root privileges, specify `LS_ROOT_USER=Y` in the lsf.licensescheduler file. This parameter allows the root user to run the **bladmin**, **blkill**, **globauth**, and **taskman** commands. When you are done, you must disable this parameter to ensure that your cluster remains secure.

If you need to enable root privileges on hosts for LSF Application Center, LSF RTM, or LSF Explorer, specify a space-separated list of hosts in the LSF_ADDON_HOSTS parameter in the lsf.conf file. The root users on these specified hosts can remotely execute commands. You must also set `LSF_DISABLE_LSRUN=N` in the lsf.conf file to enable hosts that are running LSF Application Center to use the **lsrun** and **lsgrun** commands.

Related reference

- LSF_ROOT_USER parameter in the lsf.conf file
- LSF_ADDON_HOSTS parameter in the lsf.conf file
- LS_ROOT_USER parameter in the lsf.licensescheduler file
- LSF_ROOT_REX parameter in the lsf.conf file

Other changes to IBM Spectrum LSF

The following changes affect other aspects of LSF behavior.

Job submission option files

LSF now allows you to create and use files that contain job submission options. You can create job submission files that contain supported **bsub** command options in JSON and YAML formats. Once created, use the new **bsub -json** and **-yaml** command options to specify the JSON and YAML job submission files.

Related concepts

- Job submission option files

Related tasks

- Submit a job with a JSON file to specify job submission options (bsub -json)
- Submit a job with a YAML file to specify job submission options (bsub -yaml)

Related reference

- **bsub -json** command option
- **bsub -yaml** command option

View the job submission environment

LSF now allows you to view the job submission environment. Previously, you could only do this by navigating to the LSF info directory (as specified by the LSB_JOBINFO_DIR environment variable) and directly viewing the job script.

Use the new **bjobs -env** command option to view the environment variables for the specified job. Use the new **bjobs -script** command option to view the job script. You cannot use both options together, and you must specify a single job ID or job array element. Multiple job IDs are not supported.

Related tasks

- View the job submission environment

Related reference

- **bjobs -env** command option
- **bjobs -script** command option

LSF Application Center desktop clients as submission hosts

LSF can now use LSF Application Center desktop client hosts as job submission hosts. Users can log in to the desktop client host and submit jobs to the LSF Application Center Server.

To enable this feature on the LSF Application Center desktop client, set the LSF_DESKTOP_CLIENT environment variable to `yes`. You must ensure also that you set the LSF environment on the desktop client.

If the LSF Application Center desktop client is running on a Windows machine, set the %LSF_TOP% environment variable to `C:\LSF_Desktop_Client`.

Use the **paclogon** command to log in to the desktop client and use the LSF **bsub** command to submit jobs. Commands that you run from the LSF Application Center desktop client are executed on the LSF Application Center Server.

New platform support

LSF supports the following platforms:

- RHEL 7.8, kernel 3.10, glibc 2.17
- RHEL 8.1/8.2, kernel 4.18.0, glibc 2.28

What's new in IBM Spectrum LSF Version 10.1 Fix Pack 9

The following topics summarize the new and changed behavior in LSF 10.1 Fix Pack 9.

Release date: 20 November 2019

- **GPU enhancements**
  The following enhancements affect LSF GPU support.
- **Job scheduling and execution**
  The following new features affect job scheduling and execution.
- **Container support**
  The following new feature affects LSF support for containers.
- **Command output formatting**
  The following enhancements affect LSF command output formatting.
- **Performance enhancements**
  The following enhancements affect performance.
- **Resource management**
  The following new features affect resource management and allocation.
- **Data collection**
  The following new features affect IBM Spectrum LSF data collection.
- **Other changes to IBM Spectrum LSF**
  The following changes affect other aspects of LSF behavior.

GPU enhancements

The following enhancements affect LSF GPU support.

Enable MPS daemon sharing for GPU jobs

LSF now allows you to share an NVIDIA Multi-Process Service (MPS) daemon for multiple GPU jobs if they are submitted by the same user with the same resource requirements.

To enable MPS daemon sharing, add the "`,share`" keyword to the existing `mps` value in the GPU resource requirements string (that is, the **bsub -gpu** command option, LSB_GPU_REQ parameter in the lsf.conf file, and the GPU_REQ parameter in the lsb.queues and lsb.applications files). Specify `mps=yes,share`, `mps=per_socket,share`, or `mps=per_gpu,share` in the GPU requirements to enable LSF to share the MPS daemon on the host, socket, or GPU for jobs that are submitted by the same user with the same resource requirements.

`LSB_GPU_NEW_SYNTAX=extend` must be defined in the lsf.conf file to enable MPS daemons and MPS daemon sharing.

Related concepts

- Example GPU job submissions

Related tasks

- Configuring GPU resource requirements
- Submitting jobs that require GPU resources

Related reference

- **bsub -gpu** command option
- LSB_GPU_NEW_SYNTAX parameter in the lsf.conf file
- LSB_GPU_REQ parameter in the lsf.conf file
- GPU_REQ parameter in the lsb.queues file
- GPU_REQ parameter in the lsb.applications file

Merge individual options in GPU requirements

In previous versions of LSF, when specifying GPU resource requirements at multiple levels (that is, at the job level with the **bsub -gpu** command option, the application or queue level with the GPU_REQ parameter in the lsb.applications or lsb.queues files, or the cluster level with the LSB_GPU_REQ parameter in the lsf.conf file), the entire GPU requirement string overrides the GPU requirement strings at the lower levels of precedence. Even if an individual option is not specified in the higher level GPU requirement string, the default value of the higher level GPU requirement string takes precedence.

LSF now allows you to merge all individual options in GPU requirement strings separately. Any specified options override the any options that are specified at the lower levels of precedence. If an individual option is not specified, but is explicitly specified at a lower level, then the highest level for which the option is specified takes precedence. To enable this feature, specify the new parameter `GPU_REQ_MERGE=Y` in the lsb.params file. In addition, `LSB_GPU_NEW_SYNTAX=extend` must be defined in the lsf.conf file to enable the GPU_REQ_MERGE parameter.

Related tasks

- Configuring GPU resource requirements
- Submitting jobs that require GPU resources

Related reference

- **bsub -gpu** command option
- GPU_REQ_MERGE parameter in the lsb.params file
- LSB_GPU_REQ parameter in the lsf.conf file
- GPU_REQ parameter in the lsb.queues file
- GPU_REQ parameter in the lsb.applications file

Specify GPU allocation method

You can now specify the GPU resource reservation method by specifying the method in the **bsub -gpu** option, by specifying the GPU_REQ parameter in the lsb.applications or lsb.queues file, or by specifying the LSB_GPU_REQ parameter in the lsf.conf file. Previously, you could only specify the resource reservation method at the global level by specifying the METHOD parameter in the ReservationUsage section of the lsb.resources file. The GPU resource reservation value and method at the job level overrides the value and method at the application level, which overrides the value and method at the queue level, which overrides the value and method at the cluster level.

Specify the GPU resource reservation method by using the /task or /host keyword after the GPU numeric value in the GPU requirements string. Specify the GPU resource reservation methods as follows:

- *value*/task
  Specifies GPUs per-task reservation. This is the equivalent of specifying PER_TASK for the METHOD parameter in the ReservationUsage section of the lsb.resources file.

- *value*/host
  Specifies GPUs per-host reservation of the specified resource. This is the equivalent of specifying PER_HOST for the METHOD parameter in the ReservationUsage section of the lsb.resources file.

For example, `GPU_REQ="num=2/task:mode=shared:j_exclusive=yes"`

Related tasks

- Configuring GPU resource requirements
- Submitting jobs that require GPU resources

Related reference

- **bsub -gpu** command option
- LSB_GPU_REQ parameter in the lsf.conf file
- GPU_REQ parameters in the lsb.applications file.
- GPU_REQ parameters in the lsb.queues file.

Support for NVIDIA DGX systems

LSF now supports NVIDIA DGX-1 and DGX-2 systems. LSF automatically detects and configures NVIDIA GPU support.

Job scheduling and execution

The following new features affect job scheduling and execution.

Specify time zones in time-based configurations

LSF now allows you to specify supported time zones in time-based configurations.

You can specify the time zone in any configuration file where you specify automatic time-based configurations with if-else constructs whenever you specify time windows, including the lsb.applications, lsb.hosts, lsb.params, lsb.queues, and lsb.resources files.

You can also specify time zones when specifying time windows with the RUN_WINDOW or DISPATCH_WINDOW parameters in the lsb.queues file. You can specify multiple time windows, but all time window entries must be consistent in whether they set the time zones. That is, either all entries must set a time zone, or all entries must not set a time zone.

Specifying the time zone is optional. If you do not specify a time zone, LSF uses the local system time zone. LSF supports all standard time zone abbreviations.

Note: If you specify the same abbreviation for multiple time zones, LSF might not select the correct time zone. A patch will be made available shortly after the release of Fix Pack 9 to address this issue.

Related concepts

- Automatic time-based configuration in the lsb.applications file

Related tasks

- Configure run windows

Related reference

- Automatic time-based configuration in the lsb.hosts file
- Automatic time-based configuration, RUN_WINDOW, and DISPATCH_WINDOW in the lsb.queues file.
- Automatic time-based configuration in the lsb.users file
- Automatic time-based configuration in the lsb.resources file

Related information

- Automatic time-based configuration in the lsb.params file

Application limits

LSF now allows you to limit the number of application instances and to enable resource limits on application profiles.

Specify application profiles on which limits are enforced with the new APPS and PER_APP parameters in the Limits section of the lsb.resources file.

The **blimits** command now displays resource limits for application profiles. To view resource limits for specific application profiles, use the new **blimits -A** command option.

Related reference

- **blimits** command

Related information

- APP and PER_APP parameters in the lsb.resources file

Container support

The following new feature affects LSF support for containers.

Queue level container parameters

You can now configure Docker, NVIDIA Docker, Shifter, and Singularity containers at the queue level (in addition to the application profile level) to run container jobs.

To enable containers at the queue level, configure the CONTAINER and EXEC_DRIVER parameters in the lsb.queues file in the same manner as you would specify a container application profile in the lsb.applications file.

Related concepts

- with Docker
- with Shifter
- with Singularity

Related reference

- CONTAINER, EXEC_DRIVER, and DOCKER_IMAGE_AFFINITY parameters in the lsb.queues file.

Docker image affinity

When scheduling Docker-based containerized jobs, you can now enable LSF to give preference for execution hosts that already have the requested Docker image. This reduces network bandwidth and the job start time because the execution host does not have to pull the Docker image from the repository and the job can immediately start on the execution host.

Enable or disable this feature by specifying the DOCKER_IMAGE_AFFINITY parameter in the lsb.applications, lsb.queues, or lsb.params file. You can also enable or disable this feature at the job level by specifying the

LSB_DOCKER_IMAGE_AFFINITY environment variable. The job level setting overrides the application level setting, which overrides the queue level setting, which overrides the cluster level setting.

When this feature is enabled, LSF considers Docker image location information when scheduling Docker jobs. Docker image affinity interacts with host preference and order[] string requests in the following manner:

- If host preference is specified, the host preference is honored first. Among hosts with the same preference level, hosts with the requested Docker image are given higher priority.
- If the order[] string is specified, the hosts with the requested Docker image have a higher priority first. Among hosts that all have the requested Docker image, the order[] string is then honored.

Related tasks

- Configuring to run Docker jobs
- Submitting Docker jobs to

Related reference

- DOCKER_IMAGE_AFFINITY parameter in the lsb.applications file.
- DOCKER_IMAGE_AFFINITY parameter in the lsb.params file.

Command output formatting

The following enhancements affect LSF command output formatting.

Improved access control levels for displaying job information

LSF now provides increased granularity of access control levels for displaying job information with the **bjobs**, **bhist**, and **bacct** commands.

Previously, you can prevent users from using the **bhist** and **bacct** commands to view detailed information for jobs that belong to other users by setting the SECURE_INFODIR_USER_ACCESS parameter to `Y` in the lsb.params file. You can also specify the granularity in which to display summary or detailed information with the **bjobs** command for jobs that belong to other users by specifying the SECURE_JOB_INFO_LEVEL and ENABLE_JOB_INFO_BY_ADMIN_ROLE parameters in the lsb.params file.

You can now specify the granularity of information that users can view with the **bhist** and **bacct** commands by setting SECURE_INFODIR_USER_ACCESS to `G`. The access to information is controlled by the SECURE_JOB_INFO_LEVEL and ENABLE_JOB_INFO_BY_ADMIN_ROLE parameters in the lsb.params file.

In addition, LSF now allows a higher level of granularity when specifying the access control level for displaying summary or detailed job information by providing a new level for the SECURE_JOB_INFO_LEVEL parameter in the lsb.params file. Specify `SECURE_JOB_INFO_LEVEL=5` (for the new level 5) to allow users to view summary information for all jobs, including jobs that belong to other users regardless of whether the other users belong to the same user group.

Related concepts

- Job information access control

- Setting job information access control

Related reference

- SECURE_INFODIR_USER_ACCESS parameter in the lsb.params file.
- SECURE_JOB_INFO_LEVEL parameter in the lsb.params file.

Customize user information output

Like the **bjobs -o** option, you can now also customize specific fields that the **busers** command displays by using the -o command option. This allows you to create a specific output format, allowing you to easily parse the information by using custom scripts or to display the information in a predefined format.

You can also specify the default output formatting of the **busers** command by specifying the LSB_BUSERS_FORMAT parameter in the lsf.conf file, or by specifying the LSB_BUSERS_FORMAT environment variable.

Related concepts

- Customize user information output

Related reference

- LSB_BUSERS_FORMAT parameter in the lsf.conf file

Related information

- **busers -o** command option

List of requested hosts in bjobs -o output

You can now use the **bjobs -o** option to show the list of requested hosts by specifying the `ask_hosts` field (in the **bjobs -o** command or the LSB_BJOBS_FORMAT parameter in the lsf.conf file). This list is the specific hosts that are requested with the **bsub -m** command option.

Related reference

- **bjobs -o** command option
- LSB_BJOBS_FORMAT parameter in the lsf.conf file
- **bsub -m** command option

Performance enhancements

The following enhancements affect performance.

File sanity check for LSF Data Manager jobs moved to the transfer job

The sanity check for the existence of files or folders and whether the user can access them, discovery of the size and modification time of the files or folders, and generation of the hash from the **bsub** and **bmod** commands is moved to the transfer job. This equalizes the performance of submitting and modifying jobs with and without data requirements. If needed, the new -datachk option can be used with the **bsub** or **bmod** command to perform full checking for jobs with a data requirement. The -datachk option can be specified only with the **-data** command. If the data requirement is for a tag, this option has no effect.

Regardless if -datachk is specified, the **bsub** and **bmod** commands no longer provide the file or folder size, modification time, and hash to the **mbatchd** daemon. This means that a new transfer job might need to be started for each file or folder that is requested for each new job with a data requirement.

Therefore, LSF now introduces a new lsf.datamanager configuration file parameter, CACHE_REFRESH_INTERVAL, to limit the number of transfer jobs. If multiple requests for the same file or folder come to LSF Data Manager within the configured interval in the parameter, only the first request results in a new transfer job. The assumption is that the requested file or folder has not changed during the configured interval. This also assumes that the requested file or folder was not cleaned from the staging area.

Related reference

- **bsub -data** command option
- **bsub -datachk** command option
- **bmod** command
- LSF_DATA_BSUB_CHKSUM parameter in the lsf.conf file
- lsf.datamanager Parameters section

LSF Data Manager transfer script directory

The LSF Data Manager transfer scripts are now located in the LSF_SERVERDIR directory. You can further modify these transfer scripts for your environment.

- dm_stagein_transfer.sh: Stage-in transfer script.
- dm_stagein_helper.sh: Helper script, which is invoked by the stage-in transfer script on the source host using **ssh**. The help script contains most of the operations that must be executed on the remote host, which reduces the number of times that **ssh** is used.
- dm_stageout_transfer.sh: The stage-out transfer script.

Previously, LSF Data Manager created the transfer scripts on demand.

Related information

- LSF_SERVERDIR parameter in the lsf.conf file.

Resource management

The following new features affect resource management and allocation.

Exclude swap threshold when enforcing job memory limits

When specifying the behavior of enforcing a job memory limit with the LSB_MEMLIMIT_ENF_CONTROL parameter in the lsf.conf file, you can now exclude the swap threshold and specify only the memory threshold.

To exclude the swap threshold, specify a value of `0` for the swap threshold in the LSB_MEMLIMIT_ENF_CONTROL parameter in the lsf.conf file:

LSB_MEMLIMIT_ENF_CONTROL=<Memory Threshold>:0:<Check Interval>:[all]

Related reference

- LSB_MEMLIMIT_ENF_CONTROL parameter in the lsf.conf file.

Set hard memory limits instead of per-process (soft) memory limits

When setting a memory limit for all the processes that belong to a job with the **bsub -M** command option, LSF sets a per-process soft memory limit by default. This means that when a job exceeds the memory limit, LSF passes the memory limit to the operating system. UNIX operating systems that support `RUSAGE_RSS` for the **setrlimit()** function can apply the memory limit to each process.

You can now disable this feature and force LSF to kill a job as soon as it exceeds the memory limit by adding an exclamation point (!) to the end of the memory limit that you specify with the **bsub -M** command option:

bsub -Mmemlimit[!]

If you specify the exclamation point, the memory limit is a hard limit, and LSF kills the job as soon as it exceeds this hard memory limit and does not wait for the host memory and swap threshold to be reached.

Related reference

- **bsub -M** command option
- **bmod -M** command option

Specify resource reservation method in the rusage string

You can now specify the resource reservation method at the job, application, or queue level by specifying the method in the resource usage (rusage) string of the resource requirements in the **bsub -R** option, or in the RES_REQ parameter in the lsb.applications or lsb.queues file. You can now also specify the GPU resource reservation method by specifying the method in the **bsub -gpu** option, in the GPU_REQ parameter in the lsb.applications or lsb.queues file, or in the LSB_GPU_REQ parameter in the lsf.conf file. Previously, you could only specify the resource reservation method at the global level by specifying the METHOD parameter in the ReservationUsage section of the lsb.resources file. The resource reservation value and method at the job level overrides the value and method at the application level, which overrides the value and method at the queue level, which overrides the value and method at the cluster level.

Specify the resource reservation method by using the /task, /job, or /host keyword after the numeric value in the rusage string or by using the /task or /host keyword in the GPU requirements string. You can only specify resource reservation methods for consumable resources. Specify the resource reservation methods as follows:

- *value*/task
  Specifies per-task reservation of the specified resource. This is the equivalent of specifying PER_TASK for the METHOD parameter in the ReservationUsage section of the lsb.resources file.

- *value*/job
  Specifies per-job reservation of the specified resource. This is the equivalent of specifying PER_JOB for the METHOD parameter in the ReservationUsage section of the lsb.resources file. You cannot specify per-job reservation in the GPU requirements string.

- *value*/host
  Specifies per-host reservation of the specified resource. This is the equivalent of specifying PER_HOST for the METHOD parameter in the ReservationUsage section of the lsb.resources file.

For example,

- `RES_REQ="rusage[mem=10/job:duration=10:decay=0]"`
- `RES_REQ="rusage[mem=(50 20)/task:duration=(10 5):decay=0]"`
- `GPU_REQ="num=2/task:mode=shared:j_exclusive=yes"`

Related concepts

- Usage string

Related tasks

- Configuring GPU resource requirements
- Submitting jobs that require GPU resources

Related reference

- **bsub -R** command option
- **bsub -gpu** command option
- LSB_GPU_REQ parameter in the lsf.conf file
- RES_REQ and GPU_REQ parameters in the lsb.applications file.
- RES_REQ and GPU_REQ parameters in the lsb.queues file.

Stripe tasks of a parallel job across free resources on candidate hosts

You can now enable LSF to stripe tasks of a parallel job across the free resources of the candidate hosts.

Enable job striping by using the stripe keyword in the span string of the resource requirements in the **bsub -R** option, or in the RES_REQ parameter in the lsb.applications or lsb.queues file. The span string at the job level overrides the span string at the application level, which overrides the span string at the queue level. You can limit the maximum number of tasks that are allocated to a candidate host by specifying a number for the stripe keyword.

For example,

- `span[stripe]`
- `span[stripe=2]`

Related concepts

- Span string

Related reference

- **bsub -R** command option
- RES_REQ parameter in the lsb.applications file.
- RES_REQ parameter in the lsb.queues file.

Data collection

The following new features affect IBM Spectrum LSF data collection.

Send email notification when job exits

Earlier versions of LSF allow you to enable email notification so that LSF sends an email notification whenever a job finishes. You can now enable LSF to send an email notification only when the job exits (that is, when the job is in Exit status). This ensures that you only receive an email notification when there is a job error. This feature only affects email that is sent by the **sbatchd** daemon.

To enable the job to send an email notification when the job exits, define `LSB_JOB_REPORT_MAIL=ERROR` in the lsf.conf file. You can also enable this feature at the job level by using the **bsub -Ne**, **brestart -Ne**, or **bmod -Ne** command options. You cannot use -Ne with the -N command option. Running the **bmod -Nn** command option cancels both the -N and -Ne options.

Related reference

- LSB_JOB_REPORT_MAIL parameter in the lsf.conf file
- **bsub -Ne** command option
- **bmod -Ne** command option
- **brestart -Ne** command option

Request notification for specific job states

You can now request that LSF notifies the job submission user when the job reaches any of the specified states (`EXIT`, `DONE`, `START`, or `SUSPEND`).

To enable this feature, use the **bsub -notify** command option or the LSF_AC_JOB_NOTIFICATION environment variable:

bsub -notify "[exit] [done] [start] [suspend]"

LSF_AC_JOB_NOTIFICATION="[exit] [done] [start] [suspend]"

Related reference

- **bsub -notify** command option
- **bmod -notify** command option
- LSF_AC_JOB_NOTIFICATION environment variable

Other changes to IBM Spectrum LSF

The following changes affect other aspects of LSF behavior.

Support for hosts with 3 TB memory

LSF now supports hosts that have more than 3 TB of memory.

What's new in IBM Spectrum LSF Version 10.1 Fix Pack 8

The following topics summarize the new and changed behavior in LSF 10.1 Fix Pack 8.

Release date: 4 June 2019

- **GPU enhancements**
  The following enhancements affect LSF GPU support.
- **Resource connector enhancements**
  The following enhancements affect LSF resource connector.
- **Resource management**
  The following new features affect resource management and allocation.
- **Job scheduling and execution**
  The following new features affect job scheduling and execution.
- **Container support**
  The following new feature affects LSF support for containers.
- **Data collection**
  The following new features affect IBM Spectrum LSF data collection.
- **Other changes to IBM Spectrum LSF**
  The following changes affect other aspects of LSF behavior.

GPU enhancements

The following enhancements affect LSF GPU support.

Relax GPU affinity while maintaining CPU affinity

LSF now allows you to relax GPU affinity while maintaining strict CPU affinity.

To relax GPU affinity for GPU jobs, specify `aff=no` in the GPU resource requirements string (that is, the **bsub -gpu** command option, LSB_GPU_REQ parameter in the lsf.conf file, and the GPU_REQ parameter in the lsb.queues and lsb.applications files).

By default, LSF maintains strict GPU affinity (that is, aff is set to `yes` by default).

If you specify both a gtile value and `aff=yes` in the GPU resource requirements string, strict GPU-CPU affinity binding is disabled. That is, LSF relaxes GPU-CPU affinity binding.

Note: The **bjobs** output does not show `aff=yes` even if you specify `aff=yes` in the **bsub -gpu** option. `LSB_GPU_NEW_SYNTAX=extend` must be defined in the lsf.conf file to relax GPU affinity.

Related concepts

- Example GPU job submissions

Related tasks

- Configuring GPU resource requirements
- Submitting jobs that require GPU resources

Related reference

- **bsub -gpu** command option
- LSB_GPU_NEW_SYNTAX parameter in the lsf.conf file
- LSB_GPU_REQ parameter in the lsf.conf file
- GPU_REQ parameter in the lsb.queues file
- GPU_REQ parameter in the lsb.applications file

Run multiple MPS daemons for GPU jobs

LSF now allows you to run multiple NVIDIA Multi-Process Service (MPS) daemons on a host for GPU jobs and allows you to share these MPS daemons across multiple GPU jobs.

To define the behavior of running and sharing multiple MPS daemons, new values are added to the existing `mps` keyword in the GPU resource requirements string (that is, the **bsub -gpu** command option, LSB_GPU_REQ parameter in the lsf.conf file, and the GPU_REQ parameter in the lsb.queues and lsb.applications files). These new values are per_socket and per_gpu.

mps=yes | no | per_socket | per_gpu

- LSF now allows you to start one MPS daemon per socket per job by setting `mps=per_socket` in the GPU resource requirements.
- LSF now allows you to start one MPS daemon per GPU per job by setting `mps=per_gpu` in the GPU resource requirements.

`LSB_GPU_NEW_SYNTAX=extend` must be defined in the lsf.conf file to enable MPS daemons.

Related concepts

- Example GPU job submissions

Related tasks

- Configuring GPU resource requirements
- Submitting jobs that require GPU resources

Related reference

- **bsub -gpu** command option
- LSB_GPU_NEW_SYNTAX parameter in the lsf.conf file
- LSB_GPU_REQ parameter in the lsf.conf file
- GPU_REQ parameter in the lsb.queues file
- GPU_REQ parameter in the lsb.applications file

NVIDIA Data Center GPU Manager (DCGM) integration updates

LSF, Version 10.1 Fix Pack 2 integrated with NVIDIA Data Center GPU Manager (DCGM) to work more effectively with GPUs in the LSF cluster. LSF now integrates with Version 1.4.6 of the NVIDIA Data Center GPU Manager (DCGM) API.

Enable the DCGM integration by defining the LSF_DCGM_PORT parameter in the lsf.conf file.

Related reference

- LSF_DCGM_PORT parameter in the lsf.conf file

Resource connector enhancements

The following enhancements affect LSF resource connector.

View resource connector information with the badmin command

You can now view LSF resource connector information with the **badmin** command. LSF now has the **badmin rc view** subcommand, which allows you to view LSF resource connector information, and the **badmin rc error** command, which allows you to view error messages from the host providers. To get the error messages, the third-party **mosquitto** message queue application must be running on the host.

Specify the maximum number of error messages that **badmin rc error** displays for each host provider by defining the LSB_RC_MQTT_ERROR_LIMIT parameter in the lsf.conf file.

Related tasks

- Use the badmin command to view resource connector information

Related reference

- LSB_RC_MQTT_ERROR_LIMIT parameter in the lsf.conf file
- **badmin rc** subcommand

Dynamic resource snapshot

LSF resource connector can now query any host provider API for dynamic resources at each snapshot interval. This allows LSF to natively handle problems with resource availability on resource providers.

In addition, if the **ebrokerd** daemon encounters certain provider errors while querying the host provider API for dynamic resources, LSF introduces a delay before **ebrokerd** repeats the request. Specify the new LSB_RC_TEMPLATE_REQUEST_DELAY parameter in the lsf.conf file to define this delay, in minutes.

Related reference

- LSB_RC_TEMPLATE_REQUEST_DELAY parameter in the lsf.conf file

Host management with instance IDs

LSF resource connector can now pass the instance ID of provisioned hosts to LSF when the hosts join the cluster. This provides an additional way for the resource connector hosts to identify themselves to the LSF cluster, which improves the redundancy and fault tolerance for the resource connector hosts.

In addition, the **bhosts -rc** and **bhosts -rconly** command options now display the instance IDs of the provisioned hosts.

Related reference

- **bhosts** command

Improved fault tolerance

The LSF resource connector now has improved fault tolerance by allowing `CLOSED_RC` hosts to be switched to `ok` more quickly, resulting in less wasted resources. Instance ID, cluster name, and template name are added to host local resources for more accurate information and better tracking. Instance ID is added to the `JOB_FINISH` event in the lsb.acct file.

Google provider synchronization

The LSF resource connector Google provider plugin can now synchronize hosts between LSF and the cloud.

Configure timeout values for each host provider

The LSF resource connector now allows you to configure timeout values for each host provider. Specify the provHostTimeOut parameter for each provider in the hostProviders.json file. The default value is 10 minutes. If a resource connector host does not join the LSF cluster within this timeout value, the host is relinquished.

Related tasks

- Configuring resource providers for resource connector

Related reference

- hostProviders.json file

Detect an NVIDIA sibling GPU under a PCI

LSF now enables **lim** and **elim.gpu.topology** to detect GPUs properly on hosts that have two sibling GPUs under one PCI if the first GPU is not an NVIDIA GPU but the second GPU is an NVIDIA GPU.

Note: For LSF resource connector on AWS, you must create an updated image, then use the new AMI ID to borrow the GPU instance from AWS.

Improved Azure support

The LSF resource connector now follows the official Azure documentation for HTTP/HTTPS proxies by calling the API from the Azure native library instead of the Java system library.

Candidate host group sorting

LSF now changes how the scheduler sorts candidate host groups when multiple templates are defined in LSF resource connector. The candidate host groups are now sorted based on template priority (previously, the order of these groups was undefined). LSF determines the template priority from the first host within the candidate host group.

Resource management

The following new features affect resource management and allocation.

Configure service classes with the bconf command

The **bconf** command now allows you create, delete, or modify service classes in the lsb.serviceclasses file.

Configure service classes by using the **serviceclass** keyword in the **bconf addmember**, **rmmember**, **update**, **create**, or **delete** subcommands.

Related tasks

- Configure service classes using the bconf command

Related reference

- **bconf** command

Specify SMT mode for IBM CSM jobs

You can now specify the SMT (simultaneous multithreading) mode for IBM Cluster Systems Manager (CSM) jobs

Specify the SMT mode at the job level with the new **bsub -smt** command option. Specify the SMT mode at the queue level by using the new smt keyword for the CSM_REQ parameter in the lsb.queues file.

If the smt value is specified in the CSM_REQ parameter of the lsb.queues file, that value overrides the **bsub -smt** command option. If the SMT value is not specified at any level, the default value is the first value of the CSM_VALID_SMT parameter in the lsb.params file.

Related tasks

- Configuring the CSM integration
- Submitting jobs to in easy mode
- Submitting jobs to in expert mode

Related reference

- **bsub -smt** command option
- CSM_REQ parameter in the lsb.queues file
- CSM_VALID_SMT parameter in the lsb.params file

Job scheduling and execution

The following new features affect job scheduling and execution.

Exclude fair share groups at job submission

If you belong to multiple fair share groups, LSF now allows you to exclude fair share groups from being associated with a job.

Run the **bsub -G** command option and use a tilde (~) to specify any fair share groups from which you want to exclude for the job. Use a space-separated list of tildes (~) to exclude multiple groups. If you exclude a middle node from a fair share tree, the descendant groups of that middle node are excluded along the specific path under the middle node.

For example, if there are two paths in the fair share tree to **`userA`**, which are /groupA/groupB/userA and /groupB/userA, and you specify **bsub -G "~groupA"**, **`userA`** can still be associated with the path /groupB/userA.

Related reference

- **bsub -G** command option

Pre-execution start time

LSF now adds pre-execution start time for jobs that have pre-execution processes. When the job is done, the **mbatchd** daemon records the start time in the `JOB_FINISH` event.

Container support

The following new feature affects LSF support for containers.

Run Docker images with an entry point

LSF now allows you to run Docker container jobs if the Docker image contains an entry point, but no command.

To submit a Docker job with a Docker entry point image, but no command, use the LSB_DOCKER_PLACE_HOLDER keyword instead of a command when submitting the job:

bsub -app docker_app_profile [options] LSB_DOCKER_PLACE_HOLDER
As for all Docker container jobs, you must use the -app option to specify an application profile that is configured to run Docker jobs. The Docker execution driver must be defined in this application profile in the lsb.applications file.

Related tasks

- Submitting Docker jobs to

Related reference

- LSB_DOCKER_PLACE_HOLDER keyword for the **bsub** command

Data collection

The following new features affect IBM Spectrum LSF data collection.

Use external scripts to check applications and to send notifications

LSF now includes the watchdog feature to regularly run external scripts that can check application data, logs, and other information, and to send notifications. If enabled, you can also use the watchdog feature to send these notifications to the LSF Application Center Notifications server.

Enable the watchdog profile for a particular application profile by specifying the WATCHDOG parameter in the lsb.applications file. Use the **bsub -app** option to submit jobs to application profiles with the WATCHDOG parameter enabled.

To enable LSF to send notifications to LSF Application Center, specify the URL and the listen port of the LSF Application Center Notifications server in the LSF_AC_PNC_URL parameter of the lsf.conf file.

In the external watchdog scripts, use the **bpost -N** command option to send a notification (with the message in the -d option and the specified error level) to the LSF Application Center Notifications server:

bpost -d "message" -N WARNING | ERROR | CRITICAL | INFO

Use the **bread -N** command option to see information on these notifications.

Related concepts

- Monitor applications by using external scripts

Related reference

- WATCHDOG parameter in the lsb.applications file
- LSF_AC_PNC_URL parameter in the lsf.conf file
- **bpost -N** command option
- **bread -N** command option

Other changes to IBM Spectrum LSF

The following changes affect other aspects of LSF behavior.

Maximum integer limit increased for resource limits

The maximum integer that you can specify for certain resource limits is increased from 32 bit (2^{31}) to 64 bit (2^{63}). This affects the following resource limits that have a valid value range of integers up to INFINIT_INT (which is defined in lsf.h):

- memory limit (**bsub -M** command option or the MEMLIMIT parameter in the lsb.queues or lsb.applications files)
- swap limit (**bsub -v** command option or the SWAPLIMIT parameter in the lsb.queues or lsb.applications files)
- core file size limit (**bsub -C** command option or the CORELIMIT parameter in the lsb.queues or lsb.applications files)
- stack limit (**bsub -S** command option or the STACKLIMIT parameter in the lsb.queues or lsb.applications files)
- data segment size limit (**bsub -D** command option for **malloc()**, or the DATALIMIT parameter in the lsb.queues or lsb.applications files)
- file size limit (**bsub -F** command option for **malloc()**, or the FILELIMIT parameter in the lsb.queues or lsb.applications files)

Related concepts

- configuration reference

bjobs -plan shows start and end times

The **bjobs -plan** command option now shows the planned start and end times for the job in the `PLAN_START_TIME` and `PLAN_FINISH_TIME` columns.

Related reference

- **bjobs -plan** command option

Improved bread and bstatus performance

LSF now improves the performance of the **bread** and **bstatus** commands by handing the **bread** and **bstatus** requests in the query **mbatchd** daemon, and by supporting multi-threaded querying when `LSB_QUERY_ENH=Y` is specified in the lsf.conf file. If `LSB_QUERY_ENH=N` is set or the **bread -a** command option is run (requiring data transfer), the main **mbatchd** daemon will fork a child to handle the query request.

Related reference

- **bread** command
- **bstatus** command
- LSB_QUERY_ENH parameter in the lsf.conf file
- **bjobs -plan** command option

New platform support

LSF supports the following platforms:

- RHEL 8.0, kernel 4.18.0, glibc 2.28

What's new in IBM Spectrum LSF Version 10.1 Fix Pack 7

The following topics summarize the new and changed behavior in LSF 10.1 Fix Pack 7.

Release date: 28 January 2019

- **GPU enhancements**
  The following enhancements affect LSF GPU support.
- **Resource connector enhancements**
  The following enhancements affect LSF resource connector.
- **Resource management**
  The following new features affect resource management and allocation.
- **Job scheduling and execution**
  The following new features affect job scheduling and execution.
- **Container support**
  The following new feature affects LSF support for containers.
- **Command output formatting**
  The following enhancements affect LSF command output formatting.

- **Security**
  The following new features affect cluster security.
- **Other changes to IBM Spectrum LSF**
  The following changes affect other aspects of LSF behavior.

GPU enhancements

The following enhancements affect LSF GPU support.

GPU fair share scheduling

LSF now allows you to consider GPU run time and GPU historical run time as weighting factors in the dynamic priority calculation for fair share scheduling for GPUs that are exclusively used.

To add GPU run time to the dynamic priority calculation, specify the GPU_RUN_TIME_FACTOR parameter in the lsb.params or lsb.queues file. To enable GPU historical run time to also be included in the GPU run time factor, set the ENABLE_GPU_HIST_RUN_TIME parameter to `Y` in the lsb.params or lsb.queues file. If these parameters are defined in both files, the queue-level values take precedence.

`LSB_GPU_NEW_SYNTAX=extend` must be defined in the lsf.conf file to include GPU run time in the fair share formula.

Related concepts

- GPU runtime fair share

Related reference

- GPU_RUN_TIME_FACTOR and ENABLE_GPU_HIST_RUN_TIME parameters in the lsb.queues file
- GPU_RUN_TIME_FACTOR parameter in the lsb.params file
- ENABLE_GPU_HIST_RUN_TIME parameter in the lsb.params file

GPU preemption

LSF GPU jobs now support preemptive scheduling so that a lower priority GPU job can release GPU resources for higher priority GPU jobs. GPU jobs must be either using `exclusive_process` mode or have `j_exclusive=yes` set to be preempted by other GPU jobs. Non-GPU jobs cannot preempt GPU jobs. In addition, higher priority GPU jobs can only preempt lower priority GPU jobs that are configured for automatic job migration and rerun (that is, the MIG parameter is defined and the RERUNNABLE parameter is set to `yes` in the lsb.queues or lsb.applications file).

To enable GPU preemption, configure the LSB_GPU_NEW_SYNTAX parameter in the lsf.conf file to either `Y` or `extend`, then configure the PREEMPTABLE_RESOURCES parameter in the lsb.params file to include the `ngpus_physical` resource. LSF treats the GPU resources the same as other preemptable resources.

Related concepts

- Preemptive scheduling

Related reference

- LSB_GPU_NEW_SYNTAX parameter in the lsf.conf file.

General GPU number limits

LSF now supports a general GPU number limit (that is, the number of ngpus_physical resources) when enabling the new GPU syntax.

For example, in the lsb.resources file, limit the general GPU number as follows:

```
Begin Limit
NAME = Limit1
RESOURCE=[ngpus_physical,3]
End Limit
```

To enable the general GPU number limit, enable the new GPU syntax. That is, configure the LSB_GPU_NEW_SYNTAX=extend parameter in the lsf.conf file.

Related reference

- LSB_GPU_NEW_SYNTAX parameter in the lsf.conf file.
- RESOURCE parameter in the lsb.resources file.

Resource connector enhancements

The following enhancements affect LSF resource connector.

Specifying GPU resources

LSF resource connector now allows you to create Google Compute Cloud instances with GPU resources. This allows you to run GPU jobs on Google Compute Cloud host providers.

Use the new gpuextend attribute to define the GPU topology on the template host for the Google Compute Cloud host provider. To define the GPU topology, edit the googleprov_templates.json file and specify the gpuextend attribute. This attribute is a string in the following format:

```
"key1=value1;key2=value2;..."
```

`LSB_GPU_NEW_SYNTAX=extend` must be defined in the lsf.conf file for the gpuextend attribute to take effect.

You can also define the specific type and number of GPUs for the instance. Specify the new gpuType attribute to define the type of GPU, and the new gpuNumber attribute to define the number of GPUs on the instance. LSF resource connector currently supports the following types of GPUs:

- `nvidia-tesla-k80`
- `nvidia-tesla-p100`
- `nvidia-tesla-p4`
- `nvidia-tesla-p100-vws`
- `nvidia-tesla-p4-vws`

Related reference

- googleprov_templates.json

Specifying jobs with CPU affinity requirements

In previous versions of LSF, affinity jobs did not generate demand (which triggers the LSF resource connector to create the required cloud instances) because the templates did not define CPU topology. LSF resource connector can now generate demand for affinity jobs, and when LSF resource connector provisions the cloud instances, affinity information is collected and cached. This means that the LSF scheduler can dispatch jobs to the cloud instances with affinity information that satisfies the job affinity resource requirements.

Related concepts

- Affinity string

Resource management

The following new features affect resource management and allocation.

Backfill direct data staging jobs that do not require file transfer

LSF now supports the backfill of direct data staging jobs that use the burst buffer but do not require LSF to handle the file transfer. That is, LSF manages the storage (burst buffer), but the job itself handles the file transfer. Since LSF is not handling the file transfer, LSF creates plans for the data staging jobs and reserves the necessary resources to prevent other jobs from using the resources. LSF considers these jobs to have no stage-in time, which allows the LSF scheduler to backfill the job to reserved slots.

Since the direct data staging job is handling the file transfer, LSF cannot reliably predict the storage usage during the job life cycle, so this feature includes configuration parameters in the lsf.conf file to provide limits on potential storage usage. You can now use the LSB_STAGE_STORAGE parameter to also specify a resource that reports the total storage space in addition to a resource that reports the available storage space. This prevents LSF from assigning more storage than is available because the resource information is out of date. In addition, the new LSB_STAGE_MAX_STAGE_IN parameter controls the maximum number of concurrent stage-in processes that run on a host.

Related tasks

- Configuring the CSM integration for direct data staging

Related reference

- LSB_STAGE_MAX_STAGE_IN parameter in the lsf.conf file
- LSB_STAGE_STORAGE parameter in the lsf.conf file

Guaranteed resource pool

A new parameter, ADMINISTRATORS is introduced in the GuaranteedResourcePool section of the lsb.resources file. Users can setup an individual administrator for each GuaranteedResourcePool. The administrator can then manage the corresponding resource pool by running the bconf command.

Related reference

- ADMINISTRATORS parameter under GuaranteedResourcePoolin the lsb.resources file.

Ineligible pending limit

A new parameter, INELIGIBLE is introduced in the LIMIT section of the lsb.resources file. Users can determine if they want the job to be in an ineligible pending state.

Related reference

- INELIGIBLE parameter under LIMIT in the lsb.resources file.

Best fit allocation for compute unit resource requirements

When specifying resource requirement strings using compute units, LSF can now use a best-fit allocation algorithm for job placement that considers the network topology of a cluster. This algorithm attempts to place the job in an allocation that spans the fewest possible number of compute units, while preferring to use compute units with fewer free slots to avoid fragmentation of the cluster. This algorithm also considers multiple levels of the network hierarchy when calculating the job allocation.

To use the best-fit allocation algorithm, use `bestfit` when specifying the compute unit scheduling preference in the compute unit resource requirement string:

```
bsub -R "cu[pref=bestfit]" command
```

Do not use `bestfit` with the `cu[balance]` keyword.

Related concepts

- Compute unit string
- Control job locality using compute units

Related tasks

- Submit a job with compute unit resource requirements

Job scheduling and execution

The following new features affect job scheduling and execution.

Parse job scripts with the bsub command

You can now load, parse, and run job scripts directly from the **bsub** command. Submit a job from the **bsub** command line with the job script as a command. The job script must be an ASCII text file and not a binary file, but it does not have to be an executable file.

In previous versions of LSF, you could run job scripts by using the < redirect to specify an ASCII text file that contains either Bourne shell command lines or Windows batch file command lines, or by writing the job fine one line at a time from the **bsub** interactive command line (by running **bsub** without specifying a command).

To enable this feature, define `LSB_BSUB_PARSE_SCRIPT=Y` in the lsf.conf file.

Related concepts

- Write job scripts

Related reference

- LSB_BSUB_PARSE_SCRIPT in lsf.conf
- **bsub** *job_script*

Fair share factors for absolute priority scheduling

LSF now has an updated method of calculating the fair share weight for absolute priority scheduling (**APS**). In addition, the **bqueues -r** and **-l** command options now display the normalized fair share factors in the `NORM_FS` column, if these factors are not zero.

Related reference

- **bqueues** command

Resource reservations for jobs with plans

If a job has a plan that is no longer valid (for example, because the host is down), LSF releases the resource reservations for the job, which means that the resources are now available for other jobs. This might result in other jobs being dispatched before the job with the plan, even if the other jobs have a lower priority.

LSF can now keep the resource reservations for jobs with plans, even if the plan is no longer valid, until LSF creates new plans based on updated resource availability. Enable this new behavior by specifying `LSB_PLAN_KEEP_RESERVE=Y` in the lsf.conf file.

Related reference

- LSB_PLAN_KEEP_RESERVE parameter in the lsf.conf file

Advance reservation enhancements

- A new parameter, AR_AVAILABLE_STATUS is introduced to the lsb.params file. If defined, this parameter defines which hosts are considered available for the creation of an advance reservation.

- Another parameter, AR_RUNLIMIT_CHECK is also introduced to the lsb.params file. If this parameter is enabled, a job with a predefined run limit is not allowed to dispatch if it is expected to run past the expiration time of the closed advance reservation.
- Another parameter, LSB_DISABLE_SUSPEND_AR_JOBS is introduced to the lsf.conf file. If this parameter is enabled, LSF keeps the job with an advance reservation running if the advance reservation is deleted.
- A new option -f has been added to the brsvadd parameter. This option considers the lim status of hosts when creating an advance reservation.
- A -f option is also added to the brsvdel parameter. This option force deletes an advance reservation.

Related reference

- AR_AVAILABLE_STATUS parameter in the lsb.params file.
- AR_RUNLIMIT_CHECK parameter in the lsb.params file.
- LSB_DISABLE_SUSPEND_AR_JOBS parameter in the lsb.conf file.
- brsvadd -f command
- brsvdel -f command

Container support

The following new feature affects LSF support for containers.

View information on Docker container images

LSF now has the new **bimages** command, which allows you to view information on LSF-invoked Docker container images.

You can also configure parameters for LSF to collect or expire Docker container image information. The LSF_IMAGE_INFO_PUBLISH_INTERVAL parameter specifies the interval for the **lim** process to fork a new process to collect host Docker container image information. The default interval is 60 seconds. The LSF_IMAGE_INFO_EXPIRE_INTERVAL parameter specifies how long the host image information is available in **mosquitto** before the information expires. The default time is 1 day. Specify both parameters in the lsf.conf file.

Related reference

- **bimages** command
- LSF_IMAGE_INFO_EXPIRE_INTERVAL parameter in the lsf.conf file
- LSF_IMAGE_INFO_PUBLISH_INTERVAL parameter in the lsf.conf file

Command output formatting

The following enhancements affect LSF command output formatting.

json format added to badmin perfmon view command

To support the graphing of performance metrics data in other LSF family applications (for example, IBM Spectrum LSF Application Center), a [-json] option has been added to the **badmin perfmon view** command.

Display allocated guarantee hosts with the bsla command

The **bsla** command can now display information on guarantee hosts that are being used (allocated) from each guarantee pool for the guarantee SLA. To enable the display of information on allocated guarantee hosts in the **bsla** command, configure `LSB_GSLA_DISPLAY_ALLOC_HOSTS=Y` in the lsf.conf file.

When enabled, the **bsla** command displays information on the allocated guarantee hosts in a new section (`USED GUARANTEE HOSTS`), and the hosts are organized by guarantee pool. **bsla** only displays this new section if there are jobs running in the SLA.

Related reference

- **bsla** command
- LSB_GSLA_DISPLAY_ALLOC_HOSTS parameter in the lsf.conf file

Customize host resource information output

Like the **bjobs -o** option, you can now also customize specific fields that the **lshosts** command displays by using the -o command option. This allows you to create a specific output format, allowing you to easily parse the information by using custom scripts or to display the information in a predefined format.

You can also specify the default output formatting of the **lshosts** command by specifying the LSF_LSHOSTS_FORMAT parameter in the lsf.conf file, or by specifying the LSF_LSHOSTS_FORMAT environment variable.

Related concepts

- Customize host resource information output

Related reference

- LSF_LSHOSTS_FORMAT parameter in the lsf.conf file

Related information

- **lshosts -o** command option

Add a reason when killing a job

A new option -C reason has been added to the bkill command. This gives the user the option to add a reason as to why the job is being killed.

Related reference

- **bkill -C reason** command

Total number of requested slots in bjobs -o output

You can now use the **bjobs -o** option to show the total number of requested slots for jobs by specifying the `nreq_slot` field (in the **bjobs -o** command or the LSB_BJOBS_FORMAT parameter in the lsf.conf file).

The total number of requested slots is usually the number of tasks that are requested with the **bsub -n** option. For certain job submissions, such as affinity jobs, exclusive jobs, jobs with alternative resource requirements, or jobs with compound resource requirements, the calculated number of slots is not as apparent. For pending jobs, specifying the `nreq_slot` output field allows you to view the calculated number of requested slots and can help determine why a job is pending.

Related reference

- **bjobs -o** command option
- LSB_BJOBS_FORMAT parameter in the lsf.conf file

GPU allocation information in bjobs -o output

You can now use the **bjobs -o** option to show GPU allocation information using the following output fields in the **bjobs -o** command or the LSB_BJOBS_FORMAT parameter in the lsf.conf file:

- `gpu_num`: The number of physical GPUs that the job is using. This corresponds with the `num` keyword in the GPU requirement string.
- `gpu_mode`: The GPU compute mode that the job is using (`shared` or `exclusive_process`). This corresponds with the `mode` keyword in the GPU requirement string.
- `gpu_alloc`: The job-based GPU allocation information.
- `j_exclusive`: Whether the job requested exclusive allocated GPUs (that is, if the GPUs cannot be shared with other jobs). This corresponds with the `j_exclusive` keyword in the GPU requirement string.

Related reference

- **bjobs -o** command option
- LSB_BJOBS_FORMAT parameter in the lsf.conf file

Security

The following new features affect cluster security.

User credential authorization

LSF has enhanced the security of authorizing user credentials. LSF uses an external authentication network to secure user credentials for the data stream between LSF clients and servers. LSF now has an enhanced version of the external authorization command **eauth** (named eauth.cve) that prevents users from changing the job user at the job submission time.

For more details on how to set up LSF hosts to secure the **eauth** command, refer to the *README for Fix 501633*.

Related reference

- LSF_EXT_SERVERDIR in the lsf.conf file.
- LSF_EAUTH_KEY in the lsf.sudoers file.
- LSF_ENV_OVERRIDE in the lsf.conf file.

Related information

- LSF_SERVERDIR in the lsf.conf file.

Other changes to IBM Spectrum LSF

The following changes affect other aspects of LSF behavior.

Update dynamic host group information

You can now specify the time interval for which LSF automatically updates both dynamic host group information in the lsb.hosts file and dynamic user group information in the lsb.users file automatically without running the **badmin reconfig** command. Specify the time interval, in hours, by defining the EGROUP_UPDATE_INTERVAL parameter in the lsb.params file. You can also specify the time interval in minutes by specifying the keyword `m` after the time interval.

Previously, this parameter only specified the time interval, in hours, for LSF to update dynamic user group information in the lsb.users file.

Related reference

- EGROUP_UPDATE_INTERVAL parameter in the lsb.params file.

JOBS_PER_SCHED_CYCLE parameter name change

The JOBS_PER_SCHED_CYCLE parameter in the Limit section of the lsb.resources file is changed to ELIGIBLE_PEND_JOBS. This is because LSF only considers a certain number of jobs as being eligible for scheduling.

Related reference

- lsb.resources file

Update Docker Image

A new parameter, LSB_UPDATE_CONTAINER_IMAGE is introduced in the lsf.conf file. If this parameter is defined, LSF will update the docker image before execution.

Related reference

- LSB_UPDATE_CONTAINER_IMAGE parameter in the lsf.conf file.

Error handling for LSF jobs with IBM Cluster Systems Manager

When running LSF jobs with the IBM Cluster Systems Manager (CSM) integration, LSF can now take specific actions based on CSM error codes. In addition, CSM error codes are now included in the output for the **bjobs** and **bhist** commands.

Related reference

- lsb.resources file

Kill jobs that run over the defined RUNLIMIT

LSF now allows the **mbatchd** daemon to kill jobs that are running over the defined RUNLIMIT value for a long period of time. Enable this behavior, by defining `KILL_JOBS_OVER_RUNLIMIT=Y` in the lsb.params file.

Related reference

- KILL_JOBS_OVER_RUNLIMIT parameter in the lsb.params file.

Display the exit reason in the bjobs -l -pac command

LSF now displays the exit reason for exited jobs in the **bjobs -l -pac** command. The exit reason is displayed in the `EXIT_REASON:` field.

New platform support

LSF supports the following platforms:

- SLES 15, kernel 4.12.14, glibc 2.26

What's new in IBM Spectrum LSF Version 10.1 Fix Pack 6

The following topics summarize the new and changed behavior in LSF 10.1 Fix Pack 6.

Release date: 13 June 2018

- **GPU enhancements**
  The following enhancements affect LSF GPU support.
- **Data collection**
  The following new features affect IBM Spectrum LSF data collection.
- **Resource connector enhancements**
  The following enhancements affect LSF resource connector.

- **Resource management**
  The following new features affect resource management and allocation.
- **Job scheduling and execution**
  The following new features affect LSF job scheduling and execution.
- **Command output formatting**
  The following enhancements affect LSF command output formatting.
- **Other changes to IBM Spectrum LSF**
  The following changes affect other aspects of LSF behavior.

GPU enhancements

The following enhancements affect LSF GPU support.

GPU autoconfiguration

Enabling GPU detection for LSF is now available with automatic configuration. To enable automatic GPU configuration, configure `LSF_GPU_AUTOCONFIG=Y` in the lsf.conf file.

When enabled, the **lsload -gpu**, **lsload -gpuload**, and **lshosts -gpu** commands will show host-based or GPU-based resource metrics for monitoring.

Related concepts

- Automatic GPU configuration
- Monitor GPU resources with lsload command
- Monitor GPU resources with lshosts command

Related reference

- LSF_GPU_AUTOCONFIG
- LSF_HOST_MAX_GPU

Specify additional GPU resource requirements

LSF now allows you to request additional GPU resource requirements to allow you to further refine the GPU resources that are allocated to your jobs. The existing **bsub -gpu** command option, LSB_GPU_REQ parameter in the lsf.conf file, and the GPU_REQ parameter in the lsb.queues and lsb.applications files now have additional GPU options to make the following requests:

- The gmodel option requests GPUs with a specific brand name, model number, or total GPU memory.
- The gtile option specifies the number of GPUs to use per socket.
- The gmem option reserves the specified amount of memory on each GPU that the job requires.
- The nvlink option requests GPUs with NVLink connections.

You can also use these options in the **bsub -R** command option or RES_REQ parameter in the lsb.queues and lsb.applications files for complex GPU resource requirements, such as for compound or alternative resource requirements. Use the gtile option in the span[] string and the other options (gmodel, gmem, and nvlink) in the rusage[] string as constraints on the ngpus_physical resource.

To specify these new GPU options, specify `LSB_GPU_NEW_SYNTAX=extend` in the lsf.conf file.

Related concepts

- Example GPU job submissions

Related tasks

- Configuring GPU resource requirements
- Submitting jobs that require GPU resources

Related reference

- **bsub -gpu** command option
- LSB_GPU_NEW_SYNTAX parameter in the lsf.conf file
- LSB_GPU_REQ parameter in the lsf.conf file
- GPU_REQ parameter in the lsb.queues file
- GPU_REQ parameter in the lsb.applications file

Data collection

The following new features affect IBM Spectrum LSF data collection.

IBM Spectrum Scale disk I/O accounting using Elasticsearch

LSF now uses IBM Spectrum LSF Explorer (LSF Explorer) to collect IBM Spectrum Scale disk I/O accounting data which, when combined with LSF job information, allows LSF to provide job-level IBM Spectrum Scale I/O statistics. To use this feature, LSF Explorer must be deployed in your LSF cluster, and LSF must be using IBM Spectrum Scale as the file system. To enable IBM Spectrum Scale disk I/O accounting, configure `LSF_QUERY_ES_FUNCTIONS="gpfsio"` (or `LSF_QUERY_ES_FUNCTIONS="all"`) and `LSF_QUERY_ES_SERVERS="`*`ip`*`:`*`port`*`"` in the lsf.conf file.

Use the following commands to display IBM Spectrum Scale disk I/O accounting information:

- **bacct -l** displays the total number of read/write bytes of all storage pools on IBM Spectrum Scale.
- **bjobs -l** displays the accumulated job disk usage (I/O) data on IBM Spectrum Scale.
- **bjobs -o "gpfsio"** displays the job-level disk usage (I/O) data on IBM Spectrum Scale.

Related concepts

- Use to improve the performance of the bacct and bhist commands, or to retrieve additional data

Related reference

- LSF_QUERY_ES_SERVERS parameter in the lsf.conf file
- LSF_QUERY_ES_FUNCTIONS parameter in the lsf.conf file
- **bacct -l** command option

- **bjobs -l** command option
- **bjobs -o** command option

Resource connector enhancements

The following enhancements affect LSF resource connector.

LSF resource connector auditing

With this release, LSF will log resource connector VM events along with usage information into a new file called rc.audit.x (one log entry per line in JSON format). The purpose of the rc.audit.x log file is to provide evidence to support auditing and usage accounting as supplementary data to third party cloud provider logs. The information is readable by the end user as text and is hash protected for security.

LSF also provides a new command-line tool **rclogsvalidate** to validate the logs described above. If the audit file is tampered with, the tool will identify the line which was modified and incorrect.

New parameters have been added to LSF in the lsf.conf configuration file:

- LSF_ RC_AUDIT_LOG: If set to Y, enables the resource connector auditor to generate log files.
- RC_MAX_AUDIT_LOG_SIZE: An integer to determine the maximum size of the rc.audit.x log file, in MB.
- RC_MAX_AUDIT_LOG_KEEP_TIME: An integer that specifies the amount of time that the resource connector audit logs are kept, in months.

Resource connector template prioritizing

In 10.1 Fix Pack 6, resource connector prioritizes templates.

The ability to set priorities is now provided in the resource connector template. LSF will use higher priority templates first (for example, less expensive templates should be assigned higher priorities).

LSF sorts candidate template hosts by template name. However, an administrator might want to sort them by priority, so LSF favors one template to the other. The “Priority” attribute has been added.:

```
{
      "Name": "T2",
      "MaxNumber": "2",
      "Attributes":
      {
        "type": ["String", "X86_64"],
        "ncpus": ["Numeric", "1"],
        "mem": ["Numeric", "512"],
        "template": ["String", "T2"],
        "ostkhost": ["Boolean", "1"]
      },
      "Image": "LSF10.1.0.3_OSTK_SLAVE_VM",
      "Flavor": "t2.nano",
      "UserData": "template=T2",
      "Priority": "10"
    }
```

Note: **The example above is for a template in openStack. Other templates may not contain all attributes.**

The default value of Priority is “0”, which means the lowest priority. If template hosts have the same priority, LSF sorts them by template name.

Support for a dedicated instance of AWS

One new parameter is added to the resource connector template to support a dedicated instance of AWS.

If you do not have a placement group in your AWS account, you must at least insert a placement group with a blank name inside quotation marks, because this is required to specify the tenancy. If you have a placement group, specify the placement group name inside the quotation marks. For example, `"placementGroupName": "",` or `"placementGroupName": "hostgroupA",`.

The values for tenancy can be "default", "dedicated", and "host". However, LSF currently only supports "default" and "dedicated".

The above can be applied for both on-demand and spot instances of AWS.

Full example the template file is as follows:

```
{
    "templates": [
         {
            "templateId": "aws-vm-0",
            "maxNumber": 5,
            "attributes": {
                "type": ["String", "X86_64"],
                "ncores": ["Numeric", "1"],
                "ncpus": ["Numeric", "1"],
                "mem": ["Numeric", "512"],
                "awshost": ["Boolean", "1"],
                "zone": ["String", "us_west_2d"]
            },
            "imageId": "ami-0db70175",
            "subnetId": "subnet-cc0248ba",
            "vmType": "c4.xlarge",
            "keyName": "martin",
            "securityGroupIds": ["sg-b35182ca"],
            "instanceTags": "Name=aws-vm-0",
            "ebsOptimized" : false,
            "placementGroupName": "",
            "tenancy": "dedicated",
            "userData": "zone=us_west_2d"         }
}
```

HTTP proxy server capability for LSF resource connector

This feature is useful for customers with strict security requirements. It allows for the use of an HTTP proxy server for endpoint access.
Note: For this release, this feature is enabled only for AWS.
This feature introduces the parameter "scriptOption" for the provider. For example:

```
{
    "providers":[
        {
            "name": "aws1",
            "type": "awsProv",
            "confPath": "resource_connector/aws",
            "scriptPath": "resource_connector/aws",
           "scriptOption": "-Dhttps.proxyHost=10.115.206.146 -
```

```
Dhttps.proxyPort=8888"
        }
    ]
}
```

The value of `scriptOption` can be any string and is not verified by LSF.

LSF sets the environment variable `SCRIPT_OPTIONS` when launching the scripts. For AWS plugins, the information is passed to java through syntax like the following:

```
java $SCRIPT_OPTIONS -Daws-home-dir=$homeDir -jar $homeDir/lib/AwsTool.jar --
getAvailableMachines $homeDir $inJson
```

Create EBS-Optimized instances

Creating instances with EBS-Optimized enabled is introduced in this release to archive better performance in cloud storage.

The EBS-Optimized attribute has been added to the resource connector template. The AWS provider plugin passes the information to AWS when creating the instance. Only high-end instance types support this attribute. The resource connector provider plugin will not check if the instance type is supported.

The "ebsOptimized" field in the resource connector template is a boolean value (either true or false). The default value is false. Specify the appropriate vmType that supports ebs_optimized (consult AWS documentation).

```
{
    "templates": [
        {
            "templateId": "Template-VM-1",
            "maxNumber": 4,
            "attributes": {
                "type": ["String", "X86_64"],
                "ncores": ["Numeric", "1"],
                "ncpus": ["Numeric", "1"],
                "mem": ["Numeric", "1024"],
                "awshost1": ["Boolean", "1"]
            },
            "imageId": "ami-40a8cb20",
           "vmType": "m4.large",
            "subnetId": "subnet-cc0248ba",
            "keyName": "martin",
            "securityGroupIds": ["sg-b35182ca"],
            "instanceTags" : "group=project1",
            "ebsOptimized" : true,
            "userData": "zone=us_west_2a"
        }
    ]
}
```

Resource connector policy enhancement

Enhancements have been made for administration of resource connector policies:

- A clusterwide parameter RC_MAX_REQUESTS has been introduced in the lsb.params file to control the maximum number of new instances that can be required or requested.
  After adding allocated usable hosts in previous sessions, LSF generates total demand requirement. An internal policy entry is created as below:

```
{
      "Name": "__RC_MAX_REQUESTS",
      "Consumer":
       {
        "rcAccount": ["all"],
        "templateName": ["all"],
        "provider": ["all"]
       },
      "StepValue": "$val:0"
    }
```

- The parameter LSB_RC_UPDATE_INTERVAL controls how frequent LSF starts demand evaluation. Combining with the new parameter, it plays a cluster wide “step” to control the speed of cluster grow.

Resource management

The following new features affect resource management and allocation.

Running LSF jobs with IBM Cluster Systems Manager

LSF now allows you to run jobs with IBM Cluster Systems Manager (CSM).

The CSM integration allows you to run LSF jobs with CSM features.

Related concepts

- with IBM Cluster Systems Manager

Direct data staging

LSF now allows you to run direct data staging jobs, which uses a burst buffer (for example, IBM CAST burst buffer) instead of the cache to stage in and stage out data for data jobs.

Use the CSM integration to configure LSF to run burst buffer data staging jobs.

Related concepts

- Direct data staging jobs with CSM

Job scheduling and execution

The following new features affect LSF job scheduling and execution.

Plan-based scheduling and reservations

When enabled, LSF's plan-based scheduling makes allocation plans for jobs based on anticipated future cluster states. LSF reserves resources as needed in order to carry out its plan. This helps to avoid starvation of

jobs with special resource requirements.

Plan-based scheduling and reservations addresses a number of issues with the older reservation features in LSF. For example:

- It ensures that reserved resources can really be used by the reserving jobs
- It has better job start-time prediction for reserving jobs, and thus better backfill decisions

Plan-based scheduling aims to replace legacy LSF reservation policies. When ALLOCATION_PLANNER is enabled in the lsb.params configuration file, then parameters related to the old reservation features (that is SLOT_RESERVE and RESOURCE_RESERVE in lsb.queues), are ignored with a warning.

Related reference

- ALLOCATION_PLANNER in lsb.params
- ESTIMATED_RUNTIME in lsb.params
- PLAN in lsb.params
- **bjobs -plan**
- **bjobs -l** and **bjobs -l -plan**

Automatically extend job run limits

You can now configure the LSF allocation planner to extend the run limit for jobs when the resources that are occupied by the job are not needed by other jobs in queues with the same or higher priority. The allocation planner looks at job plans to determine if there are any other jobs that require the current job's resources.

Enable extendable run limits for jobs submitted to a queue by specifying the EXTENDABLE_RUNLIMIT parameter in the lsb.queues file. Since the allocation planner decides whether the extend the run limit of jobs, you must also enable plan-based scheduling by enabling the ALLOCATION_PLANNER parameter in the lsb.params file.

Related tasks

- Configuring extendable run limits

Related reference

- ALLOCATION_PLANNER parameter in lsb.params
- EXTENDABLE_RUNLIMIT parameter in the lsb.queues file

Default epsub executable files

Similar to **esub** programs, LSF now allows you to define a default **epsub** program that runs even if you do not define mandatory **epsub** programs with the LSB_ESUB_METHOD parameter in the lsf.conf file. To define a default **epsub** program, create an executable file named epsub (with no application name in the file name) in the LSF_SERVERDIR directory.

After the job is submitted, LSF runs the default **epsub** executable file if it exists in the LSF_SERVERDIR directory, followed by any mandatory **epsub** executable files that are defined by LSB_ESUB_METHOD, followed by the **epsub** executable files that are specified by the -a option.

Related concepts

- External job submission and execution controls

Restrict users and user groups from forwarding jobs to remote clusters

You can now specify a list of users or user groups that can forward jobs to remote clusters when using the LSF multicluster capability. This allows you to prevent jobs from certain users or user groups from being forwarded to an execution cluster, and to set limits on the submission cluster.

These limits are defined at the queue level in LSF. For jobs that are intended to be forwarded to a remote cluster, users must submit these jobs to queues that have the SNDJOBS_TO parameter configured in the lsb.queues file. To restrict these queues to specific users or user groups, define the FWD_USERS parameter in the lsb.queues file for these queues.

Related tasks

- Enable multicluster queues

Related reference

- FWD_USERS parameter in the lsb.queues file

Advance reservations now support the "same" section in resource requirement strings

When using the **brsvadd -R** and **brsvmod -R** options to specify resource requirements for advance reservations, the same string now takes effect, in addition to the select string. Previous versions of LSF only allowed the select string to take effect.

This addition allows you to select hosts with the same resources for your advance reservation.

Related concepts

- Same string

Related tasks

- Adding an advance reservation

Related reference

- **brsvadd -R** command option
- **brsvmod -R** command option

Priority factors for absolute priority scheduling

You can now set additional priority factors for LSF to calculate the job priority for absolute priority scheduling (APS). These additional priority factors allow you to modify the priority for the application profile, submission user, or user group, which are all used as factors in the APS calculation. You can also view the APS and fair share user priority values for pending jobs.

To set the priority factor for an application profile, define the PRIORITY parameter in the lsb.applications file. To set the priority factor for a user or user group, define the PRIORITY parameter in the User or UserGroup section of the lsb.users file.

The new **bjobs -prio** option displays the APS and fair share user priority values for all pending jobs. In addition, the **busers** and **bugroup** commands display the APS priority factor for the specified users or user groups.

Related concepts

- Absolute priority scheduling (APS)

Related reference

- PRIORITY parameter in the lsb.applications file
- PRIORITY parameter in the User or UserGroup section of the lsb.users file
- **bjobs -prio** command option
- **busers** command
- **bugroup** command

Job dispatch limits for users, user groups, and queues

You can now set limits on the maximum number of jobs that are dispatched in a scheduling cycle for users, user groups, and queues. This allows you to control the number of jobs, by user, user group, or queue, that are dispatched for execution. If the number of dispatched jobs reaches this limit, other pending jobs that belong to that user, user group, or queue that might have dispatched will remain pending for this scheduling cycle.

To set or update the job dispatch limit, run the **bconf** command on the limit object (that is, run `bconf` *`action_type`* `limit=`*`limit_name`*) to define the JOBS_PER_SCHED_CYCLE parameter for the specific limit. You can only set job dispatch limits if the limit consumer types are USERS, PER_USER, QUEUES, or PER_QUEUE.

For example, `bconf update limit=L1 "JOBS_PER_SCHED_CYCLE=10"`

You can also define the job dispatch limit by defining the JOBS_PER_SCHED_CYCLE parameter in the Limit section of the lsb.resources file.

Related reference

- Configuring resource allocation limits
- **bconf** command
- JOBS_PER_SCHED_CYCLE parameter in the lsb.resources file

Command output formatting

The following enhancements affect LSF command output formatting.

blimits -a option shows all resource limits

The new **blimits -a** command option shows all resource allocation limits, even if they are not being applied to running jobs. Normally, running the **blimits** command with no options displays only resource allocation limits that are being applied to running jobs.

Related concepts

- Display resource allocation limits
- View information about resource allocation limits

Related reference

- **blimits -a** command option

Use bread -w to show messages and attached data files in wide format

LSF allows you to read messages and attached data files from a job in wide format with the new **bread -w** command option. The wide format displays information without truncating fields.

Related reference

- **bread -w** command option

Other changes to IBM Spectrum LSF

The following changes affect other aspects of LSF behavior.

lsportcheck utility

A new **lsportcheck** utility has been added to LSF. This utility can be used to check the required ports for LSF and include detailed information, whether it is being used or not.

The lsportcheck utility only checks ports on the host for availability. It discovers the ports by reading the configuration files. If the line is commented out or if there is no value, it will use the default values.

The lsportcheck utility must be executed by the root user, since the tool uses 'netstat' and needs root to get the complete information on the ports of the OS.

Before running this tool, you must source the profile or set the environment variable LSF_TOP.

The utility is installed at <LSF_TOP>/<VERSION>/<PLATFORM>/bin/, for example, /opt/lsf/10.1/linux2.6-glibc2.3-x86_64/bin/

Usage:

lsportcheck

lsportcheck -h

lsportcheck -l[-m | -s]

Description:

Without arguments will output command usage and exit.

-h Output command usage and exit.

-l List TCP and UDP ports on master.

-l -m List TCP and UDP ports on master.

-l -s List TCP and UDP ports on slave.

Note: lsportcheck can only be run by root.
Source the relative IBM Spectrum LSF shell script after installation:

For csh or tcsh: 'source $LSF_ENVDIR/cshrc.lsf'

For sh, ksh, or bash: 'source $LSF_ENVDIR/profile.lsf'

Example output:

Example of the output using command **lsportcheck -l** or **lsportcheck -l -m** on **LSF master:**

```
Checking ports required on host [mymaster1]
-------------------------------------------------------------------
Program Name  Port Number  Protocol  Binding Address  PID/Status
-------------------------------------------------------------------
lim           7869         TCP       0.0.0.0          1847
lim           7869         UDP       0.0.0.0          1847
res           6878         TCP       0.0.0.0          1881
sbatchd       6882         TCP       0.0.0.0          1890
mbatchd       6881         TCP       0.0.0.0          1921
mbatchd       6891         TCP       0.0.0.0          1921
pem           7871         TCP       0.0.0.0          1879
vemkd         7870         TCP       0.0.0.0          1880
egosc         7872         TCP       0.0.0.0          3226
-------------------------------------------------------------------
Optional ports:
-------------------------------------------------------------------
wsgserver     9090         TCP       0.0.0.0          [Not used]
named         53           TCP       0.0.0.0          [Not used]
named         53           UDP       0.0.0.0          [Not used]
named         953          TCP       0.0.0.0          [In use by another program]
```

Example output:

Example of the output using command **lsportcheck -l -s** on **LSF slave:**

```
Checking ports required on host [host1]
------------------------------------------------------------------
Program Name   Port Number   Protocol   Binding Address   PID/Status
------------------------------------------------------------------
lim            7869          TCP        0.0.0.0           1847
lim            7869          UDP        0.0.0.0           1847
res            6878          TCP        0.0.0.0           1881
sbatchd        6882          TCP        0.0.0.0           1890
pem            7871          TCP        0.0.0.0           1879
```

Increased project name size

In previous versions of LSF, when submitting a job with a project name (by using the **bsub -P** option, the DEFAULT_PROJECT parameter in the lsb.params file, or by using the LSB_PROJECT_NAME or LSB_DEFAULTPROJECT environment variables), the maximum length of the project name was 59 characters. The maximum length of the project name is now increased to 511 characters.

This increase also applies to each project name that is specified in the PER_PROJECT and PROJECTS parameters in the lsb.resources file.

Cluster-wide DNS host cache

LSF can generate a cluster-wide DNS host cache file ($LSF_ENVDIR/.hosts.dnscache) that is used by all daemons on each host in the cluster to reduce the number of times that LSF daemons directly call the DNS server when starting the LSF cluster. To enable the cluster-wide DNS host cache file, configure `LSF_DNS_CACHE=Y` in the lsf.conf file.

Use #include for shared configuration file content

In previous versions of LSF, you can use the `#INCLUDE` directive to insert the contents of a specified file into the beginning of the lsf.shared or lsb.applications configuration files to share common configurations between clusters or hosts.

You can now use the `#INCLUDE` directive in any place in the following configuration files:

- lsb.applications
- lsb.hosts
- lsb.queues
- lsb.reasons
- lsb.resources
- lsb.users

You can use the `#INCLUDE` directive only at the beginning of the following file:

- lsf.shared

For example, you can use `#if` ... `#endif` Statements to specify a time-based configuration that uses different configurations for different times. You can change the configuration for the entire system by modifying the common file that is specified in the `#INCLUDE` directive.

Related tasks

- Shared configuration file content

Related reference

- #INCLUDE directive in the lsb.applications file
- #INCLUDE directive in the lsb.hosts file
- #INCLUDE directive in the lsb.queues file
- #INCLUDE directive in the lsb.reasons file
- #INCLUDE directive in the lsb.resources file
- #INCLUDE directive in the lsb.users file

Showing the pending reason for interactive jobs

The **bsub -I** command now displays the pending reason for interactive jobs, based on the setting of LSB_BJOBS_PENDREASON_LEVEL, if the job is pending.

Related reference

- **bsub -I** command option
- **bsub -Ip** command option
- **bsub -IS** command option
- **bsub -ISp** command option
- **bsub -ISs** command option
- **bsub -Is** command option
- **bsub -IX** command option
- LSB_BJOBS_PENDREASON_LEVEL parameter in the lsf.conf file

Showing warning messages for interactive jobs

Interactive jobs can now show exit reasons when the jobs are killed (due to conditions such as reaching the memory or runtime limit). The exit reason is the same as the message shown for the output of the **bhist -l** and **bjobs -l** commands.

Related reference

- **bsub -I** command option
- **bsub -Ip** command option
- **bsub -IS** command option
- **bsub -ISp** command option
- **bsub -ISs** command option
- **bsub -Is** command option
- **bsub -IX** command option

Changing job priorities and limits dynamically

Through the introduction of two new parameters, LSF now supports changing job priorities and limits dynamically through an import file. This includes:

- Calling the eadmin script at a configured interval, even when a job exception has not occurred through the parameter EADMIN_TRIGGER_INTERVAL in the lsb.params file.

- Allowing job submission during a policy update or cluster restart through the parameter PERSIST_LIVE_CONFIG in the lsb.params file.
- Enhancement of the **bconf** command to override existing settings through the set action, to support the -pack option for reading multiple requests from a file.

Related reference

- EADMIN_TRIGGER_INTERVAL in the lsb.params file
- PERSIST_LIVE_CONFIG in the lsb.params file
- **bconf** command

Specify a UDP port range for LSF daemons

You can now specify a range of UDP ports to be used by LSF daemons. Previously, LSF binds to a random port number between 1024 and 65535.

To specify a UDP port range, define the LSF_UDP_PORT_RANGE parameter in the lsf.conf file. Include at least 10 ports in this range, and you can specify integers between 1024 and 65535.

Related reference

- LSF_UDP_PORT_RANGE in the lsf.conf file

What's new in IBM Spectrum LSF Version 10.1 Fix Pack 5

The following topics summarize the new and changed behavior in LSF 10.1 Fix Pack 5. This Fix Pack applies only to IBM POWER9 platforms.

Release date: 18 May 2018

- **Resource management**
  The following new features affect resource management and allocation.
- **Job scheduling and execution**
  The following new features affect LSF job scheduling and execution.
- **Command output formatting**
  The following enhancements affect LSF command output formatting.
- **Other changes to IBM Spectrum LSF**
  The following changes affect other aspects of LSF behavior.

Resource management

The following new features affect resource management and allocation.

Note: LSF 10.1 Fix Pack 5 applies only to IBM POWER9 platforms.

Running LSF jobs with IBM Cluster Systems Manager

LSF now allows you to run jobs with IBM Cluster Systems Manager (CSM).

The CSM integration allows you to run LSF jobs with CSM features.

Related concepts

- with IBM Cluster Systems Manager

Direct data staging

LSF now allows you to run direct data staging jobs, which uses a burst buffer (for example, IBM CAST burst buffer) instead of the cache to stage in and stage out data for data jobs.

Use the CSM integration to configure LSF to run burst buffer data staging jobs.

Related concepts

- Direct data staging jobs with CSM

Job scheduling and execution

The following new features affect LSF job scheduling and execution.

Note: LSF 10.1 Fix Pack 5 applies only to IBM POWER9 platforms.

Plan-based scheduling and reservations

When enabled, LSF's plan-based scheduling makes allocation plans for jobs based on anticipated future cluster states. LSF reserves resources as needed in order to carry out its plan. This helps to avoid starvation of jobs with special resource requirements.

Plan-based scheduling and reservations addresses a number of issues with the older reservation features in LSF. For example:

- It ensures that reserved resources can really be used by the reserving jobs
- It has better job start-time prediction for reserving jobs, and thus better backfill decisions

Plan-based scheduling aims to replace legacy LSF reservation policies. When ALLOCATION_PLANNER is enabled in the lsb.params configuration file, then parameters related to the old reservation features (that is SLOT_RESERVE and RESOURCE_RESERVE in lsb.queues), are ignored with a warning.

Related reference

- ALLOCATION_PLANNER in lsb.params
- ESTIMATED_RUNTIME in lsb.params
- PLAN in lsb.params

- **bjobs -plan**
- **bjobs -l** and **bjobs -l -plan**

Automatically extend job run limits

You can now configure the LSF allocation planner to extend the run limit for jobs when the resources that are occupied by the job are not needed by other jobs in queues with the same or higher priority. The allocation planner looks at job plans to determine if there are any other jobs that require the current job's resources.

Enable extendable run limits for jobs submitted to a queue by specifying the EXTENDABLE_RUNLIMIT parameter in the lsb.queues file. Since the allocation planner decides whether the extend the run limit of jobs, you must also enable plan-based scheduling by enabling the ALLOCATION_PLANNER parameter in the lsb.params file.

Related tasks

- Configuring extendable run limits

Related reference

- ALLOCATION_PLANNER parameter in lsb.params
- EXTENDABLE_RUNLIMIT parameter in the lsb.queues file

Default epsub executable files

Similar to **esub** programs, LSF now allows you to define a default **epsub** program that runs even if you do not define mandatory **epsub** programs with the LSB_ESUB_METHOD parameter in the lsf.conf file. To define a default **epsub** program, create an executable file named epsub (with no application name in the file name) in the LSF_SERVERDIR directory.

After the job is submitted, LSF runs the default **epsub** executable file if it exists in the LSF_SERVERDIR directory, followed by any mandatory **epsub** executable files that are defined by LSB_ESUB_METHOD, followed by the **epsub** executable files that are specified by the -a option.

Related concepts

- External job submission and execution controls

Restrict users and user groups from forwarding jobs to remote clusters

You can now specify a list of users or user groups that can forward jobs to remote clusters when using the LSF multicluster capability. This allows you to prevent jobs from certain users or user groups from being forwarded to an execution cluster, and to set limits on the submission cluster.

These limits are defined at the queue level in LSF. For jobs that are intended to be forwarded to a remote cluster, users must submit these jobs to queues that have the SNDJOBS_TO parameter configured in the lsb.queues file. To restrict these queues to specific users or user groups, define the FWD_USERS parameter in the lsb.queues file for these queues.

Related tasks

- Enable multicluster queues

Related reference

- FWD_USERS parameter in the lsb.queues file

Advance reservations now support the "same" section in resource requirement strings

When using the **brsvadd -R** and **brsvmod -R** options to specify resource requirements for advance reservations, the same string now takes effect, in addition to the select string. Previous versions of LSF only allowed the select string to take effect.

This addition allows you to select hosts with the same resources for your advance reservation.

Related concepts

- Same string

Related tasks

- Adding an advance reservation

Related reference

- **brsvadd -R** command option
- **brsvmod -R** command option

Priority factors for absolute priority scheduling

You can now set additional priority factors for LSF to calculate the job priority for absolute priority scheduling (APS). These additional priority factors allow you to modify the priority for the application profile, submission user, or user group, which are all used as factors in the APS calculation. You can also view the APS and fair share user priority values for pending jobs.

To set the priority factor for an application profile, define the PRIORITY parameter in the lsb.applications file. To set the priority factor for a user or user group, define the PRIORITY parameter in the User or UserGroup section of the lsb.users file.

The new **bjobs -prio** option displays the APS and fair share user priority values for all pending jobs. In addition, the **busers** and **bugroup** commands display the APS priority factor for the specified users or user groups.

Related concepts

- Absolute priority scheduling (APS)

Related reference

- PRIORITY parameter in the lsb.applications file
- PRIORITY parameter in the User or UserGroup section of the lsb.users file
- **bjobs -prio** command option
- **busers** command
- **bugroup** command

Job dispatch limits for users, user groups, and queues

You can now set limits on the maximum number of jobs that are dispatched in a scheduling cycle for users, user groups, and queues. This allows you to control the number of jobs, by user, user group, or queue, that are dispatched for execution. If the number of dispatched jobs reaches this limit, other pending jobs that belong to that user, user group, or queue that might have dispatched will remain pending for this scheduling cycle.

To set or update the job dispatch limit, run the **bconf** command on the limit object (that is, run `bconf` *`action_type`* `limit=`*`limit_name`*) to define the JOBS_PER_SCHED_CYCLE parameter for the specific limit. You can only set job dispatch limits if the limit consumer types are USERS, PER_USER, QUEUES, or PER_QUEUE.

For example, `bconf update limit=L1 "JOBS_PER_SCHED_CYCLE=10"`

You can also define the job dispatch limit by defining the JOBS_PER_SCHED_CYCLE parameter in the Limit section of the lsb.resources file.

Related reference

- Configuring resource allocation limits
- **bconf** command
- JOBS_PER_SCHED_CYCLE parameter in the lsb.resources file

Secondary Unix user group information transferred to execution host

This enhancement introduces the parameter LSF_UGROUP_TRANSFER in lsf.conf to enable the execution host to use the UNIX group information that is set by the user on the submission host. When set to "Y|y", secondary user group information is transferred from the submission host to the execution host for job execution, thereby overcoming an NFS limitation of 16 user groups.

Related reference

- LSF_UGROUP_TRANSFER parameter in the lsf.conf file.

Command output formatting

The following enhancements affect LSF command output formatting.

Note: LSF 10.1 Fix Pack 5 applies only to IBM POWER9 platforms.

blimits -a option shows all resource limits

The new **blimits -a** command option shows all resource allocation limits, even if they are not being applied to running jobs. Normally, running the **blimits** command with no options displays only resource allocation limits that are being applied to running jobs.

Related concepts

- Display resource allocation limits
- View information about resource allocation limits

Related reference

- **blimits -a** command option

Use bread -w to show messages and attached data files in wide format

LSF allows you to read messages and attached data files from a job in wide format with the new **bread -w** command option. The wide format displays information without truncating fields.

Related reference

- **bread -w** command option

Other changes to IBM Spectrum LSF

The following changes affect other aspects of LSF behavior.

Note: LSF 10.1 Fix Pack 5 applies only to IBM POWER9 platforms.

Use #include for shared configuration file content

In previous versions of LSF, you can use the `#INCLUDE` directive to insert the contents of a specified file into the beginning of the lsf.shared or lsb.applications configuration files to share common configurations between clusters or hosts.

You can now use the `#INCLUDE` directive in any place in the following configuration files:

- lsb.applications
- lsb.hosts
- lsb.queues
- lsb.reasons
- lsb.resources

- lsb.users

You can use the `#INCLUDE` directive only at the beginning of the following file:

- lsf.shared

For example, you can use `#if` ... `#endif` Statements to specify a time-based configuration that uses different configurations for different times. You can change the configuration for the entire system by modifying the common file that is specified in the `#INCLUDE` directive.

Related tasks

- Shared configuration file content

Related reference

- `#INCLUDE` directive in the lsb.applications file
- `#INCLUDE` directive in the lsb.hosts file
- `#INCLUDE` directive in the lsb.queues file
- `#INCLUDE` directive in the lsb.reasons file
- `#INCLUDE` directive in the lsb.resources file
- `#INCLUDE` directive in the lsb.users file

Showing the pending reason for interactive jobs

The **bsub -I** command now displays the pending reason for interactive jobs, based on the setting of LSB_BJOBS_PENDREASON_LEVEL, if the job is pending.

Related reference

- **bsub -I** command option
- **bsub -Ip** command option
- **bsub -IS** command option
- **bsub -ISp** command option
- **bsub -ISs** command option
- **bsub -Is** command option
- **bsub -IX** command option
- LSB_BJOBS_PENDREASON_LEVEL parameter in the lsf.conf file

Showing warning messages for interactive jobs

Interactive jobs can now show exit reasons when the jobs are killed (due to conditions such as reaching the memory or runtime limit). The exit reason is the same as the message shown for the output of the **bhist -l** and **bjobs -l** commands.

Related reference

- **bsub -I** command option
- **bsub -Ip** command option
- **bsub -IS** command option

- **bsub -ISp** command option
- **bsub -ISs** command option
- **bsub -Is** command option
- **bsub -IX** command option

Changing job priorities and limits dynamically

Through the introduction of two new parameters, LSF now supports changing job priorities and limits dynamically through an import file. This includes:

- Calling the eadmin script at a configured interval, even when a job exception has not occurred through the parameter EADMIN_TRIGGER_INTERVAL in the lsb.params file.
- Allowing job submission during a policy update or cluster restart through the parameter PERSIST_LIVE_CONFIG in the lsb.params file.
- Enhancement of the **bconf** command to override existing settings through the set action, to support the -pack option for reading multiple requests from a file.

Related reference

- EADMIN_TRIGGER_INTERVAL in the lsb.params file
- PERSIST_LIVE_CONFIG in the lsb.params file
- **bconf** command

Specify a UDP port range for LSF daemons

You can now specify a range of UDP ports to be used by LSF daemons. Previously, LSF binds to a random port number between 1024 and 65535.

To specify a UDP port range, define the LSF_UDP_PORT_RANGE parameter in the lsf.conf file. Include at least 10 ports in this range, and you can specify integers between 1024 and 65535.

Related reference

- LSF_UDP_PORT_RANGE in the lsf.conf file

What's new in IBM Spectrum LSF Version 10.1 Fix Pack 4

The following topics summarize the new and changed behavior in LSF 10.1 Fix Pack 4

Release date: December 2017

- **New platform support**
  The following new features are related to new platform support for LSF.
- **Performance enhancements**
  The following enhancements affect performance.
- **Resource management**
  The following new feature affects resource management and allocation.

- **Container support**
  The following new feature affects LSF support for containers.
- **GPU enhancements**
  The following enhancements affect LSF GPU support.
- **Job scheduling and execution**
  The following new feature affects LSF job scheduling and execution.
- **Data collection**
  The following new features affect IBM Spectrum LSF data collection.
- **Command output formatting**
  The following enhancements affect LSF command output formatting.
- **Other changes to IBM Spectrum LSF**
  The following changes affect other aspects of LSF behavior.

New platform support

The following new features are related to new platform support for LSF.

IBM POWER9

IBM Spectrum LSF 10.1 Fix Pack 4 includes support for IBM POWER9. The package for Linux on IBM Power LE (lsf10.1_lnx310-lib217-ppc64le) supports both IBM POWER8 and POWER9.

Performance enhancements

The following enhancements affect performance.

Use IBM Spectrum LSF Explorer to improve the performance of the bacct and bhist commands

The **bacct** and **bhist** commands can now use IBM Spectrum LSF Explorer (LSF Explorer) to get information instead of parsing the lsb.acct and lsb.events files. Using LSF Explorer improves the performance of the **bacct** and **bhist** commands by avoiding the need for parsing large log files whenever you run these commands.

To use this integration, LSF Explorer, Version 10.2, or later, must be installed and working. To enable this integration, edit the lsf.conf file, then define the LSF_QUERY_ES_SERVERS and LSF_QUERY_ES_FUNCTIONS parameters.

Related concepts

- Use to improve the performance of the bacct and bhist commands, or to retrieve additional data

Related reference

- LSF_QUERY_ES_SERVERS parameter in the lsf.conf file

- LSF_QUERY_ES_FUNCTIONS parameter in the lsf.conf file
- **bacct -f** command option
- **bhist -f** and **-n** command options

Resource management

The following new feature affects resource management and allocation.

What's new in resource connector for IBM Spectrum LSF

Extended AWS support

This feature extends LSF the resource connector AWS template to specify an Amazon EBS-Optimized instance. The AWS template also supports LSF exclusive resource syntax (`!resource`) in the instance attributes. LSF considers demand on the template only if a job explicitly asks for the resource in its combined resource requirement.

Related concepts

- Assigning exclusive resources to a template

Related reference

- awsprov_templates.json

Launch Google Compute Cloud instances

LSF clusters can launch instances from Google Compute Cloud to satisfy pending workload. The instances join the LSF cluster. If instances become idle, LSF resource connector automatically deletes them. Configure Google Compute Cloud as a resource provider with the googleprov_config.json and googleprov_templates.json files.

Related tasks

- Configuring Google Cloud Platform for resource connector

Related reference

- googleprov_config.json
- googleprov_templates.json

bhosts -rc and the bhosts -rconly commands show extra host information about provider hosts

Use the **bhosts -rc** and the **bhosts -rconly** command to see information about resources that are provisioned by LSF resource connector.

The -rc and -rconly options make use of the third-party **mosquitto** message queue application to support the additional information displayed by these **bhosts** options. The **mosquitto** binary file is included as part of the LSF distribution. To use the **mosquitto** daemon that is supplied with LSF, you must configure the LSF_MQ_BROKER_HOSTS parameter in the lsf.conf file to enable LIM to start the **mosquitto** daemon and for **ebrokerd** to send resource provider information to the MQTT message broker.

Related concepts

- Preparing to configure LSF resource connector for AWS

Related reference

- **bhosts -rc and -rconly**
- EBROKERD_HOST_CLEAN_DELAY
- MQTT_BROKER_PORT
- MQTT_BROKER_HOST
- LSF_MQ_BROKER_HOSTS

What's new in data manager for IBM Spectrum LSF

Enhanced LSF multicluster job forwarding

This feature enhances the LSF data manager implementation for the hybrid cloud environment using job forwarding with IBM Spectrum LSF multicluster capability (LSF multicluster capability). In this implementation, the cluster running in the public cloud is used as the execution cluster, and this feature enables the submission cluster to push the forwarding job's data requirement to the execution cluster and to receive the output back from the forwarding job. To enable this feature, specify the SNDJOBS_TO parameter in the lsb.queues file for the data transfer queue in the execution cluster, and specify the RCVJOBS_FROM parameter in the lsb.queues file for the submission cluster. The path of the FILE_TRANSFER_CMD parameter in the lsf.datamanager file for the data manager host must exist in the submission cluster.

Related concepts

- LSF data manager transfer queue

Related tasks

- Configuring the data transfer queue

Related reference

- SNDJOBS_TO and RCVJOBS_FROM in lsb.queues
- FILE_TRANSFER_CMD in lsf.datamanager

Specify a folder as the data requirement

When you specify a folder as a data requirement for a job, LSF generates a single signature for the folder as a whole, and only a single transfer job is required. You can also now use symbolically linked files in a job data requirement, and the colon (:) character can now be used in the path of a job data requirement.

When you submit a job with a data requirement, a data requirement that ends in a slash and an asterisk (/*) is interpreted as a folder. Only files at the top-level of the folder are staged. For example,

```
bsub -data "[host_name:]abs_folder_path/*" job
```

When you use the asterisk character (*) at the end of the path, the data requirements string must be in quotation marks.

A data requirement that ends in a slash (/) is also interpreted also as a folder, but all files including subfolders are staged. For example,

```
bsub -data "[host_name:]abs_folder_path/" job
```

To specify a folder a data requirement for a job, you must have access to the folder and its contents. You must have read and execute permission on folders, and read permission on regular files. If you don't have access to the folder, the submission is rejected.

Related tasks

- Specifying data requirements for your job

Related reference

- **bdata cache**
- **bdata chgrp**
- bstage in
- bstage out
- **bsub -data**

Container support

The following new feature affects LSF support for containers.

Support for systemd with Docker jobs

When running jobs for Docker containers, you can now use the **systemd** daemon as the Docker **cgroup** driver. This means that you can now run Docker jobs regardless of which **cgroup** driver is used.

To support Docker with the **systemd** cgroup driver and all other cgroup drivers, configure both the EXEC_DRIVER and CONTAINER parameters. This new configuration provides transparent Docker container support for all cgroup drivers and other container features.

Related tasks

- Configuring to run Docker jobs

Related reference

- EXEC_DRIVER parameter in the lsb.applications file.

GPU enhancements

The following enhancements affect LSF GPU support.

NVIDIA Data Center GPU Manager (DCGM) integration updates

LSF, Version 10.1 Fix Pack 2 integrated with NVIDIA Data Center GPU Manager (DCGM) to work more effectively with GPUs in the LSF cluster. LSF now integrates with Version 1.1 of the NVIDIA Data Center GPU Manager (DCGM) API. This update provides the following enhancements to the DCGM features for LSF:

- LSF checks the status of GPUs to automatically filter out unhealthy GPUs when the job allocates GPU resources, and to automatically add back the GPU if it becomes healthy again.
- DCGM provides mechanisms to check the GPU health and LSF integrates these mechanisms to check the GPU status before, during, and after the job is running to meet the GPU requirements. If LSF detects that a GPU is not healthy before the job is complete, LSF requeues the job. This ensures that the job runs on healthy GPUs.
- GPU auto-boost is now enabled for single-GPU jobs, regardless of whether DCGM is enabled. If DCGM is enabled, LSF also enables GPU auto-boost on jobs with exclusive mode that run across multiple GPUs on one host.

Enable the DCGM integration by defining the LSF_DCGM_PORT parameter in the lsf.conf file.

Related concepts

- Define GPU resources (Obsolete)

Related reference

- LSF_DCGM_PORT parameter in the lsf.conf file
- **bacct -gpu** option
- **bjobs -gpu** option
- **bhist -gpu** option

Job scheduling and execution

The following new feature affects LSF job scheduling and execution.

External job switch control with eswitch

Similar to the external job submission and execution controls (**esub**, **epsub**, and **eexec** programs), LSF now allows you to use external, site-specific binary files or scripts that are associated with a request to switch a job to another queue. By writing external job switch executable files, you can accept, reject, or change the destination queue for any **bswitch** request.

Similar to the **bsub -a** option, the new **bswitch -a** option specifies one or more application-specific external executable files (**eswitch** files) that you want LSF to associate with the switch request.

Similar to the LSB_ESUB_METHOD parameter, the new LSB_ESWITCH_METHOD environment variable or parameter in the lsf.conf file allows you to specify one or more mandatory **eswitch** executable files.

When running any job switch request, LSF first invokes the executable file named **eswitch** (without *.application_name* in the file name) if it exists in the LSF_SERVERDIR directory. If an LSF administrator specifies one or more mandatory **eswitch** executable files using the LSB_ESWITCH_METHOD parameter in the lsf.conf file, LSF then invokes the mandatory executable files. Finally, LSF invokes any application-specific **eswitch** executable files (with *.application_name* in the file name) specified by the **bswitch -a** option. An **eswitch** is run only once, even if it is specified by both the **bswitch -a** option and the LSB_ESWITCH_METHOD parameter.

Related concepts

- Use external job switch controls

Related reference

- LSB_ESWITCH_METHOD parameter in the lsf.conf file

Related information

- **bswitch -a** command option

Advance reservation enhancements

LSF now features enhancements to advance reservations. You can enable LSF to allow jobs to run on advance reservation hosts even if it cannot finish before the advance reservation becomes active (by default, these jobs are suspended when the first advance reservation job starts). The advance reservations can run a pre-script before the advance reservation starts and a post-script when the advance reservation expires. These enhancements are specified in the **brsvadd** and **brsvmod** commands (-q, -nosusp, -E, -Et, -Ep, and -Ept options).

Because the **ebrokerd** daemon starts the advance reservation scripts, you must specify LSB_START_EBROKERD=Y in the lsf.conf file to enable advance reservations to run pre-scripts and post-scripts.

Related tasks

- Adding an advance reservation

Related reference

- **brsvadd** command
- **brsvmod** command
- LSB_START_EBROKERD parameter in the lsf.conf file.

Deleting empty job groups

This enhancement supports the deleting of empty implicit job groups automatically even if they have limits. It adds a new option "all" to the JOB_GROUP_CLEAN parameter in lsb.params, to delete empty implicit job groups automatically even if they have limits.

Related reference

- JOB_GROUP_CLEAN parameter in the lsb.params file

Data collection

The following new features affect IBM Spectrum LSF data collection.

Enhanced energy accounting using Elasticsearch

This enhancement introduces the **lsfbeat** tool, which calls the **ipmitool** to collect the energy data of each host and to send the data to IBM Spectrum LSF Explorer (LSF Explorer). The **bjobs** and **bhosts** commands get the energy data from LSF Explorer and display it. To use this feature, LSF Explorer must be deployed in your LSF cluster. To enable the **lsfbeat** energy service, configure `LSF_ENABLE_BEAT_SERVICE="energy"` in the lsf.conf file, then run the **lsadmin limrestart all** command to start up the **lsfbeat** service. To query energy data with the **bhosts** and **bjobs** commands, configure `LSF_QUERY_ES_FUNCTIONS="energy"` and `LSF_QUERY_ES_SERVERS="`*`ip:port`*`"` in the lsf.conf file.

Related concepts

- Use to improve the performance of the bacct and bhist commands, or to retrieve additional data

Related reference

- LSF_ENABLE_BEAT_SERVICE parameter in the lsf.conf file
- LSF_QUERY_ES_SERVERS parameter in the lsf.conf file
- LSF_QUERY_ES_FUNCTIONS parameter in the lsf.conf file

Data provenance tools

LSF now has data provenance tools to trace files that are generated by LSF jobs.

You can use LSF data provenance tools to navigate your data to find where the data is coming from and how it is generated. In addition, you can use data provenance information to reproduce your data results when using the same job input and steps.

When submitting a job with the **bsub** command, enable data provenance by defining `LSB_DATA_PROVENANCE=Y` as an environment variable (`bsub -e LSB_DATA_PROVENANCE=Y`) or by using the **esub.dprov** application (`bsub -a 'dprov(`*`file_path`*`)'`), and use the **tag.sh** post-execution script to mark the data provenance attributes for the output data files (**-Ep 'tag.sh'**). You can also use the showhist.py script to generate a picture to show the relationship of your data files.

Data provenance requires the use of IBM Spectrum Scale (GPFS) as the file system to support the extended attribute specification of files and Graphviz, which is an open source graph visualization software, to generate

pictures from the showhist.py script.

Related concepts

- Data provenance

Related reference

- Environment variables for data provenance

Related information

- LSB_DATA_PROVENANCE environment variable

Command output formatting

The following enhancements affect LSF command output formatting.

esub and epsub enhancement

LSF users can select different esub (or epsub) applications (or scripts) using **bsub -a** (or **bmod -a**). LSF has a number of different esub applications that users can select, but the **bjobs** and **bhist** commands did not previously show details about these applications in output. This enhancement enables the **bjobs -l**, **bjobs -o esub**, and **bhist -l** commands to show detailed information about esub and epsub applications.

Related reference

- **bjobs -l** command
- **bjobs -o esub** command
- **bhist -l** command

Energy usage in output of bjobs -l, bjobs -o, and bhosts -l

If enhanced energy accounting using Elasticsearch has been enabled (with LSF_ENABLE_BEAT_SERVICE in lsf.conf), output for **bjobs -l** and **bjobs -o energy** shows the energy usage in Joule and kWh. and **bhosts -l** shows the Current Power usage in watts and total Energy Consumed in Joule and kWh.

Related reference

- LSB_ENABLE_BEAT_SERVICE parameter in lsf.conf
- **bjobs -l** command

Other changes to IBM Spectrum LSF

The following changes affect other aspects of LSF behavior.

Enhance fair share calculation for job forwarding mode in the LSF multicluster capability

In previous versions of LSF, when calculating the user priority in the fair share policies, if a job is forwarded to remote clusters while using the LSF multicluster capability, the fair share counter for the submission host is not updated. For example, if the fair share calculation determines that a user's job has a high priority and there are no local resources available, that job is forwarded to a remote cluster, but the LSF scheduler still considers the job for forwarding purposes again because the fair share counter is not updated.

To resolve this issue, LSF now introduces a new forwarded job slots factor (FWD_JOB_FACTOR) to account for forwarded jobs when making the user priority calculation for the fair share policies. To use this forwarded job slots factor, specify the FWD_JOB_FACTOR to a non-zero value in the lsb.params file for cluster-wide settings, or in the lsb.queues file for an individual queue. If defined in both files, the queue value takes precedence. In the user priority calculation, the FWD_JOB_FACTOR parameter is used for forwarded job slots in the same way that the RUN_JOB_FACTOR parameter is used for job slots. To treat remote jobs and local jobs as the same, set FWD_JOB_FACTOR to the same value as RUN_JOB_FACTOR.

When accounting for forwarded jobs in the fair share calculations, job usage might be counted twice if global fair share is used because job usage is counted on the submission cluster, then counted again when the job is running on a remote cluster. To avoid this problem, specify the duration of time after which LSF removes the forwarded jobs from the user priority calculation for fair share scheduling by specifying the LSF_MC_FORWARD_FAIRSHARE_CHARGE_DURATION parameter in the lsf.conf file. If you enabled global fair share and intend to use the new forwarded job slots factor, set the value of LSF_MC_FORWARD_FAIRSHARE_CHARGE_DURATION to double the value of the SYNC_INTERVAL parameter in the lsb.globalpolicies file (approximately 5 minutes) to avoid double-counting the job usage for forwarded jobs. If global fair share is not enabled, this parameter is not needed.

Related concepts

- Enhance fair share calculation to include the job forwarding mode
- Global fair share scheduling

Related reference

- FWD_JOB_FACTOR parameter in the lsb.queues file.
- FWD_JOB_FACTOR parameter in the lsb.params file.
- LSF_MC_FORWARD_FAIRSHARE_CHARGE_DURATION parameter in the lsf.conf file.

Dynamically load the hardware locality (hwloc) library

LSF now allows you to dynamically load the hardware locality (**hwloc**) library from the system library paths whenever it is needed to support newer hardware.

LSF for the following platforms is compiled and linked with the library and header file for **hwloc**, Version 1.11.8, and detects most of the latest hardware without enabling this feature:

- Linux x64 Kernel 2.6, glibc 2.5
- Linux x64 Kernel 3.10, glibc 2.17
- Linux ppc64le Kernel 3.10, glibc 2.17

- Linux ARMv8 Kernel 3.12, glibc 2.17
- Windows

All other platforms use **hwloc**, Version 1.8.

Enable the dynamic loading of the **hwloc** library by defining the LSF_HWLOC_DYNAMIC parameter as `Y` in the lsf.conf file.

Related concepts

- Portable hardware locality

Related reference

- LSF_HWLOC_DYNAMIC parameter in the lsf.conf file.

What's new in IBM Spectrum LSF Version 10.1.0 Fix Pack 3

The following topics summarize the new and changed behavior in LSF 10.1.0 Fix Pack 3

Release date: 12 September 2017

- **Job scheduling and execution**
  The following new features affect job scheduling and execution.
- **Resource management**
  The following new feature affects resource management and allocation.
- **Container support**
  The following new feature affects LSF support for containers.
- **Command output formatting**
  The following new features are related to the LSF command output.
- **Logging and troubleshooting**
  The following new features are related to logging and troubleshooting.
- **Other changes to IBM Spectrum LSF**
  The following changes are related to command options and LSF default behavior.

Job scheduling and execution

The following new features affect job scheduling and execution.

View jobs that are associated with an advance reservation

The new **bjobs -U** option allows you to display jobs that are associated with the specified advance reservation.

To view the reservation ID of the advance reservation that is associated with a job ID, use the **bjobs -o** option and specify the `rsvid` column name.

Related tasks

- View jobs that are associated with an advance reservation

Related reference

- **bjobs -U** command option

Dynamically scheduled reservations

A *dynamically scheduled* reservation accepts jobs based on currently available resources. Use the **brsvsub** command to create a dynamically scheduled reservation and submit a job to to fill the advance reservation when the resources required by the job are available.

Jobs that are scheduled for the reservation run when the reservation is active. Because they are scheduled like jobs, dynamically scheduled reservations do not interfere with running workload (unlike normal advance reservations, which kill any running jobs when the reservation window opens.)

Related concepts

- Advance reservation

Related reference

- brsvsub

Resource management

The following new feature affects resource management and allocation.

Request additional resources to allocate to running jobs

The new **bresize request** subcommand option allows you to request additional tasks to be allocated to a running resizable job, which grows the resizable job. This means that you can both grow and shrink a resizable job by using the **bresize** command.

Related concepts

- Resizable jobs

Related reference

- **bresize** command

Specify GPU resource requirements for your jobs

Specify all GPU resource requirement as part of job submission, or in a queue or application profile. Use the option bsub –gpu to submit jobs that require GPU resources. Specify how LSF manages GPU mode (exclusive or shared), and whether to enable the NVIDIA Multi-Process Service (MPS) for the GPUs used by the job.

The parameter LSB_GPU_NEW_SYNTAX in the lsf.conf file enables jobs to use GPU resource requirements that are specified with the **bsub -gpu** option or in the queue, application profile.

Use the bsub -gpu option to specify GPU requirements for your job or submit your job to a queue or application profile that configures GPU requirements in the GPU_REQ parameter.

Set a default GPU requirement by configuring the LSB_GPU_REQ parameter in the lsf.conf file.

Use the **bjobs -l** command to see the combined and effective GPU requirements that are specified for the job.

Related concepts

- Specify GPU resource requirements for your jobs

Related reference

- **bsub -gpu**
- LSB_GPU_NEW_SYNTAX
- LSB_GPU_REQ

What's new in resource connector for IBM Spectrum LSF

Support for Microsoft Azure as a resource provider

LSF resource connector now supports Microsoft Azure as a resource provider. LSF clusters can launch instances from Microsoft Azure if the workload demand exceeds cluster capacity. The resource connector generates requests for additional hosts from these providers and dispatches jobs to dynamic hosts that join the LSF cluster. When the demand reduces, the resource connector shuts down the LSF slave daemons and cancels allocated virtual servers.

To specify the configuration for provisioning from Microsoft Azure, use the azureprov_config.json and the azureprov_templates.json configuration files.

Submit jobs to use AWS Spot instances

Use *Spot instances* to bid on spare Amazon EC2 computing capacity. Since Spot instances are often available at a discount compared to the pricing of On-Demand instances, you can significantly reduce the cost of running your applications, grow your application’s compute capacity and throughput for the same budget, and enable new types of cloud computing applications.

With Spot instances you can reduce your operating costs by up to 50-90%, compared to on-demand instances. Since Spot instances typically cost 50-90% less, you can increase your compute capacity by 2-10 times within the same budget.

Spot instances are supported on any Linux x86 system that is supported by LSF.

Support federated accounts with temporary access tokens

Resource connector supports *federated accounts* for LSF resource connector as an option instead of requiring permanent AWS IAM account credentials. Federated users are external identities that are granted temporary credentials with secure access to resources in AWS without requiring creation of IAM users. Users are authenticated outside of AWS (for example, through Windows Active Directory).

Use the AWS_CREDENTIAL_SCRIPT parameter in the awsprov_config.json file to specify a path to the script that generates temporary credentials for federated accounts. For example,

```
AWS_CREDENTIAL_SCRIPT=/shared/dir/generateCredentials.py
```

LSF executes the script as the primary LSF administrator to generate a temporary credentials before it creates the EC2 instance.

Support starting instances within an IAM Role

IAM *roles* group AWS access control privileges together. A role can be assigned to an IAM user or an IAM instance profile. IAM *Instance Profiles* are containers for IAM roles that allow you to associate an EC2 instance with a role through the profile. The EC2 runtime environment contains temporary credentials that have the access control permissions of the profile role.

To make the roles available for resource connector to create instances, use the instanceProfile attribute in the awsprov_templates.json file to specify an AWS IAM instance profile to assign to the requested instance. Jobs running in that instance can use the instance profile credentials to access other AWS resources. Resource connector uses that information to request EC2 compute instances with particular instance profiles. Jobs that run on those hosts use temporary credentials provided by AWS to access the AWS resources that the specified role has privileges for.

Tag attached EBS volumes in AWS

The instanceTags attribute in the awsprov_templates.json file can tag EBS volumes with the same tag as the instance. EBS volumes in AWS are persistent block storage volumes used with an EC2 instance. EBS volumes are expensive, so you can use the instance ID that tags the volumes for the accounting purposes.
Note: The tags cannot start with the string `aws:`. This prefix is reserved for internal AWS tags. AWS gives an error if an instance or EBS volume is tagged with a keyword starting with `aws:`. Resource connector removes and ignores user-defined tags that start with `aws:`.

Resource connector demand policies in queues

The RC_DEMAND_POLICY parameter in the lsb.queues file defines threshold conditions to determine whether demand is triggered to borrow resources through resource connector for all the jobs in the queue. As long as pending jobs at the queue meet at least one threshold condition, LSF expresses the demand to resource connector to trigger borrowing.

The demand policy defined by the RC_DEMAND_POLICY parameter can contain multiple conditions, in an OR relationship. A condition is defined as `[` *num_pend_jobs*[,*duration*]`]`. The queue has more than the specified number of eligible pending jobs that are expected to run at least the specified duration in minutes. The num_pend_jobs option is required, and the duration is optional. The default duration is 0 minutes.

View the status of provisioned hosts with the bhosts -rc command

Use the **bhosts -rc** or the **bhosts -rconly** command to see the status of resources provisioned by LSF resource connector.

To use the -rc and -rconly options, the **mosquitto** binary file for the MQTT broker must be installed in LSF_SERVERDIR, and running (check with the `ps -ef | grep mosquitto` command). The the

LSF_MQ_BROKER_HOSTS parameter must be configured in the lsf.conf file.

For hosts provisioned by resource connector, the `RC_STATUS`, `PROV_STATUS`, and `UPDATED_AT` columns show appropriate status values and a timestamp. For other hosts in the cluster, these columns are empty.

For example,

```
bhosts -rc
HOST_NAME          STATUS       JL/U    MAX  NJOBS    RUN  SSUSP  USUSP    RSV
RC_STATUS      PROV_STATUS    UPDATED_AT
ec2-35-160-173-192 ok              -      1      0      0      0      0      0
Allocated      running        2017-04-07T12:28:46CDT
lsf1.aws.          closed          -      1      0      0      0      0      0
```

The -l option shows more detailed information about provisioned hosts.

```
bhosts -rc -l
HOST  ec2-35-160-173-192.us-west-2.compute.amazonaws.com
STATUS           CPUF  JL/U    MAX  NJOBS    RUN  SSUSP  USUSP    RSV RC_STATUS
PROV_STATUS    UPDATED_AT             DISPATCH_WINDOW
ok              60.00     -      1      0      0      0      0      0 Allocated
running        2017-04-07T12:28:46CDT      -

 CURRENT LOAD USED FOR SCHEDULING:
                r15s   r1m  r15m    ut    pg    io   ls    it   tmp   swp   mem
slots
 Total           1.0   0.0   0.0    1%   0.0    33    0     3 5504M    0M  385M
1
 Reserved        0.0   0.0   0.0    0%   0.0     0    0     0    0M    0M    0M
-
```

The -rconly option shows the status of all hosts provisioned by LSF resource connector, no matter if they have joined the cluster or not.

Related concepts

- Use AWS spot instances

Related tasks

- Configuring AWS Spot instances
- Configuring Amazon Web Services for resource connector
- Configuring AWS access with federated accounts

Related reference

- awsprov_config.json
- awsprov_templates.json
- policy_config.json
- lsf.conf file reference for resource connectory
- RC_DEMAND_POLICY in lsb.queues

Container support

The following new feature affects LSF support for containers.

Pre-execution scripts to define container options

When running jobs for Docker, Shifter, or Singularity, you can now specify a pre-execution script that outputs container options that are passed to the container job. This allows you to use a script to set up the execution options for the container job.

Related tasks

- Configuring the Docker application profiles in LSF to run Docker container jobs
- Configuring the Shifter application profiles in LSF to run Shifter container jobs
- Configuring the Singularity application profiles in LSF to run Singularity container jobs

Related reference

- CONTAINER parameter in the lsb.applications file

Command output formatting

The following new features are related to the LSF command output.

Customize host load information output

Like the **bjobs -o** option, you can now also customize specific fields that the **lsload** command displays by using the -o command option. This allows you to create a specific output format, allowing you to easily parse the information by using custom scripts or to display the information in a predefined format.

You can also specify the default output formatting of the **lsload** command by specifying the LSF_LSLOAD_FORMAT parameter in the lsf.conf file, or by specifying the LSF_LSLOAD_FORMAT environment variable.

Related concepts

- Customize host load information output

Related information

- **lsload -o** command option

View customized host load information in JSON format

With this release, you can view customized host load information in JSON format by using the new -json command option with the **lsload** command. Since JSON is a customized output format, you must use the -json option together with the -o option.

Related concepts

- Customize host load information output

Related information

- **lsload -json** command option

New output fields for busers -w

With this release, two new fields have been added to the output for **busers -w**: `PJOBS` and `MPJOBS`.

The fields shown for **busers -w** now includes:

PEND
: The number of tasks in all of the specified users' pending jobs. If used with the -alloc option, the total is `0`.

MPEND
: The pending job slot threshold for the specified users or user groups. `MPEND` is defined by the MAX_PEND_SLOTS parameter in the lsb.users configuration file.

PJOBS
: The number of users' pending jobs.

MPJOBS
: The pending job threshold for the specified users. `MPJOBS` is defined by the MAX_PEND_JOBS parameter in the configuration file lsb.users.

Related reference

- MAX_PEND_JOBS
- MAX_PEND_SLOTS
- lsb.users
- **busers -w**

Related information

- New MAX_PEND_SLOTS parameter and change to MAX_PEND_JOBS parameter

Logging and troubleshooting

The following new features are related to logging and troubleshooting.

Diagnose mbatchd and mbschd performance problems

LSF provides a feature to log profiling information for the **mbatchd** and **mbschd** daemons to track the time that the daemons spend on key functions. This can assist IBM Support with diagnosing daemon performance problems.

To enable daemon profiling with the default settings, edit the lsf.conf file, then specify `LSB_PROFILE_MBD=Y` for the **mbatchd** daemon or specify `LSB_PROFILE_SCH=Y` for the **mbschd** daemon. You can also add keywords within these parameters to further customize the daemon profilers.

Related concepts

- Logging mbatchd and mbschd profiling information

Related reference

- LSB_PROFILE_MBD parameter in the lsf.conf file
- LSB_PROFILE_SCH parameter in the lsf.conf file

Other changes to IBM Spectrum LSF

The following changes are related to command options and LSF default behavior.

Changed command options

Specify multiple email addresses with the bsub -u option

You can now specify multiple email addresses with the **bsub -u** option by enclosing the string in quotation marks and using a space to separate each email address. The total length of the address string cannot be longer than 511 characters.

The bpeek -f option now exits when the peeked job is complete

The **bpeek -f** command option now exits when the peeked job is completed.

If the peeked job is requeued or migrated, the **bpeek** command only exits if the job is completed again. In addition, the **bpeek** command cannot get the new output of the job. To avoid these issues, abort the previous **bpeek -f** command and rerun the **bpeek -f** command after the job is requeued or migrated.

Specify remote hosts with the bsub -m option

You can now specify remote hosts by using the **bsub -m** command option when using the job forwarding model with the LSF multicluster capability. To specify remote hosts, use *host_name@cluster_name*.

Changed configuration parameters

New MAX_PEND_SLOTS parameter and change to MAX_PEND_JOBS parameter

With the addition of the new parameter MAX_PEND_SLOTS, the concept of MAX_PEND_JOBS has changed. MAX_PEND_JOBS (in both lsb.users and lsb.params) has been changed to control the maximum pending "jobs" where previously it controlled the maximum pending "slot" threshold. MAX_PEND_SLOTS has been introduced therefore, to control the previous intention of MAX_PEND_JOBS.

This means that for customers who previously configured MAX_PEND_JOBS (for example, in lsb.users, for a user or group pending job slot limit), they must update the parameter to job count instead of slot count, or

replace the parameter with the new MAX_PEND_SLOTS, which is meant for backward compatibility.

Related concepts

- About job states
- Job packs
- How resource allocation limits work

Related reference

- MAX_PEND_JOBS
- MAX_PEND_SLOTS
- lsb.users

Changes to default LSF behavior

Improvements to the LSF Integration for Rational ClearCase

Daemon wrapper performance is improved with this release because the daemon wrappers no longer run the **checkView** function to check the ClearCase view (as set by the CLEARCASE_ROOT environment variable) under any conditions. In addition, the NOCHECKVIEW_POSTEXEC environment variable is now obsolete since it is no longer needed.

If the **cleartool setview** command fails when called by a daemon wrapper, the failure reason is shown in the **bjobs -l**, **bhist -l**, **bstatus**, and **bread** commands if `DAEMON_WRAP_ENABLE_BPOST=Y` is set as an environment variable.

What's new in IBM Spectrum LSF Version 10.1 Fix Pack 2

The following topics summarize the new and changed behavior in LSF 10.1 Fix Pack 2

Release date: 14 April 2017.

- **Performance enhancements**
  The following new features can improve performance.
- **Container support**
  The following new features affect LSF support for containers.
- **GPU**
  The following new features affect GPU support.
- **Installation**
  The following new features affect LSF installation.
- **Resource management**
  The following new features affect resource management and allocation.
- **Command output formatting**
  The following new features are related to the LSF command output.
- **Security**
  The following new features affect cluster security.

Performance enhancements

The following new features can improve performance.

Improved mbatchd performance and scalability

Job dependency evaluation is used to check whether each job's dependency condition is satisfied. You can improve the performance and scalability of the **mbatchd** daemon by limiting the amount of time that **mbatchd** takes to evaluate job dependencies in one scheduling cycle. This limits the amount of time that the job dependency evaluation blocks services and frees up time to perform other services during the scheduling cycle. Previously, you could only limit the maximum number of job dependencies, which only indirectly limited the amount of time spent evaluating job dependencies. Job dependency evaluation is a process that is used to check whether each job's dependency condition is satisfied.

Related reference

- EVALUATE_JOB_DEPENDENCY_TIMEOUT parameter in the lsb.params file
- EVALUATE_JOB_DEPENDENCY parameter in the lsb.params file

Improve performance of LSF daemons by automatically configuring CPU binding

You can now enable LSF to automatically bind LSF daemons to CPU cores by enabling the LSF_INTELLIGENT_CPU_BIND parameter in the lsf.conf file. LSF automatically creates a CPU binding configuration file for each master and master candidate host according to the automatic binding policy.

Related concepts

- Automatically bind LSF daemons to specific CPU cores

Related reference

- LSF_INTELLIGENT_CPU_BIND parameter in the lsf.conf file

Reduce mbatchd workload by allowing user scripts to wait for a specific job condition

The new **bwait** command pauses and waits for the specified job condition to occur before the command returns. End users can use this command to reduce workload on the **mbatchd** daemon by including **bwait** in a user script for running jobs instead of using the **bjobs** command in a tight loop to check the job status. For example, the user script might have a command to submit a job, then run **bwait** to wait for the first job to be `DONE` before continuing the script.

The new **lsb_wait()** API provides the same functionality as the **bwait** command.

Related reference

- **bwait** command
- DEFAULT_BWAIT_TIME parameter in the lsb.params file
- EVALUATE_WAIT_CONDITION_TIMEOUT parameter in the lsb.params file
- LSB_BWAIT_REREG_INTERVAL parameter in the lsf.conf file

Changes to default LSF behavior

Parallel restart of the mbatchd daemon

The **mbatchd** daemon now restarts in parallel by default. This means that there is always an **mbatchd** daemon handling client commands during the restart to help minimize downtime for LSF. LSF starts a new or child **mbatchd** daemon process to read the configuration files and replace the event file. Previously, the **mbatchd** daemon restarted in serial by default and required the use of the **badmin mbdrestart -p** command option to restart in parallel. To explicitly enable the **mbatchd** daemon to restart in serial, use the new **badmin mbdrestart -s** command option.

New default value for caching a failed DNS lookup

The default value of the LSF_HOST_CACHE_NTTL parameter in the lsf.conf file is increased to the maximum valid value of 60 seconds (from 20 seconds). This reduces the amount of time that LSF takes to repeat failed DNS lookup attempts.

Multithread mbatchd job query daemon

LSF enables the multithread mbatchd job query daemon by setting the following parameter values at the time of installation:

- The LSB_QUERY_PORT parameter in the lsf.conf file is set to `6891`, which enables the multithread **mbatchd** job query daemon and specifies the port number that the **mbatchd** daemon uses for LSF query requests.
- The LSB_QUERY_ENH parameter in the lsf.conf file is set to `Y`, which extends multithreaded query support to batch query requests (in addition to **bjobs** query requests).

Container support

The following new features affect LSF support for containers.

Running LSF jobs in Shifter containers

LSF now supports the use of Shifter, Version 16.08.3, or later, which must be installed on an LSF server host.

The Shifter integration allows LSF to run jobs in Shifter containers on demand.

Related concepts

- Use IBM Spectrum LSF with Shifter

Running LSF jobs in Singularity containers

LSF now supports the use of Singularity, Version 2.2, or later, which must be installed on an LSF server host.

The Singularity integration allows LSF to run jobs in Singularity containers on demand.

Related concepts

- Use IBM Spectrum LSF with Singularity

GPU

The following new features affect GPU support.

Integration with NVIDIA Data Center GPU Manager (DCGM)

The NVIDIA Data Center GPU Manager (DCGM) is a suite of data center management tools that allow you to manage and monitor GPU resources in an accelerated data center. LSF integrates with NVIDIA DCGM to work more effectively with GPUs in the LSF cluster. DCGM provides additional functionality when working with jobs that request GPU resources by:

- providing GPU usage information for EXCLUSIVE_PROCESS mode jobs.
- checking the GPU status before and after the jobs run to identify and filter out unhealthy GPUs.
- synchronizing the GPU auto-boost feature to support jobs that run across multiple GPUs.

Enable the DCGM integration by defining the LSF_DCGM_PORT parameter in the lsf.conf file.

Related reference

- LSF_DCGM_PORT parameter in the lsf.conf file

Installation

The following new features affect LSF installation.

Enabling support for Linux cgroup accounting to control resources

Control groups (**cgroups**) are a Linux feature that affects the resource usage of groups of similar processes, allowing you to control how resources are allocated to processes that are running on a host.

With this release, you can enable the **cgroup** feature with LSF by enabling the ENABLE_CGROUP parameter in the install.config file for LSF installation. The LSF installer sets initial configuration parameters to use the **cgroup** feature.

Related reference

- ENABLE_CGROUP parameter in the install.config file

Automatically enable support for GPU resources at installation

Support for GPU resources in previous versions of LSF required manual configuration of the GPU resources in the lsf.shared and lsf.cluster.*cluster_name* files.

With this release, you can enable LSF to support GPUS automatically by enabling the ENABLE_GPU parameter in the install.config file for LSF installation. The LSF installer sets initial configuration parameters to support the use of GPU resources.

Related reference

- ENABLE_GPU parameter in the install.config file

Resource management

The following new features affect resource management and allocation.

Accurate affinity accounting for job slots

Affinity accounting is an extension of HPC allocation feature, where LSF accounts all the slots on the allocated hosts for exclusive jobs. Previous versions of LSF miscalculated the job accounting for job slots when affinity is used in the resource requirement string (in the **bsub -R** option). LSF can now accurately account the number of slots that are consumed by jobs with affinity requirements. LSF calculates the number of slots that are required by affinity jobs when the job task is allocated to the host. The processor unit (PU) that is used for calculating the number of slots is the effective ncpus value on the host. LSF uses this effective ncpus value to calculate the number of slots that are required by affinity jobs when the job task is allocated to the host.

Enable HPC allocation and affinity accounting by defining the LSB_ENABLE_HPC_ALLOCATION parameter in the lsf.conf file.

Related reference

- LSB_ENABLE_HPC_ALLOCATION parameter in the lsf.conf file.

Pre-provisioning and post-provisioning in LSF resource connector

Set up pre-provisioning in LSF resource connector to run commands before the resource instance joins the cluster. Configure post-provisioning scripts to run clean up commands after the instance is terminated, but before the host is removed from the cluster.

Related concepts

- Pre-provisioning and post-provisioning in LSF resource connector

Related reference

- hostProviders.json file reference

Configure resource provisioning policies in LSF resource connector

LSF resource connector provides built in policies for limiting the number of instances to be launched and the maximum number of instances to be created. The default plugin framework is a single python script that communicates via stdin and stdout in JSON data structures. LSF resource connector provides an interface for administrators to write their own resource policy plugin.

Related concepts

- Configure resource provisioning policies in LSF resource connector

Related reference

- policy_config.json file reference
- hostProviders.json file reference

Improvements to units for resource requirements and limits

For the **bsub**, **bmod**, and **brestart** commands, you can now use the `ZB` (or `Z`) unit in addition to the following supported units for resource requirements and limits: `KB` (or `K`), `MB` (or `M`), `GB` (or `G`), `TB` (or `T`), `PB` (or `P`), `EB` (or `E`). The specified unit is converted to the appropriate value specified by the LSF_UNIT_FOR_LIMITS parameter. The converted limit values round up to a positive integer. For resource requirements, you can specify unit for `mem`, `swp` and `tmp` in select and rusage section.

By default, the `tmp` resource is not supported by the LSF_UNIT_FOR_LIMITS parameter. Use the parameter LSF_ENABLE_TMP_UNIT=Y to enable the LSF_UNIT_FOR_LIMITS parameter to support limits on the `tmp` resource.

When the LSF_ENABLE_TMP_UNIT=Y parameter is set and the LSF_UNIT_FOR_LIMITS parameter value is not `MB`, an updated LIM used with old query commands has compatibility issues. The unit for the `tmp` resource changes with the LSF_UNIT_FOR_LIMITS parameter in LIM, but query commands still display the unit for the `tmp` resource as `MB`.

Related reference

- LSF_ENABLE_TMP_UNIT parameter in the lsf.conf file

Command output formatting

The following new features are related to the LSF command output.

Customize host and queue information output

Like the **bjobs -o** option, you can now also customize specific fields that the **bhosts** and **bqueues** commands display by using the -o command option. This allows you to create a specific output format that shows all the required information, which allows you to easily parse the information by using custom scripts or to display the information in a predefined format.

You can also specify the default output formatting of the **bhosts** and **bqueues** commands by specifying the LSB_BHOSTS_FORMAT and LSB_BQUEUES_FORMAT parameters in the lsf.conf file, or by specifying the LSB_BHOSTS_FORMAT and LSB_QUEUES_FORMAT environment variables.

Related concepts

- Customize host information output
- Customize queue information output

Related reference

- **bqueues -o** command option

Related information

- **bhosts -o** command option

View customized information output in JSON format

With this release, you can view customized job, host, and queue information in JSON format by using the new -json command option with the **bjobs**, **bhosts**, and **bqueues** commands. Since JSON is a customized output format, you must use the -json option together with the -o option.

Related information

- View customized host information in JSON format
- View customized queue information in JSON format
- bhosts -json command option
- bqueues -json command option

View time in customized job information output in hh:mm:ss format

You can now view times in customized job information in `hh:mm:ss` format by using the new -hms command option with the **bjobs** command. Since the `hh:mm:ss` time format is a customized output format, you must

use the -hms option together with the -o or -o -json command options.

You can also enable the `hh:mm:ss` time format as the default time format for customized job information by specifying the LSB_HMS_TIME_FORMAT parameter in the lsf.conf file, or by specifying the LSB_HMS_TIME_FORMAT environment variable.

If these parameters or options are not set, the default output time for customized output is in seconds.

Related reference

- **bjobs -hms** command option
- LSB_HMS_TIME_FORMAT parameter in the lsf.conf file

Security

The following new features affect cluster security.

Improve security and authentication by updating the eauth executable file

LSF now includes an updated version of the **eauth** executable file that automatically generates a site-specific internal key by using 128-bit AES encryption. To use this updated version, you must replace the original **eauth** executable file with the new file.

Related reference

- Authentication

What's new in IBM Spectrum LSF Version 10.1 Fix Pack 1

The following topics summarize the new and changed behavior in LSF 10.1 Fix Pack 1

Release date: 11 November 2016

Simplified affinity requirement syntax

Job submission with affinity requirements for LSF jobs is simplified. An **esub** script that is named esub.p8aff is provided to generate optimal affinity requirements based on the input requirements about the submitted affinity jobs. In addition, LSF supports OpenMP thread affinity in the **blaunch** distributed application framework. LSF MPI distributions must integrate with LSF to enable the OpenMP thread affinity.

For the generated affinity requirements, LSF tries to reduce the risk of CPU bottlenecks for the CPU allocation in LSF MPI task and OpenMP thread levels.

For more information, see Submit jobs with affinity resource requirements on IBM POWER8 systems.

bsub and bmod commands export memory and swap values as esub variables

Specifying mem and swp values in an `rusage[]` string tell LSF how much memory and swap space a job requires, but these values do no limit job resource usage.

The **bsub** and **bmod** commands can export mem and swp values in the `rusage[]` string to corresponding environment variables for **esub**. You can use these environment variables in your own **esub** to match memory and swap limits with the values in the `rusage[]` string. You also can configure your **esub** to check whether the memory and swap resources are correctly defined for the corresponding limits for the job, queue, or application. If the resources are not correctly defined, LSF rejects the job.

The following environment variables are exported:

- If the **bsub** or **bmod** command has a mem value in the rusage[] string, the LSB_SUB_MEM_USAGE variable is set to the mem value in the temporary **esub** parameter file that the LSB_SUB_PARAM_FILE environment variable points to. For example, if the **bsub** command has the option `-R "rusage[mem=512]"`, the LSB_SUB_MEM_USAGE=512 variable is set in the temporary file.
- If the **bsub** or **bmod** command has a swp value in the rusage[] string, the LSB_SUB_SWP_USAGE variable is set to the mem value in the temporary **esub** parameter file that the LSB_SUB_PARAM_FILE environment variable points to. For example, if the **bsub** command has the option `-R "rusage[swp=1024]"`, the LSB_SUB_SWP_USAGE=1024 variable is set in the temporary file.

For more information on LSB_SUB_MEM_USAGE or LSB_SUB_SWP_USAGE, see Configuration to enable job submission and execution controls.

Allow queues to ignore RETAIN and DURATION loan policies

The LOAN_POLICIES parameter in the lsb.resources file allows other jobs to borrow unused guaranteed resources LSF. You can enable queues to ignore the RETAIN and DURATION loan policies when LSF determines whether jobs in those queues can borrow unused guaranteed resources. To enable the queue to ignore the RETAIN and DURATION loan policies, specify an exclamation point (`!`) before the queue name in the LOAN_POLICIES parameter definition.

For more information, see Configuration overview of guaranteed resource pools.

Running LSF jobs in Docker containers

The Docker integration allows LSF to run jobs in Docker containers on demand. LSF manages the entire lifecycle of jobs that run in the container as common jobs.

LSF supports the use of Docker Engine, Version 1.12, or later, which must be installed on an LSF server host.

For more information, see with Docker.

Running LSF jobs in Amazon Web Services instances

You can configure LSF to make allocation requests on from Amazon Web Services (AWS). With AWS configured as a resource provider in LSF resource connector, LSF can launch instances from AWS to satisfy

pending workload. The AWS instances join the LSF cluster, and are terminated when they become idle.

LSF resource connector with AWS was tested on the following systems:

- LSF10.1 master host - Linux x86 Kernel 3.10, glibc 2.17 RHEL 7.x
- VMs - Linux x86 Kernel 3.10, glibc 2.17 CentOS 7.x

LSF resource connector with AWS is assumed to work on the following systems:

- IBM Spectrum LSF10.1
- Linux x86 Kernel 2.6, glibc 2.5 RHEL 5.x
- Linux x86 Kernel 2.6, glibc 2.11 RHEL 6.x
- Linux x86 Kernel 3.0, glibc 2.11 SLES 11.x
- Linux x86 Kernel 3.11, glibc 2.18 SLES 12.x
- Linux x86 Kernel 4.4, glibc 2.23 Ubuntu 16.04 LTS

For more information, see Using the Resource Connector.

Job array performance enhancements

The performance of job array scheduling and execution is improved.

The performance of scheduling, dispatch, and execution of job array elements is affected when array elements are split from their original submitted array under various conditions. For example, if rerunnable array elements are dispatched but fail to run, the elements return to pending state. The LSF scheduler has already split these elements when job was dispatched to execution hosts. The split array elements can remain pending for an excessive amount of time.

For an array jobs with dependency conditions, LSF publishes separate job ready events to the scheduler for each element when the condition is satisfied. The scheduler splits the elements when it handles the job ready events.

The following performance improvements are made:

- Optimized recovery performance in the scheduler for jobs with many separate array elements.
- Improved handling of satisfied dependency conditions for array jobs.
- Improved dependency checking for array jobs to reduce the number of job ready events that are published to the scheduler.
- Improved the processing of events for multiple array elements for job ready event handling.
- Optimized event handling performance in the scheduler for array jobs with many split elements
- Improved handing job stop and resume, and events associated with moving jobs to the top and bottom of the queue with the **bbot** and **btop** commands.

New platform support

LSF supports the following platforms:

- Intel Knights Landing (Linux x86-64 packages)

What's new in IBM Spectrum LSF Version 10.1

The following topics summarize the new and changed behavior in LSF 10.1.

Release date: 2 June 2016

Important: IBM Platform Computing is now renamed to IBM Spectrum Computing to complement IBM's Spectrum Storage family of software-defined offerings. The IBM Platform LSF product is now IBM Spectrum LSF. Some LSF documentation in IBM Knowledge Center (http://www.ibm.com/support/knowledgecenter/SSWRJV_10.1.0) does not yet reflect this new product name.

Performance enhancements

The following are the new features in LSF 10.1 that can improve performance.

General performance improvements

Scheduler efficiency
: LSF 10.1.0 includes several binary-level and algorithm-level optimizations to help the scheduler to make faster decisions. These enhancements can make job scheduling less sensitive to the number of job buckets and resource requirement settings.

Daemon communication
: LSF 10.1.0 makes optimizations to **mbatchd**/**sbatchd** communication protocols to ensure a dedicated channel to accelerate messages that are sent and received between the **mbatchd** and **sbatchd** daemons.

Improved scheduling for short jobs

LSF can now allow multiple jobs with common resource requirements to run consecutively on the same allocation. Whenever a job finishes, LSF attempts to quickly replace it with a pending job that has the same resource requirements. To ensure that limits are not violated, LSF selects pending jobs that belong to the same user and have other attributes in common.

Since LSF bypasses most of the standard scheduling logic between jobs, reusing resource allocation can help improve cluster utilization. This improvement is most evident in clusters with several shorter jobs (that is, jobs that run from a few seconds to several minutes) with the same resource requirements.

To ensure that the standard job prioritization policies are approximated, LSF enforces a limit on the length of time that each allocation is reusable. LSF automatically sets this time limit to achieve a high level of resource utilization. By default, this reuse time cannot exceed 30 minutes. If you specify a maximum reuse time and an optional minimum reuse time with the ALLOC_REUSE_DURATION parameter, LSF adjusts the time limit within this specified range to achieve the highest level of resource utilization.

When jobs from job arrays reuse allocations, the dispatch order of these jobs might change. Dispatch order changes because jobs are chosen for allocation reuse based on submission time instead of other factors.

Advance reservations are not considered when the job allocation is reused. A job allocation that is placed on a host with advance reservations enabled cannot be reused. If an advance reservation is created on a host after the job allocation is already made, the allocation can still be reused until the reuse duration is expired or the job is suspended by the advance reservation policy.

To enable LSF to reuse the resource allocation, specify the RELAX_JOB_DISPATCH_ORDER parameter in the lsb.params file. To enable reuse for a specific queue, specify the RELAX_JOB_DISPATCH_ORDER parameter in the lsb.queues file. The RELAX_JOB_DISPATCH_ORDER parameter is now defined as `Y` at installation.

Use the **badmin perfmon view** command to show the number of jobs that are reordered as a result of this feature.

When the RELAX_JOB_DISPATCH_ORDER parameter is specified, changing job group limits is not supported.

Cluster performance improvement with job information cache

LSF has a new job information cache to reduce the load on the work directory file server. LSF caches job information such as job environment variables and data in memory from the command-line and **eexec** in a compressed format. If you have an environment with many commonly used environment variable settings, caching job information can improve job submission and job dispatch performance, especially when the work directory's shared file system is slow or at its limits.

The job information cache is enabled by default in LSF 10.1, and the default size of the lsb.jobinfo.events file is 1 GB. New job information is now stored in the new event file instead of individual job files.

The contents of the cache persist in the job information event file, which is located by default at $LSB_SHAREDIR/cluster_name/logdir/lsb.jobinfo.events. The location of the lsb.jobinfo.events file can be changed with the parameter LSB_JOBINFO_DIR in lsf.conf.

The amount of memory that is dedicated to the cache is controlled by the lsb.params parameter JOB_INFO_MEMORY_CACHE_SIZE.

As jobs are cleaned from the system, the lsb.jobinfo.events event file needs to be periodically rewritten to discard the unneeded data. By default, the job information event file is rewritten every 15 minutes. This interval can be changed with the parameter JOB_INFO_EVENT_DUMP_INTERVAL in the lsb.params file.

The values of the parameters JOB_INFO_MEMORY_CACHE_SIZE and JOB_INFO_EVENT_DUMP_INTERVAL can be viewed with the command **bparams -a** or **bparams -l**

The amount of memory that is used by the job information cache can be viewed with the command **badmin showstatus**.

Job array performance improvements

The algorithm that is used to process large job array operations is enhanced. The time to process multiple array elements in the **mbatchd** daemon and the scheduler is reduced. The processing of job array operations in the **mbatchd** daemon, log events, and publishing job events to the scheduler is more efficient. The performance and behavior of the **bmod**, **bkill**, **bresume**, **bstop**, **bswitch**, **btop**, and **bbot** commands has been improved.

The parameter JOB_ARRAY_EVENTS_COMBINE in the lsb.params file enables the performance improvements for array jobs. The formats of some event types are changed to include new fields in lsb.events, lsb.acct, lsb.stream, and lsb.status files.

The parameter JOB_ARRAY_EVENTS_COMBINE makes the parameter JOB_SWITCH2_EVENT in the lsb.params file obsolete.

Pending job management

The following new features improve the management of pending jobs.

Single pending reason

Previously, a main pending reason or a series of host-based pending reasons was given when a job cannot run. The main reason is given if the job is pending for a reason that is not related to single hosts before or during scheduling, or if it failed to dispatch or run on the allocated host after scheduling. If the job is eligible to be scheduled but no host can be allocated, the pending reason is host-based for every host, to indicate why the host cannot be used. However, this pending reason might mean that the host-based pending reasons are numerous and shown in any random order, making it difficult for users to decipher why their job does not run. This problem is especially true for large clusters.

To make the given pending reason both precise and succinct, this release introduces the option to choose a single key reason for why the job is pending. Host-based pending reasons are classified into categories, and only the top reason in the top category is shown, or a main pending reason.

Host-based pending reasons are now grouped into reasons of candidate hosts and reasons of non-candidate hosts. Reasons for non-candidate hosts are not important to users since they cannot act on them. For example, the reason `Not specified in job submission` might be given for a host that was filtered out by the user with the **bsub -m** command. In contrast, reasons for candidate hosts can be used by the user to get the job to run. For example, with the reason `Job's resource requirement for reserving resource (mem) not satisfied`, you can lower the job's memory requirement.

The new option **bjobs -p1** is introduced in this release to retrieve the single reason for a job. If the single key pending reason is a host-based reason, then the single reason and the corresponding number of hosts is shown. Otherwise, only the single reason is shown.

Note: If the main reason is the only host-based reason, the main reason is shown as the output of the **bjobs -p2** and **bjobs -p3** commands.

Categorized host-based pending reasons

To give users a better understanding of why their jobs are not running, and what they can do about it, LSF groups host-based pending reasons into two categories: reasons of candidate hosts, and reason of non-candidate hosts.

The new options **bjobs -p2** and **bjobs -p3** are introduced in this release.

Option **bjobs -p2** shows the total number of hosts in the cluster and the total number considered. For the hosts considered, the actual reason on each host is shown. For each pending reason, the number of hosts that give that reason is shown. The actual reason messages appear from most to least common.

Option **bjobs -p3** shows the total number of hosts in the cluster and the total number of candidate and non-candidate hosts. For both the candidate and non-candidate hosts, the actual pending reason on each host is shown. For each pending reason, the number of hosts that show that reason is given. The actual reason messages appear from most to least common.

Note:

- If the main reason is the only host-based reason, the main reason is shown as the output of the **bjobs -p2** and **bjobs -p3** commands.
- If the LSB_SKIP_FULL_HOSTS parameter is enabled (set to `Y`) in the lsf.conf file, LSF does not consider the `closed_Full` hosts as candidate hosts. For more details, refer to the LSB_SKIP_FULL_HOSTS parameter in the lsf.conf file.

bjobs -o "pend_reason"

Many customers use the **bjobs –u all** or **bjobs –l –u all** commands to get all information, then use a script to search through the output for the required data. The command **bjobs –o 'fmtspec'** also allows users to request just the fields that they want, and format them so that they are readily consumable.

With the continuing effort to enhance pending reasons, the new field **pend_reason** is introduced in this release to show the single (main) pending reason, including custom messages.

Configurable pending reason message and resource priority with the lsb.reasons file

This release introduces the ability to individually configure pending reason messages. Administrators can make messages clear to inform users on which action they can take to make the job run. Configure custom pending reasons in the new configuration file, **config/lsbatch/*<cluster_name>*/configdir/lsb.reasons**.

Detailed pending reasons

Reasons for why a job is pending are displayed by using the **bjobs** command, but in many cases the **bjobs** command provides only general messages for why the job is pending. The reasons do not include enough details and users might not know how to proceed. For example, the pending reason `The specified job group has reached its job limit` does not clarify which job group limit within the hierarchical tree is at its limit.

Greater detail is added to pending reason messages. Display includes, where applicable, host names, queue names, job group names, user group names, limit name, and limit value as part of the pending reason message.

The enhanced pending reason information is shown by the **bjobs** command with the -p1, -p2, and -p3 options. If the LSB_BJOBS_PENDREASON_LEVEL parameter in the lsf.conf file is set to 1, 2, or 3, the new information is shown by the **bjobs -p** command. The pending reason information is not included for the **bjobs -p0** command.

Pending reason summary

A new option, -psum, is introduced to the **bjobs** command. The -psum option displays a summary of current pending reasons. It displays the summarized number of jobs, hosts, and occurrences for each pending reason.

It can be used with the filter options that return a list of pending jobs: -p, -p(0~3), -pi, -pe, -q, -u, -G, -g, -app, -fwd, -J, -Jd, -P, -Lp, -sla, -m

The command `bjobs -psum` lists the top eligible and ineligible pending reasons in descending order by the number of jobs. If a host reason exists, further detailed host reasons are displayed in descending order by occurrences. Occurrence is a per-job per-host based number, counting the total times that each job hits the reason on every host.

Pending reason performance improvements

With this release, performance problems that are associated with displaying pending reasons are improved. Now, reasons for all jobs in a bucket are published (instead of only the top jobs in the bucket) at every interval that is specified by the PEND_REASON_UPDATE_INTERVAL parameter in the lsb.params file. Host-based reasons publishing performance is improved to support up to 20,000 buckets and 7,500 hosts without the need to enable the CONDENSE_PENDING_REASONS parameter or to use the **badmin diagnose** command.

Job start time estimation

In clusters with long running parallel jobs (such as HPC environments), a few long running jobs (that is, 100 - 1000 jobs) might be pending in the queue for several days. These jobs might run for several days or weeks.

LSF can now predict an approximate start time for these pending jobs by using a simulation-based job start time estimator that runs on the master host and is triggered by the **mbatchd** daemon. The estimator uses a snapshot of the cluster (including the running jobs and available resources in the cluster) to simulate job scheduling behavior. The estimator determines when jobs finish and the pending jobs start. This snapshot gives users an idea of when their jobs are expected to start.

To use simulation-based estimation to predict start times, jobs must be submitted with either a runtime limit (by using the **bsub -W** option or by submitting to a queue or application profile with a defined RUNLIMIT value) or an estimated run time (by using the **bsub -We** option or by submitting to an application profile with a defined RUNTIME value). LSF considers jobs without a runtime limit or an estimated run time as never finished after they are dispatched to the simulation-based estimator. If both a runtime limit and an estimated run time are specified for a job, the smaller value is used as the job's run time in the simulation-based estimator.

To enable the simulation-based estimator, define the LSB_ENABLE_ESTIMATION=Y parameter in the lsf.conf file. When LSB_ENABLE_ESTIMATION=Y is set, the estimator starts up 5 minutes after the **mbatchd** daemon starts or restarts. By default, the estimator provides predictions for the first 1000 jobs or for predicted start times up to one week in the future, whichever comes first. Estimation also ends when all pending jobs have prediction job start times.

Optionally, you can control the default values for when **mbatchd** stops the current round of estimation to balance the accuracy of the job start predictions against the computation effort on the master host. **mbatchd** stops the current round of estimation when the estimator reaches any one of the following estimation thresholds that are specified in lsb.params:

ESTIMATOR_MAX_JOBS_PREDICTION
: Specifies the number of pending jobs that the estimator predicts, which is 1000 by default.

ESTIMATOR_MAX_TIME_PREDICTION
: Specifies the amount of time into the future, in minutes, that a job is predicted to start before the estimator stops the current round of estimation. By default, the estimator stops after a job is predicted to start in one week (10080 minutes).

ESTIMATOR_MAX_RUNTIME_PREDICTION

Specifies the amount of time that the estimator runs, up to the value of the ESTIMATOR_SIM_START_INTERVAL parameter. By default, the estimator stops after it runs for 30 minutes or the amount of time as specified by the ESTIMATOR_SIM_START_INTERVAL parameter, whichever is smaller.

The estimator does not support the following **badmin** subcommands: **mbddebug**, **schddebug**, **mbdtime**, and **schdtime**. The estimator reloads the configurations from the lsf.conf file after it starts.

Eligible and ineligible pending jobs

LSF can now determine whether pending jobs are eligible or ineligible for scheduling.

A job that is in an eligible pending state is a job that LSF would normally select for resource allocation, but is pending because its priority is lower than other jobs. It is a job that is eligible for scheduling and runs if sufficient resources are available to run it.

An ineligible pending job is ineligible for scheduling and remains pending even if enough resources are available to run it. A job can remain pending and be ineligible to run for the following reasons:

- The job has a start time constraint (specified with the -b option)
- The job is suspended while it is pending (in a PSUSP state).
- The queue of the job is made inactive by the administrator or by its time window.
- The job's dependency conditions are not satisfied.
- The job cannot fit into the runtime window (RUN_WINDOW parameter)
- Delayed scheduling is enabled for the job (the NEW_JOB_SCHED_DELAY parameter is greater than zero)
- The job's queue or application profile does not exist.

A job that is not under any of the ineligible pending state conditions is treated as an eligible pending job. In addition, for chunk jobs in `WAIT` status, the time that is spent in the `WAIT` status is counted as eligible pending time.

If the TRACK_ELIGIBLE_PENDINFO parameter in the lsb.params file is set to `Y` or `y`, LSF determines which pending jobs are eligible or ineligible for scheduling. LSF uses the eligible pending time instead of total pending time to determine job priority for the following time-based scheduling policies:

- Automatic job priority escalation increases job priority of jobs that are in an eligible pending state instead of pending state for the specified period.
- For absolute priority scheduling (APS), the JPRIORITY subfactor for the APS priority calculation uses the amount of time that the job spends in an eligible pending state instead of the total pending time.

The **mbschd** daemon saves eligible and ineligible pending information to disk every 5 minutes. The eligible and ineligible pending information is recovered when the **mbatchd** daemon restarts. When the **mbatchd** daemon restarts, some ineligible pending time might be lost since it is recovered from the snapshot file, which is dumped periodically at set intervals. The lost time period is counted as eligible pending time under such conditions. To change this time interval, specify the ELIGIBLE_PENDINFO_SNAPSHOT_INTERVAL parameter, in minutes, in the lsb.params file.

Pending time limits

You can specify pending time limits and eligible pending time limits for jobs.

LSF sends the pending time limit and eligible pending time limit configurations to IBM® Spectrum LSF RTM, which handles the alarm and triggered actions such as user notification. For example, RTM can notify the user

who submitted the job and the LSF administrator, and take job control actions (for example, killing the job). LSF RTM compares the job's pending time to the pending time limit, and the eligible pending time to the eligible pending time limit. If the job is in a pending state or an eligible pending state for longer than these specified time limits, LSF RTM triggers the alarm and actions. This parameter works without LSF RTM, but LSF does not take any other alarm actions.

To specify a pending time limit or eligible pending time limit at the queue or application level, define the PEND_TIME_LIMIT or ELIGIBLE_PEND_TIME_LIMIT parameters in lsb.queues or lsb.applications. To specify the pending time limit or eligible pending time limit at the job level, use the -ptl or -eptl options for **bsub** and **bmod**:

- `PEND_TIME_LIMIT=[`*hour*`:]`*minute*
- `ELIGIBLE_PEND_TIME_LIMIT=[`*hour*`:]`*minute*
- `-ptl [`*hour*`:]`*minute*
- `-eptl [`*hour*`:]`*minute*

The pending or eligible pending time limits are in the form of [*hour*:]*minute*. The minutes can be specified as a number greater than 59. For example, three and a half hours can either be specified as 3:30, or 210.

The job-level time limits override the application-level time limits, and the application-level time limits override the queue-level time limits.

LSF does not take any alarm actions. However, LSF users and administrators can track the amount of time that jobs spend in pending or eligible pending states, and whether the jobs reach the pending time limits:

The -l option for **bjobs**, **bapp**, and **bqueues** show the job-, application-, and queue-level pending time limits (and eligible pending time limits).

To track the amount of time that current pending jobs spend in the pending and eligible pending states, and to see how much time is remaining before LSF sends an alarm notification, run the **bjobs -p -o** command to get customized output for pending jobs.

- Pending time limit

```
bjobs -p -o "id effective_plimit plimit_remain"
JOBID EFFECTIVE_PLIMIT PLIMIT_REMAIN
101   1800             -60
102   3600              60
```

- Eligible pending time limit

```
bjobs -p -o "id effective_eplimit eplimit_remain"
JOBID EFFECTIVE_EPLIMIT EPLIMIT_REMAIN
101   600               -60
102   900                60
```

The `EFFECTIVE_PLIMIT` and `EFFECTIVE_EPLIMIT` columns indicate the pending and eligible pending time limits for the job. The `PLIMIT_REMAIN` and `EPLIMIT_REMAIN` columns display the amount of time that remains before LSF sends an alarm notification. A negative number indicates that the time limit was reached and shows the amount of time since the limit was reached.

Job scheduling and execution

The following new features affect job scheduling and execution.

Global fair share scheduling policy

Many LSF customers run clusters in geographic sites that are connected by LSF multicluster capability to maximize resource utilization and throughput. Most customers configure hierarchical fair share to ensure resource fairness among projects and users. The same fair share tree can be configured in all clusters for the same organization because users might be mobile and can log in to multiple clusters. However, fair share is local to each cluster and resource usage might be fair in the context of one cluster, but unfair from a more global perspective.

The LSF global fair share scheduling policy divides the processing power of IBM® Spectrum LSF multicluster capability (LSF multicluster capability) and the LSF/XL feature of IBM Spectrum LSF Advanced Edition among users. The global fair share scheduling policy provides fair access to all resources, making it possible for every user to use the resources of multiple clusters according to their configured shares.

Global fair share is supported in IBM Spectrum LSF Standard Edition and IBM Spectrum LSF Advanced Edition.

Global fair share scheduling is based on queue-level user-based fair share scheduling. LSF clusters that run in geographically separate sites that are connected by LSF multicluster capability can maximize resource utilization and throughput.

Global fair share supports the following types of fair share scheduling policies:

- Queue level user-based fair share
- Cross-queue user-based fair share
- Parallel fair share

In cross-queue user-based fair share policies, you configure the master queue as a participant of global fair share. Participants can be any queues, users, or user groups that participate in the global fair share policy. Configuring a slave queue as a participant is not needed, since it does not synchronize data for the global fair share policy.

For parallel fair share, LSF can consider the number of CPUs when you use global fair share scheduling with parallel jobs.

Resource connector for LSF

The resource connector for LSF feature (also called *host factory*) enables LSF clusters to borrow resources from supported resource providers (for example, enterprise grid orchestrator or OpenStack based on workload.

The resource connector generates requests for extra hosts from a resource provider and dispatches jobs to dynamic hosts that join the LSF cluster. When the resource provider needs to reclaim the hosts, the resource connector re-queues the jobs that are running on the LSF hosts, shuts down LSF daemons, and releases the hosts back to the resource provider.

Use the **bsub** command to submit jobs that require hosts that are borrowed from resource provider. Use the **bhosts** command to monitor the status of borrowed hosts.

LSF with Apache Hadoop

The IBM Spectrum LSF integration with Apache Hadoop provides a connector script that allows users to submit Hadoop applications as regular LSF jobs.

Apache Hadoop ("Hadoop") is a framework for large-scale distributed data storage and processing on computer clusters that uses the Hadoop Distributed File System ("HDFS") for the data storage and MapReduce programming model for the data processing. Since MapReduce workloads might represent only a small fraction of overall workload, but typically requires their own stand-alone environment, MapReduce is difficult to support within traditional HPC clusters. However, HPC clusters typically use parallel file systems that are sufficient for initial MapReduce workloads, so you can run MapReduce workloads as regular parallel jobs that run in an HPC cluster environment. Use the IBM Spectrum LSF integration with Apache Hadoop to submit Hadoop MapReduce workloads as regular LSF parallel jobs.

To run your Hadoop application through LSF, submit it as an LSF job. After the LSF job starts to run, the **blaunch** command automatically provisions and monitors an open source Hadoop cluster within LSF allocated resources, then submits actual MapReduce workloads into this Hadoop cluster. Since each LSF Hadoop job has its own resource (cluster), the integration provides a multi-tenancy environment to allow multiple users to share the common pool of HPC cluster resources. LSF is able to collect resource usage of MapReduce workloads as normal LSF parallel jobs and has full control of the job lifecycle. After the job is complete, LSF shuts down the Hadoop cluster.

By default, the Apache Hadoop integration configures the Hadoop cluster with direct access to shared file systems and does not require HDFS. You can use existing file systems in your HPC cluster without having to immediately invest in a new file system. Through the existing shared file system, data can be stored in common share locations, which avoids the typical data stage-in and stage-out steps with HDFS.

LSF with Apache Spark

The IBM Spectrum LSF integration with Apache Spark provides connector scripts that allow users to submit Spark applications as regular LSF jobs.

Apache Spark ("Spark") is an in-memory cluster computing system for large-scale data processing. Based on Apache Hadoop ("Hadoop"), it provides high-level APIs in Java, Scala and Python, and an optimized engine that supports general execution graphs. It also provides various high-level tools, including Spark SQL for structured data processing, Spark Streaming for stream processing, and Mlib for machine learning.

Spark applications require distributed computed nodes, large memory, a high-speed network, and no file system dependencies, so Spark applications can run in a traditional HPC environment. Use the IBM Spectrum LSF integration with Apache Spark to take advantage of the comprehensive LSF scheduling policies to allocate resources for Spark applications. LSF tracks, monitors, and controls the job execution.

To run your Spark application through LSF, submit it as an LSF job, and the scheduler allocates resources according to the job's resource requirements, while the **blaunch** command starts a stand-alone Spark cluster. After the job is complete, LSF shuts down the Spark cluster.

Resizable jobs with resource requirements

LSF now allows the following resource requirements with resizable jobs:

- Alternative resource requirements
- Compound resource requirements
- Compute unit requirements

When you use the **bresize release** command to release slots from compound resource requirements, you can release only the slots that are represented by the last term of the compound resource requirement. To release slots in earlier terms, run **bresize release** repeatedly to release slots in subsequent last terms.

In addition, autoresizable jobs can now be submitted with compute unit resource requirements. The `maxcus` keyword is enforced across the job's entire allocation as it grows, while the `balance` and `usablecuslots` keywords apply only to the initial resource allocation.

For example,

- `bsub -n 11,60 -R "cu[maxcus=2:type=enclosure]" -app resizable -ar myjob`
  An automatically resizable job that spans the fewest possible compute units for a total allocation of at least 11 slots that use at most 2 compute units of type enclosure. If the automatically job grows, the entire job still uses at most 2 compute units of type enclosure.

- `bsub -n 64 -R "cu[balance:maxcus=4:type=enclosure]" -app resizable -ar myjob`
  An automatically resizable job that spans the fewest possible compute units for a balanced allocation of 64 slots that use 4 or less compute units of type enclosure. If the automatically job grows, each subsequent allocation is a balanced allocation. The entire job (that is, the total of the initial and subsequent job allocations) still uses at most 4 compute units of type enclosure, but the job as a whole might not be a balanced allocation.

- `bsub -n 64 -R "cu[excl:maxcus=8:usablecuslots=10]" -app resizable -ar myjob`
  An automatically job that allocates 64 slots over 8 or less compute units in groups of 10 or more slots per compute unit. One compute unit possibly uses fewer than 10 slots. If the automatically job grows, each subsequent allocation allocates in groups of 10 or more slots per compute unit (with one compute unit possible using fewer than 10 slots). The entire job (that is, the total of the initial and subsequent job allocations) still uses at most 8 compute units. Since each subsequent allocation might have one compute unit that uses fewer than 10 slots, the entire job might have more than one compute unit that uses fewer than 10 slots. The default compute unit type set in the COMPUTE_UNIT_TYPES parameter is used, and is used exclusively by `myjob`.

Specifying compute unit order by host preference

Previously, the compute unit order was determined only by the compute unit `pref` policies (`cu[pref=config | maxavail | minavail]`). Host preference (specified by `-m` or the HOSTS parameter in the `lsb.queues` file) only affected the host order within each compute unit. This release allows the user to specify compute unit order in a more flexible manner, by host preference. LSF now allows use of the host preference to specify compute unit order along with the `cu[pref=config | maxavail | minavail]` policy.

The following example illustrates use of the `-m` preference to specify compute unit order as `cu1>cu2>cu3>cu4`

```
bsub -n 2 -m "cu1+10 cu2+5 cu3+1 cu4" -R "cu[]" ./app
```

Sorting forwarded jobs by submission time

The parameter MC_SORT_BY_SUBMIT_TIME is added to the lsb.params file. Enabling this parameter in a IBM Spectrum LSF multicluster capability environment allows forwarded jobs on the execution cluster to be sorted and run based on their original submission time (instead of their forwarded time). When the maximum rescheduled time is reached, pending jobs are rescheduled on the execution cluster. Pending jobs are ordered based on their original submission time (the time when the job was first submitted on the submission cluster) and not the forwarding time (the time when the job was reforwarded to the execution cluster).

Compute unit feature functions with the alternative and compound resource requirements

This release now supports compute unit (`cu`) strings in alternative and compound resource requirements except you use the `cu` keywords `excl` or `balance`. Other `cu` keywords (such as `type`, `pref`, `maxcus`, or `usablecuslot`) are fully supported. Jobs are rejected if the merged result of the queue-, application-, and job-level resource requirement is compound or alternative with `cu[excl]` or `cu[balance]`.

External post-submission with epsub

Using the same mechanism for external job submission executable files (**esub**), you can now specify post-submission executable files to run after a job is submitted. An **epsub** is an executable file that you write to meet the post-submission job requirements at your site with information that is not available before job submission. The following are some of the things that you can use an **epsub** to do:

- Pass job information to an external entity
- Post job information to a local log file
- Perform general logic after a job is submitted to LSF

When a user submits a job by using **bsub**, and modifies a job by using the **bmod** command, or restarts a job by using the **brestart** command, LSF runs the **epsub** executable files on the submission host immediately after the job is accepted. The job might or might not be running while **epsub** is running.

For interactive jobs, **bsub** or **bmod** runs **epsub**, then resumes regular interactive job behavior (that is, **bsub** or **bmod** runs **epsub**, then runs the interactive job).

The **epsub** file does not pass information to **eexec**, nor does it get information from **eexec**. **epsub** can read information only from the temporary file that contains job submission options (as indicated by the LSB_SUB_PARM_FILE environment variable) and from the environment variables. The following information is available to the **epsub** after job submission:

- A temporary file that contains job submission options, which are available through the LSB_SUB_PARM_FILE environment variable. The file that this environment variable specifies is a different file from the one that is initially created by **esub** before the job submission.
- The LSF job ID, which is available through the LSB_SUB_JOB_ID environment variable. For job arrays, the job ID includes the job array index.
- The name of the final queue to which the job is submitted (including any queue modifications that are made by **esub**), which is available through the LSB_SUB_JOB_QUEUE environment variable.
- The LSF job error number if the job submission failed, which is available through the LSB_SUB_JOB_ERR environment variable.

If the **esub** rejects a job, the corresponding **epsub** file does not run.

After job submission, the **bsub** or **bmod** command waits for the **epsub** scripts to finish before it returns. If the **bsub** or **bmod** return time is crucial, do not use **epsub** to perform time-consuming activities. In addition, if **epsub** hangs, **bsub** or **bmod** waits indefinitely for the **epsub** script to finish. This behavior is similar to the **esub** behavior because **bsub** or **bmod** hangs if an **esub** script hangs.

If an LSF administrator specifies one or more mandatory **esub**/**epsub** executable files that use the parameter LSB_ESUB_METHOD, LSF starts the corresponding mandatory **epsub** executable files (as specified by using the parameter LSB_ESUB_METHOD), followed by any application-specific **epsub** executable files (with *.application_name* in the file name).

If a mandatory program that is specified by the LSB_ESUB_METHOD parameter does not have a corresponding **esub** executable file (esub.*application_name*), but has a corresponding **epsub** executable file (epsub.*application_name*), the job is submitted normally by using the normal external job submission and post-submission mechanisms.

Except for these differences, **epsub** uses the same framework as **esub**.

Save a snapshot of the job scheduler buckets

LSF can now save a snapshot of the current contents of the scheduling buckets to help administrators diagnose problems with the scheduler. Jobs are put into scheduling buckets based on resource requirements and different scheduling policies. Saving the contents into a snapshot file is useful for data analysis by parsing the file or by performing a simple text search on its contents.

This feature is helpful if you want to examine a sudden large performance impact on the scheduler. Use the snapshot file to identify any users with many buckets or large attribute values.

To use this feature, run the **badmin diagnose -c jobreq** command.

This feature enables **mbschd** to write an active image of the scheduler job buckets into a snapshot file as raw data in XML or JSON format. A maximum of one snapshot file is generated in each scheduling cycle.

Use the -f option to specify a custom file name and path and the -t option to specify whether the file is in XML or JSON format.

By default, the name of the snapshot file is jobreq_*<hostname>*_*<dateandtime>*.*<format>*, where *<format>* is xml or json, depending on the specified format of the snapshot file. By default, the snapshot file is saved to the location specified in the DIAGNOSE_LOGDIR parameter.

Using logging threads to log messages

The **mbatchd** and **mbschd** daemons now use dedicated threads to write messages to the log files. Using dedicated threads reduces the impact of logging messages on the performance of **mbatchd** and **mbschd**.

Define the `LSF_LOG_QUEUE_SIZE=`*`integer`* parameter in the lsf.conf file as an integer between 100 and 500000 to specify the maximum size of the logging queue. The logging queue, which contains the messages to be logged in the log files, is full when the number of entries reaches this number.

Define the LSF_DISCARD_LOG parameter in the lsf.conf file to specify the behavior of the logging thread if the logging queue is full. If set to `Y`, the logging thread discards all new messages at a level lower than LOG_WARNING when the logging queue is full. LSF logs a summary of the discarded messages later.

If the LSF_DISCARD_LOG parameter is set to `N`, LSF automatically extends the size of the logging queue if the logging queue is full.

Specifying resource requirements for stopped checkpointable jobs

The **brestart** command now includes the -R option to reserve resources when you restart a stopped checkpointable job. You can specify resources with **brestart -R** when you restart the job. Specify multiple -R options on the **brestart** command for multiple resource requirement strings, compound resource requirements, and alternative resource requirements.

For example, if you submitted the following checkpointable job:

```
bsub -R "select[mem>100] rusage[mem=100]" -M 100 myjob
```

You can restart this checkpointable job by using the **brestart -R** command to specify a new resource requirement:

```
brestart -R "select[mem>5000] rusage[mem=5000]" -M 5000 checkpointdir/pid
```

No size limitations for resource requirement strings

LSF no longer has any size limitations on resource requirement strings. Previously, resource requirement strings were restricted to 512 bytes. You can now submit resource requirement strings with the -R option with no limitations on the length of the string.

You must upgrade all hosts in the cluster to LSF 10.1 to submit resource requirement strings with no size limitations. If hosts in the cluster still run earlier versions of LSF, resource requirement strings still have the following limitations:

- In the IBM Spectrum LSF multicluster capability job forwarding mode, if the execution cluster is running an earlier version of LSF:
  - Any jobs with a job-level resource requirement string that is longer than 511 bytes remain pending on the submission cluster.
  - LSF rejects any **bmod** commands that modify a job that is forwarded to the execution cluster with a job-level resource requirement string that is longer than 511 bytes.
- If you run the **bjobs** command from a host with an earlier version of LSF and the job-level resource requirement string is longer than 4096 bytes, the **bjobs -l** command output shows a truncated resource requirement string.
- If you run the **bacct** or **bhist** commands from a host with an earlier version of LSF and the effective resource requirement string is longer than 4096 bytes, the command might fail.

Host-related features

The following new features are related to host management and display.

Condensed host format

When you specify host names or host groups with condensed notation, you can now use colons (`:`) to specify a range of numbers. Colons are used the same as hyphens (`-`) are currently used to specify ranges and can be used interchangeably in condensed notation. You can also use leading zeros to specify host names.

You can now use multiple square brackets (with the supported special characters) to define multiple sets of non-negative integers anywhere in the host name. For example, `hostA[1,3]B[1-3]` includes `hostA1B1`, `hostA1B2`, `hostA1B3`, `hostA3B1`, `hostA3B2`, and `hostA3B3`.

The additions to the condensed notation apply to all cases where you can specify condensed notation, including commands that use the -m option or a host list to specify multiple host names, the lsf.cluster.*clustername* file (in `HOSTNAME` column of the `Hosts` section), and the lsb.hosts file (in the `HOST_NAME` column of the `Host` section, the `GROUP_MEMBER` column of the `HostGroup` section, and the `MEMBER` column of the `ComputeUnit` section).

For example, submit a job by using the **bsub -m** command.

- `bsub -m "host[1-100].example.com"`
  The job is submitted to host1.example.com, host2.example.com, host3.example.com, all the way to host100.example.com.

- `bsub -m "host[01-03].example.com"`
  The job is submitted to host01.example.com, host02.example.com, and host03.example.com.

- `bsub -m "host[5:200].example.com"`
  The job is submitted to host5.example.com, host6.example.com, host7.example.com, all the way to host200.example.com.

- `bsub -m "host[05:09].example.com"`
  The job is submitted to host05.example.com, host06.example.com, all the way to host09.example.com.

- `bsub -m "host[1-10,12,20-25].example.com"`
  The job is submitted to host1.example.com, host2.example.com, host3.example.com, up to and including host10.example.com. It is also submitted to host12.example.com and the hosts between and including host20.example.com and host25.example.com.

- `bsub -m "host[1:10,20,30:39].example.com"`
  The job is submitted to host1.example.com, host2.example.com, host3.example.com, up to and including host10.example.com. It is also submitted to host20.example.com and the hosts between and including host30.example.com and host39.example.com.

- `bsub -m "host[10-20,30,40:50].example.com"`
  The job is submitted to host10.example.com, host11.example.com, host12.example.com, up to and including host20.example.com. It is also submitted to host30.example.com and the hosts between and including host40.example.com and host50.example.com.

- `bsub -m "host[01-03,05,07:09].example.com"`
  The job is submitted to host01.example.com, up to and including host03.example.com. It is also submitted to host05.example.com, and the hosts between and includinghost07.example.com and host09.example.com.

- `bsub -m "hostA[1-2]B[1-3,5].example.com"`
  The job is submitted to hostA1B1.example.com, hostA1B2.example.com, hostA1B3.example.com, hostA1B5.example.com, hostA2B1.example.com, hostA2B2.example.com, hostA2B3.example.com, and hostA2B5.example.com.

Register LSF host names and IP addresses to LSF servers

You can now register the IP and host name of your local LSF host with LSF servers so that LSF does not need to use the DNS server to resolve your local host. This addresses previous issues of resolving the host name and IP address of LSF hosts with non-static IP addresses in environments where the DNS server is not able to properly resolve these hosts after their IP addresses change.

To enable host registration, specify `LSF_REG_FLOAT_HOSTS=Y` in the lsf.conf file on each LSF server, or on one LSF server if all servers have access to the LSB_SHAREDIR directory. This parameter enables LSF daemons to look for records in the reghostscache file when it attempts to look up host names or IP addresses.

By default, the reghostscache file is stored in the file path as defined by the LSB_SHAREDIR parameter in the lsf.conf file. Define the LSB_SHAREDIR parameter so that the reghostscache file can be shared with as many

LSF servers as possible. For all LSF servers that have access to the shared directory defined by the LSB_SHAREDIR parameter, only one of these servers needs to receive the registration request from the local host. The reghostscache file reduces network load by reducing the number of servers to which the registration request must be sent. If all hosts in the cluster can access the shared directory, the registration needs to be sent only to the master LIM. The master LIM records the host information in the shared reghostscache file that all other servers can access. If the LSB_SHAREDIR parameter is not defined, the reghostscache file is placed in the LSF_TOP directory.

The following example is a typical record in the reghostscache file:

```
MyHost1    192.168.1.2    S-1-5-21-5615612300-9789239785-9879786971
```

Windows hosts that register have their computer SID included as part of the record. If a registration request is received from an already registered host, but its SID does not match with the corresponding record's SID in the reghostscache file. This new registration request is rejected, which prevents malicious hosts from imitating another host's name and registering itself as another host.

After you enable host registration, you can register LSF hosts by running the **lsreghost** command from the local host. Specify a path to the hostregsetup file:

- On UNIX, lsreghost -s file_path/hostregsetup
  You must run the UNIX command with root privileges. If you want to register the local host at regular intervals, set up a cron job to run this command.
- On Windows, lsreghost -i file_path\hostregsetup
  The Windows command installs **lsreghost** as a Windows service that automatically starts up when the host starts up.

The hostregsetup file is a text file with the names of the LSF servers to which the local host must register itself. Each line in the file contains the host name of one LSF server. Empty lines and **#comment** text are ignored.

The bmgroup command displays leased-in hosts in the resource leasing model for IBM® Spectrum LSF multicluster capability

The **bmgroup** command displays compute units, host groups, host names, and administrators for each group or unit. For the resource leasing model, host groups with leased-in hosts are displayed by default as `allremote` in the `HOSTS` column.

You can now expand the `allremote` keyword to display a list of the leased-in hosts in the host group with the **bmgroup**.

By default, the `HOSTS` column now displays a list of leased-in hosts in the form *host_name@cluster_name*.

For example, if `cluster_1` defined a host group that is called `master_hosts` that contains only `host_A`, and a host group that is called `remote_hosts` with leased-in hosts as members, and `cluster_2` contains `host_B` and `host_C` that are both being leased in by `cluster_1`:

By default, the `HOSTS` column displays a list of leased-in hosts:

```
GROUP_NAME    HOSTS
master_hosts host_A
remote_hosts host_B@cluster_2 host_C@cluster_2
```

If the LSB_BMGROUP_ALLREMOTE_EXPAND=N parameter is configured in the `lsf.conf` file or as an environment variable, leased-in hosts are represented by a single keyword `allremote` instead of being displayed as a list.

```
GROUP_NAME    HOSTS
master_hosts host_A
remote_hosts allremote
```

RUR job accounting replaces CSA for LSF on Cray

In the LSF integration with Cray Linux, Comprehensive System Accounting (CSA) is now deprecated and replaced with Resource Utility Reporting (RUR).

To modify the default RUR settings, edit the following parameters in the lsf.conf file:

LSF_CRAY_RUR_ACCOUNTING
: Specify `N` to disable RUR job accounting if RUR is not enabled in your Cray environment, or to increase performance. Default value is `Y` (enabled).

LSF_CRAY_RUR_DIR
: Location of the RUR data files, which is a shared file system that is accessible from any potential first execution host. Default value is LSF_SHARED_DIR/*<cluster_name>*/craylinux/*<cray_machine_name>*/rur.

LSF_CRAY_RUR_PROLOG_PATH
: File path to the RUR prolog script file. Default value is /opt/cray/rur/default/bin/rur_prologue.py.

LSF_CRAY_RUR_EPILOG_PATH
: File path to the RUR epilog script file. Default value is /opt/cray/rur/default/bin/rur_epilogue.py.

RUR does not support host-based resource usage (`LSF_HPC_EXTENSIONS="HOST_RUSAGE"`).

The LSF administrator must enable RUR plug-ins, including output plug-ins, to ensure that the LSF_CRAY_RUR_DIR directory contains per-job accounting files (rur.*<job_id>*) or a flat file (rur.output).

Other changes to LSF behavior

See details about changes to default LSF behavior.

General LSF behavior

You cannot use the **bconf** command to define project limits when the cluster has no project limits set.

You cannot delete an advance reservation while jobs are still running in it.

If host preference is specified, compute unit preference is also determined by host preference. Before LSF 10.1, compute unit preference is determined only by the `cu` preference string (`pref=config | maxavail | minavail`).

The JOB_SCHEDULING_INTERVAL parameter in the lsb.params file now specifies the minimal interval between subsequent job scheduling sessions. Specify in seconds, or include the keyword `ms` to specify in milliseconds. If set to 0, subsequent sessions have no minimum interval between them. Previously, this parameter specified the amount of time that **mbschd** sleeps before it starts the next scheduling session.

The job information cache is enabled by default (the JOB_INFO_MEMORY_CACHE_SIZE parameter in the lsb.params file), and the default size of the lsb.jobinfo.events file is 1024 MB (1 GB). New job information is now stored in the new event file instead of individual job files.

The parameter JOB_SWITCH2_EVENT in the lsb.params file is obsolete in LSF 10.1 and later. To take advantage of enhancements to job array performance, set the JOB_ARRAY_EVENTS_COMBINE=Y parameter.

New event replay mechanism writes files to LSF_TMPDIR

On execution hosts, the **sbatchd** daemons write their events to a file under LSF_TMPDIR (the default directory is /tmp). If the LSF temporary directory becomes full, **sbatchd** cannot write to its event file, and the daemons do not recover normally. You must make sure to maintain enough free space in the LSF_TMPDIR directory.

System requirements and compatibility

The following sections describe the system requirements and compatibility information for version 10.1 of IBM® Spectrum LSF.

- **Operating system support**
  LSF supports various operating systems.
- **Management host selection**
  To achieve the highest degree of performance and scalability, use a powerful management host.
- **Server host compatibility**
  LSF 10.1 or later servers are compatible with IBM Spectrum LSF 10.1 management hosts. All LSF 10.1 or later features are supported by IBM Spectrum LSF 10.1 management hosts.
- **LSF add-on compatibility**
  IBM Spectrum LSF 10.1 is compatible with LSF family add-ons.
- **API compatibility**
  To take full advantage of new IBM Spectrum LSF 10.1 features, recompile your existing LSF applications with IBM Spectrum LSF 10.1.

Operating system support

LSF supports various operating systems.

Table 1. Supported operating systems

Operating system	Version	CPU or hardware	Package name	Support
Linux x64 kernel	RHEL 7.8 and 7.9, kernel 3.10, glibc 2.17	x86	• lsf10.1_linux2.6-glibc2.3-x86_64.tar.Z • lsf10.1_lnx310-lib217-x86_64.tar.Z (for kernels above 3.10)	Certified
	RHEL 8.4, 8.5, 8.6, 8.7, and 8.8, kernel 4.18, glibc 2.28			Certified
	RHEL 9.0, 9.1, and 9.2, kernel 5.14, glibc 2.34			Certified

Operating system	Version	CPU or hardware	Package name	Support
	SLES 12.*x*, kernel 3.11, glibc 2.18			Certified
	SLES 15, kernel 4.16, glibc 2.26			Certified
	SLES 15.2, kernel 5.3.18, glibc 2.26			Certified
	CentOS 7.9, kernel 3.10, glibc 2.17			Certified
	CentOS stream 8, kernel 4.18, glibc 2.28			Certified
	CentOS stream 9, kernel 5.14, glibc 2.34			Supported
	Ubuntu 18.04 LTS, kernel 4.15, glibc 2.27			Certified
	Ubuntu 20.04 LTS, kernel 5.4, glibc 2.31			Certified
	Ubuntu 22.04 LTS, kernel 5.15, glibc 2.31			Certified
Generic Linux: Debian, Ubuntu, and OEL	Kernel 3.0, glibc 2.3 or later.			Supported
Linux kernel on IBM Power LE (Little Endian)	RHEL 7.8 and 7.9, kernel 3.10, glibc 2.17	IBM Power 8 LE, IBM Power 9 LE, and IBM Power 10 LE	lsf10.1_lnx310-lib217-ppc64le.tar.Z	Certified
	RHEL 8.4, 8.5, 8.6, 8.7, and 8.8, kernel 4.18, glibc 2.28			Certified
	RHEL 9.0, 9.1, and 9.2, kernel 5.14, glibc 2.34			Supported
	SLES 12.*x*, kernel 3.11, glibc 2.18			Certified
	CentOS stream 8, kernel 4.18, glibc 2.28			Certified
	Ubuntu 18.04 LTS, kernel 4.15, glibc 2.27			Certified
	Ubuntu 20.04 LTS, kernel 5.4, glibc 2.31			Certified
	Ubuntu 22.04 LTS, kernel 5.15, glibc 2.35			Certified

Operating system	Version	CPU or hardware	Package name	Support
Linux ARM	ARMv7	x1_358	lsf10.1_lnx36-lib215-armv7.tar.Z	Supported
	ARMv8	AArch64	lsf10.1_linux3.12-glibc2.17-armv8.tar.Z	Supported
Apple OS X (compute host only)	10.8	x86-64	lsf10.1_macosx.tar.Z	Supported
	10.9			Supported
	10.15	x86-64	lsf10.1_macos-x86_64.tar.Z	Supported
	13.4	ARM64	lsf10.1_macos-arm64.tar.Z	Certified
Microsoft Windows	Windows Server 2012 x64	x86-64	lsf10.1_win-x64.msi	Supported
	Windows Server 20 R2 x64			Certified
	Windows 10 x64			Supported
	Windows Server 2016 TP4 x64			Certified
	Windows Server 2019 x64			Certified
	Windows 11 x64			Supported
	Windows Server 2022 x64			Supported
IBM AIX	7.*x*	IBM Power 8, IBM Power 9, and IBM Power 10	lsf10.1_aix-64.tar.Z	Supported
SUN Solaris on SPARC	10	SPARC 64bit (T2)SPARC 64bit (T2)	lsf10.1_sparc-sol10-64.tar.Z	Supported
	11			Supported
SUN Solaris on x86-64	10	x86-64	lsf10.1_x86-64-sol10.tar.Z	Supported
	11			Supported
HP-UX	11.31 (IA64)	HP Integrity Servers (Itanium2)	lsf10.1_hpuxia64.tar.Z	Supported

End of life (EOL) notice

When a specific release or version of an operating system reaches end of life (EOL) or its end of support date, the operating system vendor no longer supports and releases updates for the product, including automatic fixes, updates, and online technical assistance.

LSF, in turn, is not tested on EOL operating systems. If you have extended support to continue to use EOL operating systems, you can use these operating system versions with LSF. Any issue with using LSF will be supported only if it can be reproduced on a supported operating system.

Management host selection

To achieve the highest degree of performance and scalability, use a powerful management host.

LSF has no minimum CPU requirement. For the platforms on which LSF is supported, any host with sufficient physical memory can run LSF as management host. Swap space is normally configured as twice the physical memory. LSF daemons in a cluster on Linux x86-64 use about 488 MB of memory when no jobs are running. Active jobs use most of the memory that LSF requires.

Note: If a Windows host must be installed as the management host, only supported Windows Server operating systems are recommended as LSF management hosts.

Cluster size	Active jobs	Minimum required memory (typical)	Recommended server CPU (Intel, AMD, OpenPower, or equivalent)
Small (<100 hosts)	1,000	1 GB (32 GB)	Any server CPU
	10,000	2 GB (32 GB)	Recent server CPU
Medium (100 - 1000 hosts)	10,000	4 GB (64 GB)	Multi-core CPU (2 cores)
	50,000	8 GB (64 GB)	Multi-core CPU (4 cores)
Large (>1000 hosts)	50,000	16 GB (128 GB)	Multi-core CPU (4 cores)
	500,000	32 GB (256 GB)	Multi-core CPU (8 cores)

Server host compatibility

LSF 10.1 or later servers are compatible with IBM® Spectrum LSF 10.1 management hosts. All LSF 10.1 or later features are supported by IBM Spectrum LSF 10.1 management hosts.

Important: To take full advantage of all new features that are introduced in the latest release of IBM Spectrum LSF, you *must* upgrade all hosts in your cluster.

LSF add-on compatibility

IBM® Spectrum LSF 10.1 is compatible with LSF family add-ons.

IBM Spectrum LSF RTM and IBM Platform RTM

You can use IBM Platform RTM 8.3 or later to collect data from IBM Spectrum LSF 10.1 clusters. When you add the cluster, select Poller for LSF 8 or Poller for LSF 9.1.

IBM Spectrum LSF License Scheduler and IBM Platform LSF License Scheduler

IBM Platform LSF License Scheduler 8.3 or later are compatible with IBM Spectrum LSF 10.1.

IBM Spectrum LSF Process Manager and IBM Platform Process Manager

IBM Platform Process Manager 9.1 and later, and IBM Spectrum LSF Process Manager is compatible with IBM Spectrum LSF 10.1.

IBM Spectrum LSF Analytics and IBM Platform Analytics

If you use earlier versions of IBM Platform Analytics, do not enable the JOB_ARRAY_EVENTS_COMBINE parameter in the lsb.params file. The parameter introduces an event format that is not compatible with earlier versions of IBM Platform Analytics.

IBM Platform Analytics 9.1.2.2 is compatible with IBM Spectrum LSF 10.1.

IBM Spectrum LSF Application Center and IBM Platform Application Center

If you upgrade earlier versions of IBM Spectrum LSF to version 10.1.0, but you do not upgrade IBM Platform Application Center in an existing LSF cluster, IBM Platform Application Center 9.1.3 and later versions are compatible with IBM Spectrum LSF 10.1.0.

Install a new LSF 10.1 cluster before you install IBM Spectrum LSF Application Center 10.1 to avoid compatibility issues.

API compatibility

To take full advantage of new IBM® Spectrum LSF 10.1 features, recompile your existing LSF applications with IBM Spectrum LSF 10.1.

You must rebuild your applications if they use APIs that changed in IBM Spectrum LSF 10.1.

New and changed LSF APIs

The following APIs or data structures are changed or are new for LSF 10.1:

- struct _limitInfoEnt
- struct _limitInfoReq
- struct _lsb_reasonConf
- struct _lsb_reasonMsgConf
- struct _lsb_rsrcConf
- struct _reasonRefEntry
- struct allLevelReasonMsg
- struct appInfoEnt
- struct estimationResults
- struct globalFairshareLoadEnt
- struct globalShareAcctEnt
- struct gpuRusage

- struct hostInfo
- struct hRusage
- struct jobArrayID
- struct jobArrayIndex
- struct jobCleanLog
- struct jobFinishLog
- struct jobFinish2Log
- struct jobForwardLog
- struct jobInfoEnt
- struct jobInfoHead
- struct jobInfoReq
- struct jobModLog
- struct jobMoveLog
- struct jobPendingSummary
- struct jobPendingSummaryElem
- struct jobStartLog
- struct jobStatusLog
- struct jobSwitchLog
- struct jobStatus2Log
- struct jRusage
- struct keyValue
- struct KVPair
- struct packSubmitReply
- struct parameterInfo
- struct participantShareLoad
- struct pendingReasonInfo
- struct perfmonLog
- struct queryInfo
- struct queueInfoEnt
- struct queueKVP
- struct reasonMessage
- struct reasonRefString
- struct reasonRefStrTab
- struct rmtJobCtrlRecord2
- struct sbdAsyncJobStatusReplyLog
- struct sbdAsyncJobStatusReqLog
- struct sbdJobStartAcceptLog
- struct sbdJobStatusLog
- struct shareLoadInfo
- struct signalLog
- struct slotInfoRequest
- struct statusInfo
- struct submit
- union eventLog
- API ls_eligible()
- API extsched()

New LSF Data Manager APIs

The following APIs are new for LSF Data Manager 10.1:

- das_init

- das_perror
- das_reset
- das_strerror
- dm_admin
- dm_cache
- dm_chgrp
- dm_chmod
- dm_clusters
- dm_delete_tag
- dm_list_tags
- dm_local
- dm_params
- dm_stage_in
- dm_stage_out
- getRegisteredDmdCluster
- freeDmParams

IBM Spectrum LSF product packages

The IBM Spectrum LSF product consists of distribution packages for supported operating systems, installation packages, and entitlement files.

Supported operating systems

For detailed LSF operating system support information, refer to Operating system support.

UNIX and Linux installer packages

The same installer packages are used for all LSF editions on UNIX and Linux.

lsf10.1.0.*fix_pack_level*_lsfinstall.tar.Z
: The standard installer package. Use this package in a heterogeneous cluster with a mix of systems other than x86-64. Requires approximately 1 GB free space.

lsf10.1.0.*fix_pack_level*_lsfinstall_linux_x86_64.tar.Z
: Use this smaller installer package in a homogeneous x86-64 cluster. If you add other non-x86-64 hosts, you must use the standard installer package. Requires approximately 100 MB free space.

lsf10.1.0.*fix_pack_level*_no_jre_lsfinstall.tar.Z
: For all platforms not requiring the JRE. JRE version 1.4 or higher must already be installed on the system. Requires approximately 1 MB free space.

lsf10.1.0.*fix_pack_level*_lsfinstall_linux_ppc64le.tar.Z
: Installer package for Linux on IBM Power 6, 7, and 8 Little-Endian (LE) systems

Entitlement files

The following LSF entitlement configuration files are available:

LSF Standard Edition
: lsf_std_entitlement.dat

LSF Express Edition

lsf_exp_entitlement.dat
LSF Advanced Edition
lsf_adv_entitlement.dat

Bugs fixed

LSF 10.1 releases and Fix Packs contain bugs that were fixed since the general availability of LSF.

Note: Starting in Fix Pack 14, instead of reading this bugs fixed topic, refer to the fix pack README file on IBM Fix Central. Each README file lists the contents of the fix pack, including any fixes for issues since the last fix pack, and any new solutions or features included with the fix pack.

Fix Pack 13

LSF 10.1 Fix Pack 13 contains fixes to all bugs that were resolved between 11 June 2021 and 15 April 2022.

The list of fixed bugs is available in the README for LSF 10.1 Fix Pack 13, available at IBM Fix Central.

Fix Pack 12

LSF 10.1 Fix Pack 12 contains fixes to all bugs that were resolved before 10 June 2021.

The list of fixed bugs is available in the README for LSF 10.1 Fix Pack 12: https://ibm.biz/lsf10-1_fp12.

Fix Pack 11

LSF 10.1 Fix Pack 11 contains fixes to all bugs that were resolved before 12 November 2020.

The list of fixed bugs is available in the README for LSF 10.1 Fix Pack 11: https://ibm.biz/lsf10-1_fp11.

Fix Pack 10

LSF 10.1 Fix Pack 10 contains fixes to all bugs that were resolved before 10 April 2020.

The list of fixed bugs is available in the README for LSF 10.1 Fix Pack 10: https://ibm.biz/lsf10-1_fp10.

Fix Pack 9

LSF 10.1 Fix Pack 9 contains fixes to all bugs that were resolved before 16 October 2019.

The list of fixed bugs is available in the README for LSF 10.1 Fix Pack 9: https://ibm.biz/lsf10-1_fp9.

Fix Pack 8

LSF 10.1 Fix Pack 8 contains fixes to all bugs that were resolved before 10 May 2019.

The list of fixed bugs is available in the README for LSF 10.1 Fix Pack 8: https://ibm.biz/lsf10-1_fp8.

Fix Pack 7

LSF 10.1 Fix Pack 7 contains fixes to all bugs that were resolved before 18 December 2018.

The list of fixed bugs is available in the README for LSF 10.1 Fix Pack 7: https://ibm.biz/lsf10-1_fp7.

Fix Pack 6

LSF 10.1 Fix Pack 6 contains fixes to all bugs that were resolved before 24 May 2018.

The list of fixed bugs is available in the README for LSF 10.1 Fix Pack 6: https://ibm.biz/lsf10-1_fp6.

Fix Pack 5

LSF 10.1 Fix Pack 5, which only applies to IBM POWER 9 platforms, contains fixes to all bugs that were resolved before 27 March 2018.

The list of fixed bugs is available in the README for LSF 10.1 Fix Pack 5: https://ibm.biz/lsf10-1_fp5.

Fix Pack 4

LSF 10.1 Fix Pack 4 contains fixes to all bugs that were resolved before 20 November 2017.

The list of fixed bugs is available in the README for LSF 10.1 Fix Pack 4: https://ibm.biz/lsf10-1_fp4.

Fix Pack 3

LSF 10.1 Fix Pack 3 contains fixes to all bugs that were resolved before 6 July 2017.

The list of fixed bugs is available in the README for LSF 10.1 Fix Pack 3: https://ibm.biz/lsf10-1_fp3.

Fix Pack 2

LSF 10.1 Fix Pack 2 contains fixes to all bugs that were resolved before 15 February 2017.

The list of fixed bugs is available in the README for LSF 10.1 Fix Pack 2: https://ibm.biz/lsf10-1_fp2.

Fix Pack 1

LSF 10.1 Fix Pack 1 contains fixes to all bugs that were resolved before 20 October 2016.

The list of fixed bugs is available in the README for LSF 10.1 Fix Pack 1: https://ibm.biz/lsf10-1_fp1.

June 2016 release

The June 2016 release of LSF 10.1 contains fixes to all bugs that were resolved before 29 April 2016.

Known issues

LSF 10.1 has the following known issues.

- The Nvidia Data Center GPU Manager (DCGM) integration is enabled by defining the LSF_DCGM_PORT parameter in the lsf.conf file. Note, however, that Nvidia Multi-Instance GPU (MIG) integration with DCGM does not work with LSF and will be addressed in a future LSF fix.
- When you set global limits for job resource allocation, the global limit is only set for the resource on the local cluster; it does not does not take effect on the resource globally. For instance, if you check global resource usage, by running the **bgpinfo resource** command, it does not show that the resource has been released after running the job (that is, the reserved resources number does not reduce). However, if you check the local resource usage, by running the **bhosts -s** command on the local cluster, this shows released resources for that cluster. The local cluster, is however, unaware of any global limits set on the global resources.
- If the LSF_STRICT_CHECKING parameter is not defined in the $LSF_ENVDIR/lsf.conf file, there are known issues for the existing running or suspended **blaunch** jobs after the cluster is updated to LSF 10.1 Fix Pack 12:
  - There are XDR error messages in the **sbatchd** and RES log files.
  - LSF cannot update the runtime resource usage information of existing running **blaunch** jobs.
  - LSF cannot stop existing running **blaunch** jobs.
  - LSF cannot resume existing suspended **blaunch** jobs.

  To avoid these problems, ensure that there are no running or suspended **blaunch** jobs before you update your cluster to LSF 10.1 Fix Pack 12.

- External authentication (**eauth**) fails when it cannot resolve the true operating system uid or gid of the calling process. These situations might occur within nested namespaces.
- When specifying time zones in automatic time-based configurations, if you specify the same abbreviation for multiple time zones, LSF might not select the correct time zone. A patch will be made available shortly after the release of Fix Pack 9 to address this issue.
- The DCGM (NVIDIA Data Center GPU Manager) integration does not work as expected due to a missing libdcgm.so file. To resolve this issue, create a softlink to ensure that the libdcgm.so file exists and is accessible:

  ```
  sudo ln -s /usr/lib64/libdcgm.so.1 /usr/lib64/libdcgm.so
  ```

- On RHEL 8, the LSF cluster cannot start up due to a missing libnsl.so.1 file. To resolve this issue, install the libnsl package to ensure that the libnsl.so.1 exists.

  ```
  yum install libnsl
  ```

- On AIX, a TCL parser issue causes jobs to pend when the LSF_STRICT_RESREQ=N parameter is set in the lsf.conf file, even though AIX hosts are available. To avoid the problem, make sure that LSF_STRICT_RESREQ=Y.
- While running a job, a RHEL 7.2 server host may fail with the following error messages in the system log file or the system console:

  ```
  INFO: rcu_sched self-detected stall on CPU { number}
  INFO: rcu_sched detected stalls on CPUs/tasks:
  BUG: soft lockup - CPU#number stuck for time! [res:16462]
  ```

  This is an issue with RHEL 7.2 kernel-3.10.0-327.el7. To resolve this issue, download and apply a RHEL kernel security update. For more details, refer to https://rhn.redhat.com/errata/RHSA-2016-2098.html.

Limitations

LSF 10.1 has the following limitations.

Difference in decay behavior for local resources and global resources

If there a limit set on a resource, the resource usage does not decay (decrease of reserved resource amounts). For a global resource, if there is global limit set on it, the resource usage will not decay when you query by running the **bgpinfo resource -s** command; however, you can still can see that the resource usage decayed from the local cluster, by running the **bhosts -s** command.

For global limits set on a global resource, check the decayed value for the global resource from the global cluster

You can define global limit resource allocation so and that global resources decay (decrease reserved resource amounts) when running submitted jobs.

If you set a global limit on a global resource, and check the global resource usage by running the **bgpinfo resource -s** command, this reflects the decayed value of the resource. If you check the local resource usage by running the **bhosts -s** command, this does not show the decayed value, because the local cluster is not aware of global limits set on global resources.

Number of requested slots for pending jobs in the bjobs command

In jobs with affinity resource requirements, LSF generally calculates the number of requested slots (in the nreq_slot field) only when the pu_type value is consistent with the value of the EGO_DEFINE_NCPUS parameter, but there are certain conditions where this is not possible. The following are conditions under which LSF cannot calculate the nreq_slot value:

- The pu_type is `numa`.
- `EGO_DEFINE_NCPUS=procs` and the affinity string contains `exclusive=(numa,alljobs)`
- `EGO_DEFINE_NCPUS=cores` and the affinity string contains `exclusive=(numa,alljobs)` or `exclusive=(socket,alljobs)`
- `EGO_DEFINE_NCPUS=threads` and the affinity string contains `exclusive=(numa,alljobs)`, `exclusive=(socket,alljobs)`, or `exclusive=(core,alljobs)`

For jobs with alternative resource requirements, the nreq_slot field shows the number of requested slots according to the first alternative resource requirement.

For parallel jobs that are submitted with a specified minimum and maximum number of tasks (`bsub -n` *`min`*`,`*`max`*), the nreq_slot field shows the minimum number of requested slots, which is calculated from the minimum number of requested tasks from the `bsub -n` *`min`* command.

For exclusive jobs or compute unit exclusive jobs, LSF cannot calculate the nreq_slot value.

GPU jobs preempt more jobs than expected

If `LSB_GPU_NEW_SYNTAX=extend` is set in the lsf.conf file and GPU preemption is enabled, higher priority GPU jobs might preempt more lower priority GPU jobs than expected.

CPU affinity information for cloud instances is lost when mbatchd restarts

LSF caches CPU affinity information from cloud instances into the **mbatchd** daemon memory. Since this information is not persisted, if LSF restarts the **mbatchd** daemon, all cached information is lost, including information on CPU affinity information for cloud instances.

If you submit new jobs with affinity resource requirements, LSF has to cache the CPU affinity information again.

Cloud instance provisioning for jobs with affinity resource requirements

The **mbatchd** daemon only collects CPU affinity information when LSF resource connector provisions the cloud instance. If the job affinity resource requirements are not satisfied by the provisioned cloud instances, the job pends. LSF only provisions the cloud instance one time and does not try to provision the cloud instance again.

To work around this issue, use the **bmod** command to change your affinity resource requirements. LSF attempts to provision another cloud instance to meet the update job affinity requirement.

Job start time prediction

Job start time prediction has limited support for guaranteed SLA. The estimator cannot schedule the jobs that borrow the resources in the guarantee pool. The estimator scheduler bypasses backfilling scheduling, which calls the guarantee reserve plug-in to schedule loan jobs.

GPU MPS solution

The MPS Server supports up to 16 client CUDA contexts concurrently. And this limitation is per user per job. That means MPS can handle at most 16 CUDA processes at one time even though LSF allocated multiple GPUs.

Registering dynamic LSF host IP address or name into the management host LIM

In shared LSF environments that frequently change IP addresses, client hosts need to register with the management host only. If client hosts do not register, the cache file is overwritten by other LIM hosts and the cache file becomes inaccurate. Windows client hosts with the same IP address and a new SID, administrators must manually remove old records from cache file and restart the management host LIM to reregister.

Simplified affinity requirement syntax

The **esub.p8aff** script cannot modify the environment variables when called by the **bmod** command. The SMT argument (the OMP_NUM_THREADS environment variable) cannot be applied to the execution hosts, but the cpus_per_core and distribution_policy arguments can be modified. Therefore, when calling the **esub.p8aff** script from the **bmod** command, you must ensure that the specified SMT argument is the same as the SMT

argument in the original job submission. Otherwise, the generated affinity string might not match the effective SMT mode on execution hosts, which might produce unpredictable affinity results.

Detailed changes

The following topics describe new and changed commands, options, output, configuration parameters, environment variables, accounting, and job event fields in LSF 10.1.

Note: Starting in Fix Pack 14, instead of reading this topic, refer to the What's new information for each fix pack (which contain changed and new details, including direct references to affected topics).

- **New and changed commands, options, and output**
  The following command options and output are new or changed for LSF 10.1.
- **New and changed configuration parameters and environment variables**
  The following configuration parameters and environment variables are new or changed for LSF 10.1.
- **New and changed accounting and job event fields**
  The following job event fields are added or changed for LSF 10.1.

New and changed commands, options, and output

The following command options and output are new or changed for LSF 10.1.

Note: Starting in Fix Pack 14, instead of reading this topic, refer to the What's new information for each fix pack (which contain changed and new details, including direct references to affected topics).

Commands that support host names as a parameter also accept host groups

Starting in LSF Version 10.1 Fix Pack 13, existing LSF commands that support host names as a parameter option now also accept host groups. For details on the affected commands, see **battr**, **bresume**, **brvs**, **lshosts**, and **lsload**. Additionally, to use the host group parameter with the **lshosts** and **lsload** commands, you must first configure the LSF_HOSTGROUP_INFO=Y setting in the lsf.conf file.

Commands that use condensed host names

All commands where you can specify condensed notation, including commands that use the -m option or a host list to specify multiple host names, now allow colons (`:`) to specify a range of numbers. Colons are used the same as hyphens (–) are currently used to specify ranges and can be used interchangeably in condensed notation. You can also use leading zeros to specify host names.

You can also use multiple square brackets (with the supported special characters) to define multiple sets of non-negative integers anywhere in the host name. For example, `hostA[1,3]B[1-3]` includes `hostA1B1`, `hostA1B2`, `hostA1B3`, `hostA3B1`, `hostA3B2`, and `hostA3B3`.

Commands that specify resource requirement strings

All commands where you specify resource requirement strings by using the -R no longer have any size limitations on the resource requirement string. Previously, the string was limited to 512 bytes.

bacct

- **bacct**: The default output includes the expansion factor of a job, which is its turnaround time divided by its run time. The **bacct** command displays the average, maximum, and minimum expansion factors.
- **bacct -E**: If the TRACK_ELIGIBLE_PENDINFO parameter in the lsb.params file is set to `Y` or `y`, the output format is the same as the **bacct** command with no options, except that it uses eligible pending time to calculate the wait time, turnaround time, and expansion factor (turnaround time/run time), instead of total pending time.
  If TRACK_ELIGIBLE_PENDINFO is disabled and LSF did not log any eligible or ineligible pending time, the ineligible pending time is counted as 0 and all pending time is regarded as eligible.

- **bacct -l** now displays energy consumption information if `LSF_COLLECT_ENERGY_USAGE=Y` is enabled in lsf.conf and you are running LSF on Cray.
- In LSF Version 10.1 Fix Pack 4, **bacct -f** now allows you to specify – as the argument (`bacct -f -`) to force the **bacct** command to use the lsb.acct log file for accounting statistics. If you are using LSF Explorer to retrieve accounting log records, the -f - option (or any -f argument that specifies a log file) forces the **bacct** command to bypass LSF Explorer and uses the log file instead.
- In LSF Version 10.1 Fix Pack 4, the new **bacct -gpu** option displays information on GPU resource usage for the job. Use only with the -l option.
- In LSF Version 10.1 Fix Pack 6, the **bacct -l** option now displays the total approximate number of read/write bytes of all storage pools on IBM® Spectrum Scale if IBM Spectrum Scale I/O accounting with IBM Spectrum LSF Explorer is enabled by setting `LSF_QUERY_ES_FUNCTIONS="gpfsio"` or `"all"` in the lsf.conf file.
- In LSF Version 10.1 Fix Pack 11, the **bacct** command now displays the scheduler efficiency for all finished jobs in a cluster.
- In LSF Version 10.1 Fix Pack 11, the new **bacct -rcacct** command option displays accounting statistics for jobs that are associated with the specified LSF resource connector account name.
- In LSF Version 10.1 Fix Pack 11, the new **bacct -rcalloc** command option displays accounting statistics for jobs that are associated with the specified LSF resource connector account name and actually ran on an LSF resource connector host.
- In LSF Version 10.1 Fix Pack 12, the **bacct -cname** command option is deprecated and will be removed in a future version.

badmin

- **gpdckconfig**: This new subcommand checks the global policy configuration file lsb.globalpolicies located in the LSB_CONFDIR/cluster_name/configdir directory.
- **gpddebug**: This new subcommand sets the message log level for the global fair share policy daemon (**gpolicyd**) to include additional information in log files. You must be **`root`** or the LSF administrator to use this command.
- **gpdrestart**: This new subcommand dynamically reconfigures LSF global policies and restarts **gpolicyd**.
- **gpdtime**: This new subcommand sets the timing level for **gpolicyd** to include more timing information in log files. You must be **`root`** or the LSF administrator to use this command.
- The **showconf** subcommand has a new option gpd to display all configured and their values set in lsf.conf or ego.conf that affect **gpolicyd**.
- The **mbddebug -c** option now has a new type of log class (LC2_G_FAIR) to log global fair share messages.
- The -s option for the **mbddebug** and **schddebug** subcommands is used to temporarily change the size of the **mbbatchd** or **mbschd** logging queue. This value temporarily overrides the value of

LSF_LOG_QUEUE_SIZE in lsf.conf, but this value is ignored if `LSF_LOG_THREAD=N` is defined in lsf.conf.

- **badmin diagnose -c jobreq** saves an active image of the scheduler job buckets into a snapshot file as raw data in XML or JSON format. By default, the name of the snapshot file is jobreq_*<hostname>*_*<dateandtime>*.*<format>*, is in XML format, and is saved to the location specified in the DIAGNOSE_LOGDIR parameter. Use the -f option to specify a custom file name and path and the -t option to specify whether the file is in XML or JSON format.
- **badmin perfmon view** now shows the number of jobs that are reordered, that is, the number of jobs that reused the resource allocation of a finished job.
- In LSF Version 10.1 Fix Pack 2, **badmin mbdrestart** now restarts the **mbatchd** daemon in parallel by default (that is, **badmin mbdrestart -p** is now the default behavior). To force the **mbatchd** daemon to restart in serial (which was the previous default behavior), run **badmin mbdrestart -s**
- In LSF Version 10.1 Fix Pack 7, a -json option has been added to the **badmin perfmon view** command to support the graphing of performance metrics data in other LSF family applications (for example, IBM Spectrum LSF Application Center).
- **rc view**: In LSF Version 10.1 Fix Pack 8, this new subcommand shows LSF resource connector information from the specified host providers.
- **rc error**: In LSF Version 10.1 Fix Pack 8, this new subcommand shows LSF resource connector error messages from the specified host providers. To get the error messages, the third-party **mosquitto** message queue application must be running on the host.
- **hclose -i**: In LSF Version 10.1 Fix Pack 10, this new option attaches a specific lock ID to the closed host. Each lock ID is a string that can contain up to 128 alphanumeric and underscore (_) characters. The keyword all is reserved and cannot be used as the lock ID. A closed host can have multiple lock IDs, and the host remains closed until there are no more lock IDs attached to the host.
- **hopen -i**: In LSF Version 10.1 Fix Pack 10, this new option removes the specified lock ID from the closed host. If there are no more lock IDs attached to the host, this command also opens the host.
- In LSF Version 10.1 Fix Pack 11, the **badmin** command no longer has the setuid bit set at installation. This means that this command no longer runs with root privileges.
- In LSF Version 10.1 Fix Pack 11, the **badmin perfmon view** command now displays the scheduler efficiency for finished jobs within a sampling period and all finished jobs in the cluster.
- In LSF Version 10.1 Fix Pack 11, the following **sbatchd** daemon control commands are obsolete and will be deprecated in a future release:
  - **badmin hrestart**
  - **badmin hshutdown**
  - **badmin hstartup**

  Instead of these obsolete commands, use the `bctrld` *`action`* `sbd` (where *`action`* is `restart`, `start`, or `stop`) to control the **sbatchd** daemon.

- In LSF Version 10.1 Fix Pack 12, the new **badmin security view** command displays the current configuration of the LSF security mechanism.

bapp

- The **bapp -l** command now shows information on pending time limits and eligible pending time limits for jobs in the application profile.

battach (New)

In LSF Version 10.1 Fix Pack 10, the **battach** command runs a shell process (/bin/sh by default) to connect to an existing job execution host or container.

battr (New)

In LSF Version 10.1 Fix Pack 10, the **battr** command manages host attributes for job affinity scheduling.

bconf

- In LSF Version 10.1 Fix Pack 5 and Fix Pack 6 the **bconf -pack** option has been added to allow the **bconf** command to read multiple requests and send them to mbatchd at the same time. **bconf** reads and parses the text file, with each line an individual **bconf** request. However, the requests are grouped together and sent to mbatchd at one time. Also, a **set** action is introduced to force an update or create action on users and user groups.
- In LSF Version 10.1 Fix Pack 8, the `serviceclass` keyword is added to the **addmember**, **rmmember**, **update**, **create**, and **delete** sub-commands. This new keyword allows you to modify service classes in the lsb.serviceclasses file.

bctrld (New)

In LSF Version 10.1 Fix Pack 11, the **bctrld** command is an administrative tool to start, stop, or restart the **LIM**, **RES**, and **sbatchd** daemons. The **bctrld** executable file is installed with the setuid flag turned off and this command can be used only by root.

bgdel

- -d: In LSF 10.1 Fix Pack 13, this new option deletes job groups with idle times that are larger than the specified idle_time.

bgpinfo (New)

Use the **bgpinfo** command with global fair share scheduling. The **bgpinfo** command has subcommands to display information about global fair share, including global fair share policy configurations, aggregated fair share load for global fair share policies, and the status of the global policy daemon (**gpolicyd**) and global fair share policies.

- **resource** subcommand: In LSF 10.1 Fix Pack 13, this new subcommand displays the following information on global resources: resource usage, resource type, and reservation type.
- **policy** subcommand: In LSF 10.1 Fix Pack 13, this new subcommand displays the distribution policies for global resources and global limits.
- **fsload -l** subcommand option: In LSF 10.1 Fix Pack 13, this option now displays the **`STARTED_JOBS`** and **`RESERVED_JOBS`** fields when the fair share_JOB_COUNT parameter is enabled in the lsb.params file.

bhist

- **bhist -l**: If TRACK_ELIGIBLE_PENDINFO in lsb.params is set to `Y` or `y`, this option also shows the total amount of time that the job spends in the eligible and ineligible pending states after the job starts.
- **bhist -l** also shows information on pending time limits and eligible pending time limits for the job.
- In LSF Version 10.1 Fix Pack 3, the **bhist -l** option displays the failure reason of the **cleartool setview** command when it is called by the daemon wrapper with the LSF Integration for Rational ClearCase.
- In LSF Version 10.1 Fix Pack 4,**bhist -f** now allows you to specify – as the argument (`bhist -f -`) to force the **bhist** command to use the lsb.events log file. If you are using IBM Spectrum LSF Explorer (LSF Explorer) to retrieve event log records, the -f - option (or any -f argument that specifies a log file) forces the **bhist** command to bypass LSF Explorer and use the log file instead.

- In LSF Version 10.1 Fix Pack 4, **bhist -n** controls how many job records (blocks of jobs) return from LSF Explorer if you are using LSF Explorer to retrieve event log records. The block size is configured in LSF Explorer.
- In LSF Version 10.1 Fix Pack 4, the new **bhist -gpu** option displays information on GPU resource usage for the job. Use only with the -l option.
- In LSF Version 10.1 Fix Pack 4, the **bhist -l** option displays energy accounting if you are using LSF Explorer to retrieve energy data.
- In LSF Version 10.1 Fix Pack 4, the **bhist -l** option displays the names of any **esub** (or **epsub**) used to submit a job (using the **bsub -a** command). If multiple **esubs** were used, first the default then the user-specified **esubs** are displayed in output. If a user-specified **esub** script is the same as the default **esub** script, the duplicate **esubs** will show as one entry. If a job is submitted with an **esub** containing parameters, the **esub** and its parameters will be shown as well, and the format of the **esub** is the same as that specified in the job submission.
- In LSF Version 10.1 Fix Pack 10, the **bhist -l** option shows the following additional information:
  - Resources that are borrowed from the GSLA pools.
  - Job's attribute requirements and history of modifications to the attribute requirements.
  - User-specified reasons for suspending (stopping) or resuming a job, and the hosts that issued job kill, suspend, or resume requests.

bhosts

- In LSF Version 10.1 Fix Pack 2, the new **bhosts -o** option customizes specific fields that the **bhosts** command displays.
- In LSF Version 10.1 Fix Pack 2, the new **bhosts -json** option displays customized **bhosts -o** command output in JSON format. The -json option must be used together with the -o command option.
- In LSF Version 10.1 Fix Pack 4, the **bhosts -aff** option ignores the NUMA level if the host only has a single NUMA node.
- In LSF Version 10.1 Fix Pack 4, use the **bhosts -rc** and the **bhosts -rconly** commands to see information about resources that are provisioned by LSF resource connector. The -rc and -rconly options make use of the third-party **mosquitto** message queue application to support the additional information displayed by these **bhosts** options. The **mosquitto** binary file is included as part of the LSF distribution. To use the **mosquitto** daemon that is supplied with LSF, you must configure the LSF_MQ_BROKER_HOSTS parameter in the lsf.conf file to enable LIM to start the **mosquitto** daemon and for **ebrokerd** to send resource provider information to the MQTT message broker.
- In LSF Version 10.1 Fix Pack 8, the **bhosts -rc** and **bhosts -rconly** commands now show the instance ID of the provisioned hosts.
- In LSF Version 10.1 Fix Pack 10, the -l option now displays any locked IDs that are attached to the closed host. The new -C option, which is used with -l, displays the lock IDs and comments in tabular format.
- In LSF Version 10.1 Fix Pack 10, the -l option now displays detailed information on any attributes that are attached to the host.
- In LSF Version 10.1 Fix Pack 10, the new -attr option displays information on host attributes for attribute affinity scheduling.
- In LSF Version 10.1 Fix Pack 10, the -o option now has the attribute field to display information on host attributes for attribute affinity scheduling.
- In LSF Version 10.1 Fix Pack 10, the **bhosts -l** option no longer displays GPU-based information if `LSF_GPU_RESOURCE_IGNORE=Y` is defined in the lsf.conf file. Use the **bhosts -gpu** option to view this information.
- In LSF Version 10.1 Fix Pack 10, the **`LOCATION`** field in the **bhosts -s** command option displays **`ALL`** instead of displaying each individual host in the cluster if the LOCATION parameter in the lsf.cluster.*clustername* file is set to `all` to indicate that the resource is shared by all hosts in the cluster. Use the new **bhosts -loc** command option to display each individual host in the cluster.

- In LSF Version 10.1 Fix Pack 11, the **bhosts -gpu** command option now displays the GPU vendor type (AMD or Nvidia).
- In LSF Version 10.1 Fix Pack 11, the **bhosts -gpu -l** command option now shows additional MIG device information.
- In LSF Version 10.1 Fix Pack 11, the **bhosts -o** option has a new mig_alloc keyword to display the MIG allocation information in the **bhosts** customized output.
- In LSF Version 10.1 Fix Pack 12, the **bhosts -cname** command option is deprecated and will be removed in a future version.

bimages (New)

Use the new **bimages** command to display information on Docker container images.

bjobs

- If the TRACK_ELIGIBLE_PENDINFO parameter in the lsb.params file is set to `Y` or `y`:
  - **bjobs -l** also shows the total amount of time that the job is in the eligible and ineligible pending states after the job starts.
  - **bjobs -pei** shows pending jobs in lists of jobs that are eligible for scheduling and ineligible for scheduling.
  - **bjobs -pe** shows only pending jobs that are eligible for scheduling.
  - **bjobs -pi** shows only pending jobs that are ineligible for scheduling.
  - **bjobs -o** now has the `pendstate`, `ependtime`, and `ipendtime` fields that you can specify to display jobs' pending state, eligible pending time, and ineligible pending time.
- The -rusage "resource_name[,resource_name,. . .]" option is added to display only jobs that request the resources that are specified by the filter.
- **bjobs -l** also shows information on job-level and effective (that is, the merged value of queue-, application-, and job-level) pending time limits and eligible pending time limits for the job.
- **bjobs -o** now has the following fields:
  - `effective_plimit`, `plimit_remain`, `effective_eplimit`, and `eplimit_remain` to display the job's pending time limit, remaining pending time, eligible pending time limit, and remaining eligible pending. You can use the -p option to show only information for pending jobs.
  - `pend_reason` shows the pending reason field of a job. If a job has no pending reason (for example, the job is running), then the **`pend_reason`** field of the job is NULL and shows a hyphen (`-`).
- **bjobs -p** now includes the levels 0 - 3, for example, **bjobs -p2**.

  Level 0
  : Displays all pending reasons as in previous releases.

  Level 1
  : Displays only a single key reason.

  Level 2
  : Shows categorized host-based pending reasons for candidate hosts in the cluster. For the candidate hosts, the actual reason on each host is shown. For each pending reason, the number of hosts that give that reason is shown. The actual reason messages appear from most to least common.

  Level 3
  : Shows the categorized host-based pending reasons for both candidate and non-candidate hosts in the cluster. For the hosts considered, gives the actual reason on each host. For the hosts not considered, gives the actual reason on each host. For each pending reason, gives the number of hosts that show that reason. The actual reason messages appear from most to least common.

See also the new parameters LSB_BJOBS_PENDREASON_LEVEL and LSB_SUPPRESS_CUSTOM_REASONS for configuration options.

- **bjobs -psum** (equivalent to **bjobs -p -psum** by default) displays a summarized list of the number of jobs, hosts, and occurrences for each pending reason.
- In LSF Version 10.1 Fix Pack 2, the **bjobs -o** option now has the following fields:
  - `jobindex` shows the job array index.
  - `estimated_run_time` shows estimated run time of the job.
  - `ru_utime` and `ru_stime` show the user time used and the system time used from the resource usage information for the job.
  - `nthreads` shows the number of threads that the job used
  - `hrusage` shows the per-host resource usage information.
  - `plimit` and `eplimit` show the pending time limit and eligible time limit.
  - `licproject` shows the license project information.
  - `srcjobid`, `dstjobid`, and `source_cluster` show the submission cluster job ID, execution cluster job ID, and the name of the submission cluster when using the LSF multicluster capability.
- In LSF Version 10.1 Fix Pack 2, the new **bjobs -json** option displays customized **bjobs -o** command output in JSON format. The -json option must be used together with the -o command option.
- In LSF Version 10.1 Fix Pack 2, the new **bjobs -hms** option displays times from the customized **bjobs -o** command output in **`hh:mm:ss`** format. The -hms option must be used together with the -o or -o -json command options.
- In LSF Version 10.1 Fix Pack 2, the new **bjobs -ignorebadjobid** option removes unmatched jobs from the **bjobs** command output. The **bjobs** command does not display any unmatched jobs, such as jobs that are not found. Jobs with invalid job IDs are still displayed.
- In LSF Version 10.1 Fix Pack 3, the **bjobs -o** option now has the following fields:
  - `rsvid` shows the reservation ID, if the job is associated with an advance reservation.
- In LSF Version 10.1 Fix Pack 3, the new **bjobs -U** option displays jobs that are associated with the specified advance reservation ID.
- In LSF License Scheduler Version 10.1 Fix Pack 3, the pending reason messages shown with the **bjobs -p** command options (including levels 1 - 3) now include the project name for project mode or fast dispatch mode features, and the cluster name for cluster mode features.
- In LSF Version 10.1 Fix Pack 3, the new **bjobs -U** option displays jobs that are associated with the specified advance reservation ID.
- In LSF Version 10.1 Fix Pack 3, the **bjobs -l** option displays the failure reason of the **cleartool setview** command when it is called by the daemon wrapper with the LSF Integration for Rational ClearCase.
- In LSF Version 10.1 Fix Pack 4, the new **bjobs -gpu** option displays information on GPU resource usage for the job. Use only with the -l or -UF options.
- In LSF Version 10.1 Fix Pack 4, the **bjobs -l** and the **bjobs -o energy** options display energy accounting if you are using LSF Explorer to retrieve energy data.
- In LSF Version 10.1 Fix Pack 4, the **bjobs -l** and the **bjobs -o esub** options display the names of any **esub** (or **epsub**) used to submit a job (using the **bsub -a** command). If multiple esubs were used, first the default then the user-specified esubs are displayed in output. If a user-specified **esub** script is the same as the default **esub** script, the duplicate esubs will show as one entry. If a job is submitted with an **esub** containing parameters, the **esub** and its parameters will be shown as well, and the format of the **esub** is the same as that specified in the job submission.
- In LSF Version 10.1 Fix Pack 5 and Fix Pack 6, the new **bjobs -prio** option displays the detailed absolute priority scheduling (APS) factor values for all pending jobs.
- In LSF Version 10.1 Fix Pack 5 and Fix Pack 6, the new **bjobs -plan** option is used as a filter to display jobs with allocation plans. At the same time, **bjobs -l** will show the planned start time for all jobs with an allocation plan and **bjobs -l -plan** will filter all jobs with plans and show the planned allocations and job start times.
- In LSF Version 10.1 Fix Pack 6, the **bjobs -o** option now has the following fields:

    - `gpfsio` shows job usage (I/O) data on IBM Spectrum Scale if IBM Spectrum Scale I/O accounting with IBM Spectrum LSF Explorer is enabled by setting `LSF_QUERY_ES_FUNCTIONS="gpfsio"` or `"all"` in the lsf.conf file.
- In LSF Version 10.1 Fix Pack 5 and Fix Pack 6, the **bjobs -l** option only displays a table of message indices that have an associated message. Previously, **bjobs -l** displayed a table of messages with a line for all messages from 0 to the highest message index, even for indices with no messages.
- In LSF Version 10.1 Fix Pack 6, the **bjobs -l** option now displays approximate accumulated job disk usage (I/O) data on IBM Spectrum Scale if IBM Spectrum Scale I/O accounting with IBM Spectrum LSF Explorer is enabled by setting `LSF_QUERY_ES_FUNCTIONS="gpfsio"` or `"all"` in the lsf.conf file.
- In LSF Version 10.1 Fix Pack 6, the **bjobs -o "hrusage"** option now displays the PIDs of each host.
- In LSF Version 10.1 Fix Pack 7, the **bjobs -o** option now has the following fields:
    - `nreq_slot` shows the calculated number of requested slots for jobs.
    - `gpu_num` shows the number of physical GPUs that the job is using.
    - `gpu_mode` shows the GPU compute mode that the job is using (**`shared`** or **`exclusive_process`**).
    - `gpu_alloc` shows the job-based GPU allocation information.
    - `j_exclusive` shows whether the job requested exclusive allocated GPUs (that is, if the GPUs cannot be shared with other jobs).
    - `kill_reason` shows the user-specified reason for killing the job.
- In LSF Version 10.1 Fix Pack 7, the **bjobs -l -pac** option now displays the exit reason for exited jobs. The exit reason is displayed in the **`EXIT_REASON:`** field.
- In LSF Version 10.1 Fix Pack 8, the **bjobs -plan** option displays the planned start and end times for the job in the **`PLAN_START_TIME`** and **`PLAN_FINISH_TIME`** columns.
- In LSF Version 10.1 Fix Pack 9, the **bjobs -o** option now has the `ask_hosts` field, which shows the list of requested hosts as specified by the **bsub -m** command option.
- In LSF Version 10.1 Fix Pack 10, the **bjobs -l** option shows the following additional information:
    - Resources that are borrowed from the GSLA pools.
    - Job's attribute requirements.
- In LSF Version 10.1 Fix Pack 10, the new **bjobs -env** option displays the environment variables in the job submission environment for the specified job.
- In LSF Version 10.1 Fix Pack 10, the new **bjobs -script** option displays the job script for the specified job from the LSF info directory.
- In LSF Version 10.1 Fix Pack 10, the new **bjobs -p** option displays the total global limit amount for the resource allocation limit in the **`Global`** field if the job is blocked by the global limit value.
- In LSF Version 10.1 Fix Pack 10, the **bjobs -o** option now has the following fields:
    - `suspend_reason` shows the user-specified reason for suspending (stopping) the job.
    - `resume_reason` shows the user-specified reason for resuming the job.
    - `kill_issue_host` shows the host that issued the job kill request.
    - `suspend_issue_host` shows the host that issued the job suspend (stop) request.
    - `resume_issue_host` shows the host that issued the job resume request.
- In LSF Version 10.1 Fix Pack 11, the **bjobs -o** option now allows you to display a unit prefix for the following resource fields: , **`max_mem`**, **`avg_mem`**, **`memlimit`**, **`swap`**, **`swaplimit`**, **`corelimit`**, **`stacklimit`**, and **`hrusage`** (for **`hrusage`**, the unit prefix is for **`mem`** and **`swap`** resources only). In addition, the default width for these resource fields except **`hrusage`** are increased from 10 to 15. That is, the following output fields now have a default width that is increased from 10 to 15:**`mem`**, **`max_mem`**, **`avg_mem`**, **`memlimit`**, **`swap`**, **`swaplimit`**, **`corelimit`**, and **`stacklimit`**.
- In LSF Version 10.1 Fix Pack 11, the **bjobs -l** option now shows additional Nvidia Multi-Instance GPU (MIG) information.
- In LSF Version 10.1 Fix Pack 12, the **bjobs -cname** command option is deprecated and will be removed in a future version.

bjgroup

- **IDLE_TIME**: In LSF 10.1 Fix Pack 13, this new default output field displays the idle time value of each job group.

bkill

- -d: In LSF 10.1 Fix Pack 13, this new option kills jobs, and records jobs as **DONE** after the jobs exit.
- -stat: In LSF 10.1 Fix Pack 13, this new option kills jobs in the specified status.

blimits

- In LSF Version 10.1 Fix Pack 5 and Fix Pack 6, the new **blimits -a** option shows all resource allocation limits, even if they are not being applied to running jobs.
- In LSF Version 10.1 Fix Pack 9, the **blimits** command now displays the application profiles on which limits are enforced in the **APPS** column. You can also display limits for specific application profiles by using the new -A option.
- In LSF Version 10.1 Fix Pack 10, when using the LSF multicluster capability, the new **blimits -gl** command option displays current usage of global limits that are configured in the Limit sections of the lsb.globalpolicies file instead of the resource configurations in the Limit sections of the lsb.resources file.
  If specified with the **blimits -c** option (that is, by running the **blimits -gl -c** command option), displays the global limits configurations in the lsb.globalpolicies file instead of the resource configurations in the lsb.resources file.

- In LSF Version 10.1 Fix Pack 13, the new **blimits -o** command option displays information about resource allocation limits, using customized column widths for the output, rather than the default column widths for each resource type.

bmgroup

- For the IBM Spectrum LSF multicluster capability resource leasing model, the **HOSTS** column now displays a list of leased-in hosts in the form ***host_name*@*cluster_name*** by default.
  If the LSB_BMGROUP_ALLREMOTE_EXPAND=N parameter is configured in the `lsf.conf` file or as an environment variable, leased-in hosts are represented by a single keyword **allremote** instead of being displayed as a list.

- In LSF Version 10.1 Fix Pack 12, the **bmgroup -cname** command option is deprecated and will be removed in a future version.

bmod

- In LSF Version 10.1 Fix Pack 1, the **bmod** command now exports mem and swp values in the rusage[] string to the following corresponding environment variables for **esub**:
  - If the **bmod** command has a mem value in the rusage[] string, the LSB_SUB_MEM_USAGE variable is set to the mem value in the temporary **esub** parameter file that the LSB_SUB_PARAM_FILE environment variable points to.
  - If the **bmod** command has a swp value in the rusage[] string, the LSB_SUB_SWP_USAGE variable is set to the mem value in the temporary **esub** parameter file that the LSB_SUB_PARAM_FILE environment variable points to.

- In LSF Version 10.1 Fix Pack 2, you can now use the `ZB` (or `Z`) unit when specifying resource requirements and limits.
- In LSF Version 10.1 Fix Pack 9, a -datachk option is added to **bmod -data** to perform a full file sanity check since this role is moved to the transfer job. This equalizes modification performance between jobs with and without data requirements. The -datachk option can be specified only with the **-data** command. If the data requirement is for a tag, this option has no effect.
- `bmod -Ne`: In LSF Version 10.1 Fix Pack 9, this new option enables the **sbatchd** daemon to send mail only when the job exits (that is, when the job is under **`Exit`** status). This ensures than an email notification is only sent on a job error. The `bmod -Nn` option cancels the -Ne option in addition to the -N option.
- `bmod -notify` and **-notifyn**: In LSF Version 10.1 Fix Pack 9, the new -notify option requests that the user be notified when the job reaches any of the specified states. The new -notifyn option cancels the -notify option.
- `bmod -M`: In LSF Version 10.1 Fix Pack 9, you can now set a hard memory limit by adding an exclamation point (`!`) to the end of the memory limit. LSF kills the job as soon as it exceeds this hard memory limit and does not wait for the host memory and swap threshold to be reached.
- `bmod -jobaff` and `-jobaffn`: In LSF Version 10.1 Fix Pack 10, this new command option modifies host attribute affinity preferences for the job. This allows you to select hosts for job scheduling based on which attributes are on the hosts, or which hosts or compute units are running specific jobs. The new -jobaffn option cancels the -jobaff option.
- `bmod -rcacct` and `-rcacctn`: In LSF Version 10.1 Fix Pack 11, this new command option modifies the LSF resource connector account name that is assigned to a job, which is then tagged to the resource connector host that runs the job. The new -rcacctn option either removes the resource connector account name or resets it to the default value. These options are available only if you enabled `ENABLE_RC_ACCOUNT_REQUEST_BY_USER=Y` in the lsb.params file.

bpeek

- In LSF Version 10.1 Fix Pack 3, the **bpeek -f** command option now exits when the peeked job is completed.
  If the peeked job is requeued or migrated, the **bpeek** command only exits if the job is completed again. In addition, the **bpeek** command cannot get the new output of the job. To avoid these issues, abort the previous **bpeek -f** command and rerun the **bpeek -f** command after the job is requeued or migrated.

bpost

- In LSF Version 10.1 Fix Pack 8, the new **bpost -N** command option sends a message (from the -d option) at the specified notification level to LSF Application Center Notifications as specified by the LSF_AC_PNC_URL parameter in the lsf.conf file.

bqueues

- The **bqueues -r** option now displays the global fair share policy name for the participating queue, and displays the remote share load (**`REMOTE_LOAD`** column) for each share account in the queue.
- The **bqueues -l** option now shows information on pending time limits and eligible pending time limits for jobs in the queue.
- In LSF Version 10.1 Fix Pack 2, the new **bqueues -o** option customizes specific fields that the **bqueues** command displays.
- In LSF Version 10.1 Fix Pack 2, the new **bqueues -json** option displays customized **bqueues -o** command output in JSON format. The -json option must be used together with the -o command option.

- In LSF Version 10.1 Fix Pack 7, the **bqueues -r** and **-l** command options now display the normalized fair share factors in the **NORM_FS** column, if these factors are not zero.
- -o: In LSF 10.1 Fix Pack 13, this option now has the following fields for limits and resources:
  - max_corelimit, max_cpulimit, default_cpulimit, max_datalimit, default_datalimit, max_filelimit, max_memlimit, default_memlimit, max_processlimit, max_runlimit, default_runlimit, max_stacklimit, max_swaplimit, max_tasklimit, min_tasklimit, default_tasklimit, max_threadlimit, default_threadlimit, res_req, hosts.
  - The following resource limit fields show the same content as their corresponding maximum resource limit fields: corelimit, cpulimit, datalimit, filelimit, memlimit, processlimit, runlimit, stacklimit, swaplimit, tasklimit, threadlimit. For example, corelimit is the same as max_corelimit.
- -l: In LSF 10.1 Fix Pack 13, this option now displays the **STARTED_JOBS** and **RESERVED_JOBS** fields if fair share_JOB_COUNT=Y is enabled in the lsb.params file.
  In addition, the new IMPT_JOBLIMIT and IMPT_TASKLIMIT parameters allow you to specify how many multicluster jobs or tasks, from remote clusters, that can be configured at the receive-jobs queue.

bread

- In LSF Version 10.1 Fix Pack 3, the **bread** displays the failure reason of the **cleartool setview** command when it is called by the daemon wrapper with the LSF Integration for Rational ClearCase.
- In LSF Version 10.1 Fix Pack 5 and Fix Pack 6, the new **bread -w** option displays messages and attached data files from a job in wide format without truncating fields.
- In LSF Version 10.1 Fix Pack 8, the new **bread -N** command option displays the **NOTIFICATION** field to indicate whether the message was sent to LSF Application Center Notifications. A hyphen (-) indicates that this was not sent to LSF Application Center Notifications (either because it is a normal message that was sent without the -N option or the LSF_AC_PNC_URL parameter is not configured correctly in the lsf.conf file).

bresize

- **request**: In LSF Version 10.1 Fix Pack 3, this new subcommand requests for additional tasks to be allocated to a running resizable job. After running this command, the job is no longer autoresizable unless you requeue or rerun the job.

bresources

- The **bresources -g** option shows additional information for package and slot type guarantees.

brestart

- The **brestart -R** option reserves resources when you restart a stopped checkpointable job. You can specify resources with the **brestart -R** command when you restart the job. Use the **brestart** command to specify multiple -R options for multiple resource requirement strings, specify compound resource requirements, and specify alternative resource requirements.
- In LSF Version 10.1 Fix Pack 2, you can now use the ZB (or Z) unit when specifying resource requirements and limits.
- brestart -Ne: In LSF Version 10.1 Fix Pack 9, this new option enables the **sbatchd** daemon to send mail only when the job exits (that is, when the job is under **Exit** status). This ensures than an email

notification is only sent on a job error.

bresume

- In LSF Version 10.1 Fix Pack 10, the new **bresume -C** command option specifies a reason for resuming the job.

brsvadd

- In LSF Version 10.1 Fix Pack 4, the new **brsvadd -q** option specifies queues are allowed to run on the advance reservation hosts even if the jobs cannot finish before the advance reservation starts (that is, the run limit or estimated run time continues after the start time of the advance reservation).
- In LSF Version 10.1 Fix Pack 4, the new **brsvadd -nosusp** option enables LSF to not suspend non-advance reservation jobs that are running on the advance reservation hosts when the first advance reservation job starts. Non-advance reservation jobs continue to run, and advance reservation jobs do not start until resources are available.
- In LSF Version 10.1 Fix Pack 4, the new **brsvadd -E** option specifies a script (pre-script) to run on the master host before the advance reservation starts.
- In LSF Version 10.1 Fix Pack 4, the new **brsvadd -Et** option specifies the amount of time, in minutes, before the advance reservation starts for LSF to run the prescript and to stop dispatching new jobs to the advance reservation hosts.
- In LSF Version 10.1 Fix Pack 4, the new **brsvadd -Ep** option specifies a script (post-script) to run on the master host when the advance reservation expires.
- In LSF Version 10.1 Fix Pack 4, the new **brsvadd -Ept** option specifies the amount of time, in minutes, before the expiry of the advance reservation for LSF to run the post-script.
- In LSF Version 10.1 Fix Pack 5 and Fix Pack 6, the **brsvadd -R** option now allows the same string to take effect, in addition to the select string.

brsvmod

- In LSF Version 10.1 Fix Pack 4, the new **brsvmod -q** option changes the queues whose jobs are allowed to run on the advance reservation hosts even if the jobs cannot finish before the advance reservation starts (that is, the run limit or estimated run time continues after the start time of the advance reservation).
- In LSF Version 10.1 Fix Pack 4, the new **brsvmod -qn** option removes the -q option from the advance reservation.
- In LSF Version 10.1 Fix Pack 4, the new **brsvmod -nosusp** option enables LSF to no longer suspend non-advance reservation jobs that are running on the advance reservation hosts when the first advance reservation job starts. Non-advance reservation jobs continue to run, and advance reservation jobs do not start until resources are available.
- In LSF Version 10.1 Fix Pack 4, the new **brsvmod -nosuspn** option removes the -nosusp option from the advance reservation.
- In LSF Version 10.1 Fix Pack 4, the new **brsvmod -E** option replaces the script (pre-script) to run on the master host before the advance reservation starts.
- In LSF Version 10.1 Fix Pack 4, the new **brsvmod -En** option removes the advance reservation prescript so that no scripts are run when the advance reservation starts.
- In LSF Version 10.1 Fix Pack 4, the new **brsvmod -Et** option changes the amount of time, in minutes, before the start of the advance reservation for LSF to run the prescript and to stop dispatching new jobs to the advance reservation hosts.
- In LSF Version 10.1 Fix Pack 4, the new **brsvmod -Etn** option removes the -Et option from the advance reservation and enables LSF to run the pre-script at the start time of the advance reservation.

- In LSF Version 10.1 Fix Pack 4, the new **brsvmod -Ep** option replaces the script (post-script) to run on the master host when the advance reservation expires.
- In LSF Version 10.1 Fix Pack 4, the new **brsvmod -Epn** option removes the advance reservation post-script so that no scripts are run when the advance reservation expires.
- In LSF Version 10.1 Fix Pack 4, the new **brsvmod -Ept** option changes the amount of time, in minutes, before the expiry of the advance reservation for LSF to run the post-script.
- In LSF Version 10.1 Fix Pack 4, the new **brsvmod -Eptn** option removes the -Ept option from the advance reservation and enables LSF to run the post-script at the expiry time of the advance reservation.
- In LSF Version 10.1 Fix Pack 5 and Fix Pack 6, the **brsvmod -R** option now allows the same string to take effect, in addition to the select string.
- In LSF Version 10.1 Fix Pack 11, the **brsvmod addhost** command now has the -f option, which enables LSF to dynamically select hosts based on the resource requirements as specified in the -R or -m command options..

brsvs

- In LSF Version 10.1 Fix Pack 4, the **brsvs -l** option now displays queues that can use the advance reservation hosts before the advance reservation starts, the prescript, prescript time, post-script, post-script time, and whether non-advance reservation jobs are allowed to continue running after the first advance reservation starts.

bsla

- In LSF Version 10.1 Fix Pack 7, the **bsla** command displays information on the allocated guarantee hosts in a new section (**`USED GUARANTEE HOSTS`**) if `LSB_GSLA_DISPLAY_ALLOC_HOSTS=Y` is enabled in lsf.conf. **bsla** only displays this new section if there are jobs running in the SLA.

bstatus

- In LSF Version 10.1 Fix Pack 3, the **bstatus** displays the failure reason of the **cleartool setview** command when it is called by the daemon wrapper with the LSF Integration for Rational ClearCase.

bstop

- In LSF Version 10.1 Fix Pack 10, the new **bstop -C** command option specifies a reason for stopping the job.

bsub

- The **bsub -a** command now runs **epsub** executable files after the job is submitted. After the job is submitted, LSF runs any mandatory **epsub** executable files (as specified by using the parameter LSB_ESUB_METHOD), followed by any application-specific **epsub** executable files (with *.application_name* in the file name) as specified by the -a option.
- The **bsub -eptl** command specifies the eligible pending time limit for the job.
- The **bsub -ptl** command specifies the pending time limit for the job.
- In LSF Version 10.1 Fix Pack 1, the **bsub** command now exports mem and swp values in the rusage[] string to the following corresponding environment variables for **esub**:

    - If the **bsub** command has a mem value in the rusage[] string, the LSB_SUB_MEM_USAGE variable is set to the mem value in the temporary **esub** parameter file that the LSB_SUB_PARAM_FILE environment variable points to.
    - If the **bsub** command has a swp value in the rusage[] string, the LSB_SUB_SWP_USAGE variable is set to the mem value in the temporary **esub** parameter file that the LSB_SUB_PARAM_FILE environment variable points to.
- In LSF Version 10.1 Fix Pack 2, you can now use the `ZB` (or `Z`) unit when specifying resource requirements and limits.
- In LSF Version 10.1 Fix Pack 3, you can now use the **bsub -m** option to specify remote hosts when using the job forwarding model with the LSF multicluster capability, using *host_name@cluster_name*.
- In LSF Version 10.1 Fix Pack 3, you can now specify multiple email addresses with the **bsub -u** option by enclosing the string in quotation marks and using a space to separate each email address. The total length of the address string cannot be longer than 511 characters.
- LSF Version 10.1 Fix Pack 5 and Fix Pack 6 now have the following **bsub** options for running jobs with IBM Cluster Systems Manager (CSM):
- In LSF Version 10.1 Fix Pack 6, when using the **bsub -P** option to specify the project name, that project name can now be up to 511 characters long (previously, this limit was 59 characters).
- In LSF Version 10.1 Fix Pack 7, you can now run job scripts directly from the **bsub** command line instead of using the < redirect. The job script must be an ASCII text file and not a binary file, but it does not have to be an executable file.
- In LSF Version 10.1 Fix Pack 7, you can now use the `bsub -R cu [pref=bestfit]` keyword to use the best-fit allocation algorithm for jobs with resource requirements. This allocation algorithm places the job using the fewest number of compute units as possible while preferring compute units that are already occupied.
- `bsub -gpu` has the following changes in LSF Version 10.1 Fix Pack 8:
    - You can now specify `aff=no` in the `bsub -gpu` command option to relax GPU affinity while maintaining CPU affinity. By default, `aff=no` is set to maintain strict GPU-CPU affinity binding.
    - You can now specify `mps=per_socket` in the `bsub -gpu` command option to enable LSF to start one MPS daemon per socket per job.
    - You can now specify `mps=per_gpu` in the `bsub -gpu` command option to enable LSF to start one MPS daemon per GPU per job.
- In LSF Version 10.1 Fix Pack 8, you can now use the LSB_DOCKER_PLACE_HOLDER keyword instead of a command to submit a Docker job with a Docker entry point image.
- In LSF Version 10.1 Fix Pack 8, the maximum integer that you can specify for the following resource limits is increased from 32 bit (2^{31}) to 64 bit (2^{63}):
    - memory limit (**bsub -M**)
    - swap limit (**bsub -v**)
    - core file size limit (**bsub -C**)
    - stack limit (**bsub -S**)
    - data segment size limit (**bsub -D**)
    - file size limit (**bsub -F**)
- In LSF Version 10.1 Fix Pack 8, you can now use the new `bsub -smt` command option to specify the SMT mode for CSM jobs.
- In LSF Version 10.1 Fix Pack 8, you can now use the `bsub -G` command option to exclude fair share groups from being associated with a job by using a tilde (~) to specify any fair share groups from which you want to exclude for the job. Use a space-separated list of tildes (~) to exclude multiple groups.
- In LSF Version 10.1 Fix Pack 9, a -datachk option is added to **bsub -data** to perform a full file sanity check since this role is moved to the transfer job. This equalizes modification performance between jobs with and without data requirements. The -datachk option can be specified only with the **-data** command. If the data requirement is for a tag, this option has no effect.
- `bsub -R`: In LSF Version 10.1 Fix Pack 9, you can now specify the resource reservation method (by task, by job, or by host) in the rusage string by using the /task, /job, or /host keyword after the numeric

value of the resource. You can only specify resource reservation methods for consumable resources. In addition, you can now enable LSF to stripe tasks of a parallel job across the free resources of the candidate hosts by using the stripe keyword in the span string of the resource requirements.

- `bsub -gpu`: In LSF Version 10.1 Fix Pack 9, you can now specify `mps=yes,share`, `mps=per_socket,share`, and `mps=per_gpu,share` in the `bsub -gpu` command option to enable LSF to share the MPS daemon on the host, socket, or GPU for jobs that are submitted by the same user with the same resource requirements.
  That is, you can now add "`,share`" to the mps value to enable MPS daemon sharing for the host, socket, or GPU.

  In addition, you can now assign the number of GPUs per task or per host by specifying `num=`*`number`*`/task` or `num=`*`number`*`/host`. By default, the number of GPUs is still assigned per host.

- `bsub -Ne`: In LSF Version 10.1 Fix Pack 9, this new option enables the **sbatchd** daemon to send mail only when the job exits (that is, when the job is under **`Exit`** status). This ensures than an email notification is only sent on a job error.
- `bsub -notify`: In LSF Version 10.1 Fix Pack 9, this new option requests that the user be notified when the job reaches any of the specified states.
- `bsub -M`: In LSF Version 10.1 Fix Pack 9, you can now set a hard memory limit by adding an exclamation point (`!`) to the end of the memory limit. LSF kills the job as soon as it exceeds this hard memory limit and does not wait for the host memory and swap threshold to be reached.
- `bsub -K`: In LSF Version 10.1 Fix Pack 10, this command option now displays the job ID of a job after it is finished if the LSB_SUBK_SHOW_JOBID parameter is set to `Y` or `y` in the lsf.conf file.
- `bsub -jobaff`: In LSF Version 10.1 Fix Pack 10, this new command option specifies host attribute affinity preferences for the job. This allows you to select hosts for job scheduling based on which attributes are on the hosts, or which hosts or compute units are running specific jobs.
- `bsub -json`: In LSF Version 10.1 Fix Pack 10, this new command option submits a job using a JSON file to specify job submission options.
- `bsub -yaml`: In LSF Version 10.1 Fix Pack 10, this new command option submits a job using a YAML file to specify job submission options.
- `bsub -gpu` has the following changes in LSF Version 10.1 Fix Pack 10:
  - You can now add the `,nocvd` keyword to the existing `mps` value in the GPU resource requirements string to disable the CUDA_VISIBLE_DEVICES environment variable for MPS jobs.
  - You can now specify `block=yes` in the GPU resource requirements string to enable block distribution of allocated GPUs.
  - You can now specify `gpack=yes` in the GPU resource requirements string to enable pack scheduling for shared mode GPU jobs.
- `bsub -rcacct`: In LSF Version 10.1 Fix Pack 11, this new command option assigns an LSF resource connector account name to a job and tags the account name to the resource connector host that runs the job. This option is available only if you enabled `ENABLE_RC_ACCOUNT_REQUEST_BY_USER=Y` in the lsb.params file.
- In LSF Version 10.1 Fix Pack 11, the **bsub -gpu** command option and **bsub -R "rusage[]"** resource string now has the mig option to specify Nvidia Multi-Instance GPU (MIG) device requirements.
- In LSF Version 10.1 Fix Pack 12, the **bsub -network** command option is deprecated and will be removed in a future version.

bsubmit (New)

In LSF Version 10.1 Fix Pack 11, the **bsubmit** command allows job submission users to submit jobs as another job execution user. Define the user mapping policies for the **bsubmit** by creating the lsf.usermapping file in the $LSF_ENVDIR directory.

To use this command, you must download and deploy the **bsubmit** executable file, which is a wrapper for the **bsub** command. For more details, refer to the following link: https://github.com/IBMSpectrumComputing/lsf-utils/tree/master/bsubmit

bswitch

- In LSF Version 10.1 Fix Pack 4, the new **bswitch -a** command option specifies an application-specific **eswitch** executable file that you want LSF to associate with the job switch request.

bugroup

- The -l option now displays the global fair share policy (FS_POLICY) configuration for the user group.
- In LSF Version 10.1 Fix Pack 5 and Fix Pack 6, the **bugroup** command now displays a new field, **PRIORITY**, which shows the absolute priority scheduling (APS) user group factors.

busers

- The -w option now displays two new fields: **PJOBS** and **MPJOBS**.
- In LSF Version 10.1 Fix Pack 5 and Fix Pack 6, the **busers** command now displays a new field, **PRIORITY**, which shows the absolute priority scheduling (APS) factors for the specified users or user groups.
- In LSF Version 10.1 Fix Pack 9, the new **busers -o** option customizes specific fields that the **busers** command displays.

bwait (New)

In LSF Version 10.1 Fix Pack 2, the **bwait** command pauses and waits for the specified job condition to occur before the command returns. End users can use this command to reduce workload on the **mbatchd** daemon by including **bwait** in a user script for running jobs instead of using the **bjobs** command in a tight loop to check the job status.

- In LSF 10.1 Fix Pack 13, you can now prevent LSF from using the **bwait** command within a job by setting the new LSB_BWAIT_IN_JOBS parameter in the lsf.conf file to N.

ch

In LSF Version 10.1 Fix Pack 12, this command is deprecated and will be removed in a future version.

epsub

- In LSF Version 10.1 Fix Pack 5 and Fix Pack 6, you can now define a default **epsub** program by adding an executable file named epsub (with no application name in the file name) in the LSF_SERVERDIR directory.

elim

In LSF Version 10.1 Fix Pack 12, all **elim.gpu.*** ELIMs are deprecated and removed.

hostsetup

- `hostsetup --setuid`: In LSF Version 10.1 Fix Pack 10, this new command option enables the setuid bit for LSF administration commands (**badmin**, **lsadmin**, **egosh**, **utmpreg**, **swtbl_api**, **ntbl_api**, **lstbl_nid**, and **swtbl_poe**). Since this allows LSF administration commands to run with root privileges, do not enable the setuid bit if you do not want these LSF commands to run with root privileges.

lim

- In LSF Version 10.1 Fix Pack 4, the **lim -t** and **-T** options ignore the NUMA level if the host only has a single NUMA node.
- In LSF Version 10.1 Fix Pack 4, the **lim -t** option shows the number of NUMA nodes as a hyphen (`-`) instead of `0` if the host has no NUMA nodes.
- -T: In LSF 10.1 Fix Pack 13, this option now display the physical index for the socket and core, similar to the existing output for the **bhosts -aff** command.

lsadmin

In LSF Version 10.1 Fix Pack 11, the **lsadmin** command no longer has the setuid bit set at installation. This means that this command no longer runs with root privileges.

- In LSF Version 10.1 Fix Pack 11, the following LIM and RES daemon control commands are obsolete and will be deprecated in a future release:
  - **lsadmin limrestart**
  - **lsadmin limstartup**
  - **lsadmin limshutdown**
  - **lsadmin resrestart**
  - **lsadmin resstartup**
  - **lsadmin resshutdown**

  Instead of these obsolete commands, use the `bctrld` *`action`* `lim` or `bctrld` *`action`* `res` (where *`action`* is `restart`, `start`, or `stop`) to control the LIM or RES daemons.

lseligible

In LSF Version 10.1 Fix Pack 12, this command is deprecated and will be removed in a future version.

lsfmon

In LSF Version 10.1 Fix Pack 12, this command is deprecated and will be removed in a future version.

lsgrun

- In LSF Version 10.1 Fix Pack 10, tasks in the execution side that the **lsgrun** command runs use the UNIX group information that is set by the user on the client side, if the LSF_UGROUP_TRANSFER parameter is enabled in the lsf.conf file.

lshosts

- In LSF Version 10.1 Fix Pack 6, the new **lshosts -gpu** option displays GPU topology information for the cluster.
- In LSF Version 10.1 Fix Pack 7, the new **lshosts -o** option customizes specific fields that the **lshosts** command displays.
- In LSF Version 10.1 Fix Pack 7, the new **lshosts -json** option displays customized **lshosts -o** command output in JSON format. The -json option must be used together with the -o command option.
- In LSF Version 10.1 Fix Pack 10, the **LOCATION** field in the **lshosts -s** command displays **ALL** instead of displaying each individual host in the cluster if the LOCATION parameter in the lsf.cluster.*clustername* file is set to `all` to indicate that the resource is shared by all hosts in the cluster. Use the new **lshosts -loc** command option to display each individual host in the cluster.
- In LSF Version 10.1 Fix Pack 11, the **lshosts -gpu** command now displays the GPU vendor type (AMD or Nvidia).
- In LSF Version 10.1 Fix Pack 11, the **lshosts -gpu** command option now shows whether the GPU supports MIG functions.
- In LSF Version 10.1 Fix Pack 11, the new **lshosts -mig** command option displays detailed information on MIG instances. The -mig option must be specified together with the **lshosts -gpu** command option.
- In LSF Version 10.1 Fix Pack 12, the **lshosts -cname** command option is deprecated and will be removed in a future version.
- -T: In LSF 10.1 Fix Pack 13, this option now display the physical index for the socket and core, similar to the existing output for the **bhosts -aff** command.

lsload

- In LSF Version 10.1 Fix Pack 12, the **lsload -cname** command option is deprecated and removed.

lslogin

In LSF Version 10.1 Fix Pack 12, this command is deprecated and will be removed in a future version.

lsmon

In LSF Version 10.1 Fix Pack 12, this command is deprecated and will be removed in a future version.

lsportcheck (New)

In LSF Version 10.1 Fix Pack 9, this new command displays ports that LSF is currently using or the LSF ports that will be used before starting LSF..

lsload

- In LSF Version 10.1 Fix Pack 3, the new **lsload -o** option customizes specific fields that the **lsload** command displays.
- In LSF Version 10.1 Fix Pack 3, the new **lsload -json** option displays customized **lsload -o** command output in JSON format. The -json option must be used together with the -o command option.
- In LSF Version 10.1 Fix Pack 4, the **lsload -o** option now has the following fields:
  - `gpu_status*` shows the status of the GPU (**ok**, **error**, or **warning**). If more than 1 GPU is reported, an index is appended to the resource name, starting at 0. For example, `gpu_status0` and `gpu_status1`.
  - `gpu_error*` shows the detailed error or warning message if the `gpu_status*` field is not **ok**. If more than 1 GPU is reported, an index is appended to the resource name, starting at 0. For

example, `gpu_status0` and `gpu_status1`.

- In LSF Version 10.1 Fix Pack 6, the new **lsload -gpu** option displays host-based GPU information.
- In LSF Version 10.1 Fix Pack 6, the new **lsload -gpuload** option displays GPU-based information.
- In LSF Version 10.1 Fix Pack 10, the **lsload -s** and **lsload -l** command options no longer display GPU-based information if `LSF_GPU_RESOURCE_IGNORE=Y` is defined in the lsf.conf file. Use the **lsload -gpu** or **lsload -gpuload** options to view this information.
- In LSF Version 10.1 Fix Pack 10, the **`LOCATION`** field in the **lsload -s** command option displays **`ALL`** instead of displaying each individual host in the cluster if the LOCATION parameter in the lsf.cluster.*clustername* file is set to `all` to indicate that the resource is shared by all hosts in the cluster. Use the new **lsload -loc** command option to display each individual host in the cluster.
- In LSF Version 10.1 Fix Pack 11, the **lsload -gpuload** command option now displays the **`gpu_power`** column to show the power utilization per GPU on the host.
- In LSF Version 10.1 Fix Pack 12, the **lsload -cname** command option is deprecated and will be removed in a future version.

lsltasks

In LSF Version 10.1 Fix Pack 12, this command is deprecated and will be removed in a future version.

lsreghost (New)

The **lsreghost** command directly registers LSF host names and IP addresses with LSF servers so that LSF servers can internally resolve these hosts without requiring a DNS server. You can resolve the host name and IP address for LSF hosts with non-static IP addresses in environments where DNS is not able to properly resolve these hosts after their IP addresses change.

lsrtasks

In LSF Version 10.1 Fix Pack 12, this command is deprecated and will be removed in a future version.

lsrun

- In LSF Version 10.1 Fix Pack 10, tasks in the execution side that the **lsrun** command runs use the UNIX group information that is set by the user on the client side, if the LSF_UGROUP_TRANSFER parameter is enabled in the lsf.conf file.

lstcsh

In LSF Version 10.1 Fix Pack 12, this command is deprecated and will be removed in a future version.

New and changed configuration parameters and environment variables

The following configuration parameters and environment variables are new or changed for LSF 10.1.

Note: Starting in Fix Pack 14, instead of reading this topic, refer to the What's new information for each fix pack (which contain changed and new details, including direct references to affected topics).

Files that use automatic time-based configuration

All files where you specify automatic time-based configurations with if-else constructs now allow you to specify time zones whenever you specify time windows, including the lsb.applications, lsb.hosts, lsb.params, lsb.queues, and lsb.resources. LSF supports all standard time zone abbreviations. If you do not specify the time zone, LSF uses the local system time zone.

install.config

- ENABLE_CGROUP: In LSF Version 10.1 Fix Pack 2, this new parameter enables LSF to track processes' CPU and memory accounting based on Linux **cgroup** memory and cpuacct subsystems. If set to `Y`, the installer sets parameters in the lsf.conf file enable these functions in LSF.
- ENABLE_GPU: In LSF Version 10.1 Fix Pack 2, this new parameter enables LSF to support GPUs so that applications can use GPU resources in a Linux environment. LSF supports parallel jobs that require GPUs based on availability. If set to `Y`, the installer sets parameters in the lsf.conf file enable these functions in LSF, and adds resource definitions to the lsf.cluster.*cluster_name* and lsf.shared files for GPU resources.

lsb.applications

- ELIGIBLE_PEND_TIME_LIMIT: Specifies the eligible pending time limit for jobs in the application profile.
- PEND_TIME_LIMIT: Specifies the pending time limit for jobs in the application profile.
- CONTAINER: In LSF Version 10.1 Fix Pack 1, this new parameter enables Docker container jobs to run in the application profile by using the docker[] keyword.
  In LSF Version 10.1 Fix Pack 2, enables Shifter and Singularity container jobs to run in the application profile by using the shifter[] and singularity[] keywords.

  In LSF Version 10.1 Fix Pack 3, allows you to specify a pre-execution script to run before the container job runs by specifying an at sign (`@`) and a full file path to the script in the option() keyword. The output of this script is used as container startup options.

  In LSF Version 10.1 Fix Pack 4, the starter[] keyword is obsolete.

- EXEC_DRIVER: In LSF Version 10.1 Fix Pack 4, this new parameter specifies the execution driver framework for the application profile.
- #INCLUDE: In LSF Version 10.1 Fix Pack 5 and Fix Pack 6, you can now use this directive in any place in this file. Previously, you could only use this directive in the beginning of the lsb.applications file.
- PRIORITY: In LSF Version 10.1 Fix Pack 5 and Fix Pack 6, this new parameter specifies a priority that is used as a factor when calculating the job priority for absolute priority scheduling (APS).
- ESTIMATED_RUNTIME: In LSF Version 10.1 Fix Pack 5 and Fix Pack 6, this parameter is introduced to configure estimated runtimes for jobs in an application, and is meant to replace the existing RUNTIME parameter from the lsb.applications file. Can also be configured at the queue level (lsb.queues file) and cluster level (lsb.params file).
- PLAN: In LSF Version 10.1 Fix Pack 5 and Fix Pack 6, this parameter is introduced for use when plan-based scheduling is enabled with the parameter ALLOCATION_PLANNER=Y. This parameter controls whether or not jobs are candidates for plan-based scheduling. Can also be configured at the queue level (lsb.queues file), cluster level (lsb.params file), and job level.

- GPU_REQ: In LSF Version 10.1 Fix Pack 6, this new parameter specifies the GPU requirements for the application profile.
  In LSF Version 10.1 Fix Pack 8, GPU_REQ has the following changes:
  - You can now specify `aff=no` in the GPU requirements to relax GPU affinity while maintaining CPU affinity. By default, `aff=no` is set to maintain strict GPU-CPU affinity binding.
  - You can now specify `mps=yes,share` in the GPU requirements to enable LSF to share the one MPS daemon per host for all jobs that are submitted by the same user with the same resource requirements, and these jobs use the same MPS daemon on the host.
  - You can now specify `mps=per_socket` in the GPU requirements to enable LSF to start one MPS daemon per socket per job. You can also use the `mps=per_socket,share` to enable LSF to share the one MPS daemon per socket for all jobs that are submitted by the same user with the same resource requirements, and these jobs use the same MPS daemon for the socket.
  - You can now specify `mps=per_gpu` in the GPU requirements to enable LSF to start one MPS daemon per GPU per job. You can also use the `mps=per_gpu,share` to enable LSF to share the one MPS daemon per GPU for all jobs that are submitted by the same user with the same resource requirements, and these jobs use the same MPS daemon for the GPU.
- WATCHDOG: In LSF Version 10.1 Fix Pack 8, this new parameter enables LSF to use the watchdog feature to regularly run external scripts that check application data, logs, and other information. LSF can use these scripts to pass on the job information.
- In LSF Version 10.1 Fix Pack 8, the maximum integer that you can specify for the following resource limits is increased from 32 bit (2^{31}) to 64 bit (2^{63}):
  - memory limit (MEMLIMIT parameter)
  - swap limit (SWAPLIMIT parameter)
  - core file size limit (CORELIMIT parameter)
  - stack limit (STACKLIMIT parameter)
  - data segment size limit (DATALIMIT parameter)
  - file size limit (FILELIMIT parameter)
- GPU_REQ: In LSF Version 10.1 Fix Pack 9, you can now specify `mps=yes,share`, `mps=per_socket,share`, and `mps=per_gpu,share` in the GPU requirements to enable LSF to share the MPS daemon on the host, socket, or GPU for jobs that are submitted by the same user with the same resource requirements.
  That is, you can now add "`,share`" to the mps value to enable MPS daemon sharing for the host, socket, or GPU.

  In addition, you can now assign the number of GPUs per task or per host by specifying `num=`*`number`*`/task` or `num=`*`number`*`/host`. By default, the number of GPUs is still assigned per host.

- RES_REQ: In LSF Version 10.1 Fix Pack 9, you can now specify the resource reservation method (by task, by job, or by host) in the rusage string by using the /task, /job, or /host keyword after the numeric value in the rusage string. You can only specify resource reservation methods for consumable resources.
  In addition, you can now enable LSF to stripe tasks of a parallel job across the free resources of the candidate hosts by using the stripe keyword in the span string of the resource requirements.

- DOCKER_IMAGE_AFFINITY: In LSF Version 10.1 Fix Pack 9, this new parameter enables LSF to give preference for execution hosts that already have the requested Docker image when submitting or scheduling Docker jobs.
- GPU_REQ has the following changes in LSF Version 10.1 Fix Pack 10:
  - You can now add the "`,nocvd`" keyword to the existing `mps` value in the GPU resource requirements string to disable the CUDA_VISIBLE_DEVICES environment variable for MPS jobs.
  - You can now specify `block=yes` in the GPU resource requirements string to enable block distribution of allocated GPUs.

    - You can now specify `gpack=yes` in the GPU resource requirements string to enable pack scheduling for shared mode GPU jobs.
- USE_PAM_CREDS has the following changes in LSF Version 10.1 Fix Pack 10:
    - You can now specify the `session` keyword to enable LSF to open a PAM session when submitting jobs to Linux hosts using PAM..
    - You can now specify the `limits` keyword to apply the limits that are specified in the PAM configuration file to an application. This is functionally identical to enabling USE_PAM_CREDS=y except that you can define `limits` together with the `session` keyword.
- CONTAINER: LSF Version 10.1 Fix Pack 11 enables the following new containers with this parameter:
    - Pod Manager (Podman) container jobs run in the application profile by using the docker[] keyword to run Podman jobs with the Docker execution driver. The options[] keyword specifies Podman job run options for the **podman run** command, which are passed to the job container. Because Podman uses the Docker execution driver to run Podman container jobs, **podman** and **docker** command options are not compatible, and execution driver permissions are not the same, you cannot use LSF to run Docker container jobs if you are using LSF to run Podman container jobs.
    - Enroot container jobs run in the application profile by using the new enroot[] keyword to run Enroot jobs with the Enroot execution driver. The options[] keyword specifies Enroot job run options for the **enroot start** command, which are passed to the job container.
- EXEC_DRIVER: In LSF Version 10.1 Fix Pack 11, this parameter specifies the execution driver framework for the following container jobs:
    - Pod Manager (Podman) container jobs if LSF is configured to run Podman container jobs instead of Docker container jobs. The user keyword must be set to `default` for Podman jobs, and the starter and controller file permissions must be set to 0755. The monitor file is not required for podman jobs, but if it is used, the monitor file permission must also be set to 0755. Because Podman uses the Docker execution driver to run Podman container jobs, **podman** and **docker** command options are not compatible, and execution driver permissions are not the same, you cannot use LSF to run Docker container jobs if you are using LSF to run Podman container jobs.
    - Enroot container jobs. The starter file permission must be set to 0755. The monitor and controller files are ignored. This parameter is optional for Enroot container jobs, and is `context[user(default)] starter[`*`/path/to/serverdir/`*`enroot-starter.py]` by default.
- GPU_REQ has the following changes in LSF Version 10.1 Fix Pack 11:
    - The new gvendor keyword in the GPU requirements strings enables LSF to allocate GPUs with the specified vendor type. Specify `gvendor=nvidia` to request Nvidia GPUs and `gvendor=amd` to request AMD GPUs.
    - The nvlink=yes keyword in the GPU requirements string is deprecated. Replace `nvlink=yes` in the GPU requirements string with `glink=nvlink` instead.
    - The new glink keyword in the GPU requirements strings specifies the connections among GPUs. Specify `glink=nvlink` for the NVLink connection for Nvidia GPUs or `glink=xgmi` for the xGMI connection for AMD GPUs. Do not use glink with the nvlink keyword, which is now deprecated.
- In LSF Version 10.1 Fix Pack 12, the following parameters are deprecated and will be removed in a future version:
    - CHUNK_JOB_SIZE
    - NETWORK_REQ
- In LSF Version 10.1 Fix Pack 13, the supported Podman version is 3.3.1. For both the lsb.applications and lsb.queues files, the CONTAINER parameter now supports podman configuration, and requires the EXEC_DRIVER parameter (with mandatory `[user(default)]` configuration for the Podman jobs to start, and `controller[]` configuration (such as `controller[/path/to/serverdir/docker-control.py]`) for Podman to work.
- In LSF Version 10.1 Fix Pack 13, the CONTAINER parameter supports Apptainer container jobs to run in the application profile by using the apptainer[] keyword. Apptainer is the rebranded product name for Singularity.

lsb.globalpolicies (New)

The lsb.globalpolicies file defines global policies for multiple clusters. This file is optional, but is required to enable global fair share scheduling. It is installed by default in LSB_CONFDIR/cluster_name/configdir.

Global fair share policies are defined in the GlobalFairshare section.

After you change the lsb.globalpolicies file, use the **badmin gpdrestart** command to reconfigure the global policy daemon (**gpolicyd**).

- Limits section: In LSF Version 10.1 Fix Pack 10, you can now specify global resource allocations in the Limits sections.
  Specify global resource allocations the same way you would specify local resource allocation limits in the Limit sections of the lsb.resources file, using the following parameters: APPS, ELIGIBLE_PEND_JOBS, INELIGIBLE, JOBS, JOBS_PER_SCHED_CYCLE, LIC_PROJECTS, MEM, NAME, PROJECTS, QUEUES, RESOURCE, SLOTS, SWP, TMP, and USERS.

  When using the LSF multicluster capability, global resource allocation limits apply to all clusters.

- Per-consumer limits: LSF Version 10.1 Fix Pack 11, you can now specify global per-consumer resource allocations in the Limits sections by specifying the following new parameters: PER_APP, PER_LIC_PROJECT, PER_PROJECT, PER_QUEUE, and PER_USER.
- Resource section: In LSF 10.1 Fix Pack 13, this new section defines global resources that are shared between all clusters.
- ResourceMap section: In LSF 10.1 Fix Pack 13, this new section defines the mapping between shared resources and their sharing clusters.
- DistributePolicy section: In LSF 10.1 Fix Pack 13, this new section defines the distribution policies for global resources and global limits.
- ReservationUsage section: In LSF 10.1 Fix Pack 13, this new section defines the method of reserving global resources.

lsb.hosts

- In the `ComputeUnit` section, the MEMBER parameter now allows use colons (`:`) to specify a range of numbers when you specify condensed notation for host names. Colons are used the same as hyphens (`-`) are currently used to specify ranges and can be used interchangeably in condensed notation. You can also use leading zeros to specify host names.
- #INCLUDE: In LSF Version 10.1 Fix Pack 5 and Fix Pack 6, you can use this new directive to insert the contents of a specified file into this configuration file.
- In the Host section, the HOST_NAME parameter now supports condensed notation for host names. Use square brackets (`[]`) to enclose the multiple numbers, and use a hyphen (`-`) or colon (`:`) to specify a range of numbers. Use a comma (`,`) to separate multiple ranges of numbers or to separate individual numbers. You can also use leading zeros to specify host names.

  Use multiple sets of square brackets (with the supported special characters) to define multiple sets of non-negative integers anywhere in the host name. For example, `hostA[1,3]B[1-3]` includes `hostA1B1`, `hostA1B2`, `hostA1B3`, `hostA3B1`, `hostA3B2`, and `hostA3B3`.

- In the HostGroup section, the GROUP_MEMBER parameter now allows colons (`:`) to specify a range of numbers when you specify condensed notation for host names. Colons are used the same as hyphens (`-`) are currently used to specify ranges and can be used interchangeably in condensed notation. You can also use leading zeros to specify host names.

Use multiple sets of square brackets (with the supported special characters) to define multiple sets of non-negative integers anywhere in the host name. For example, `hostA[1,3]B[1-3]` includes `hostA1B1`, `hostA1B2`, `hostA1B3`, `hostA3B1`, `hostA3B2`, and `hostA3B3`.

- LSF 10.1 Fix Pack 13, in the HostGroup section, the GROUP_MEMBER parameter now supports preferred host groups to indicate your preference for dispatching a job to a certain host group. LSF supports using a plus sign (+) and a positive number, after the names of host groups that you would prefer to use. A higher number indicates a higher preference (for example, (**`hostA groupB+2 hostC+1`**) indicates that groupB is the most preferred and hostA is the least preferred.

lsb.params

- JOB_ARRAY_EVENTS_COMBINE: This parameter is introduced to improve performance with large array jobs. When enabled (set to `Y`), more events for operations on array jobs or elements are generated, specific to the array job. For job arrays with large array sizes, mbatchd daemon performance is improved because operations are used by events specific to the array jobs. When this parameter is enabled, the following events are modified in the lsb.events file to accommodate array index ranges: `JOB_CLEAN`, `JOB_MODIFY2`, `JOB_MOVE`, `JOB_SIGNAL`, `JOB_STATUS`, and `JOB_SWITCH`. In the lsb.acct file, the `JOB_FINISH` event is modified. In the lsb.stream file, the `JOB_FINISH2` event is modified. In the lsb.status file, the JOB_STATUS2 event is modified.
- JOB_INFO_EVENT_DUMP_INTERVAL: Controls how frequently the job information events file is rewritten. If the dump interval is too frequent, it means a greater load on I/O operations. If the dump interval is too infrequent, events replay will take longer to finish. The parameter specifies the interval in number of minutes. Specify any positive integer between 1 and 2147483646. The default interval is 15 minutes.
- JOB_INFO_MEMORY_CACHE_SIZE: Configures how much memory to use for the job information cache. The job information cache can reduce the load on the work directory file server by caching job information such as the job's environment variables, command-line and **eexec** data in memory, in a compressed format. Set this parameter to the amount of memory in MB allocated to cached job information. The cache is enabled by default and the default cache size is 1024 MB (1 GB). The cache can be disabled by setting this parameter to 0. The minimum recommended cache size is 500 MB. Valid values are greater than or equal to zero, and less than MAX_INT. The value of the JOB_INFO_MEMORY_CACHE_SIZE parameter can be viewed with the command **bparams -a** or **bparams -l**. Real cache memory that is used can affect **mbatchd** fork performance.
- JOB_SWITCH2_EVENT: Obsolete in LSF 10.1. Replaced with the JOB_ARRAY_EVENTS_COMBINE parameter.
- RELAX_JOB_DISPATCH_ORDER: Allows LSF to deviate from standard job prioritization policies to improve cluster utilization by allowing multiple jobs with common resource requirements to run consecutively on the same allocation.
  By default, the same allocation can be reused for up to 30 minutes. You can also specify a custom allocation reuse time by specifying the `ALLOC_REUSE_DURATION` keyword with a maximum value and an optional minimum value.

- DIAGNOSE_LOGDIR: This parameter no longer requires ENABLE_DIAGNOSE to be enabled. The DIAGNOSE_LOGDIR parameter is also the default location for the snapshot of the scheduling job buckets (**badmin diagnose -c jobreq**) in addition to default location of the log file for query source information (**badmin diagnose -c query**). The ENABLE_DIAGNOSE parameter is required for the query source information log file to be saved to this location.
- CONDENSE_PENDING_REASONS: Previously, set to Y at time of installation for the HIGH_THROUGHPUT configuration template. If otherwise undefined, then N as default. For this release, it is removed from the HIGH_THROUGHPUT configuration template. Therefore, the default is always N.

For this release, when the condensed pending reason feature is enabled, the single key pending reason feature and the categorized pending reason feature will be overridden by the main reason, if there is one.

- MC_SORT_BY_SUBMIT_TIME - When set to `Y/y` allows forwarded jobs on the execution cluster to be sorted and run based on their original submission time (instead of their forwarded time). Available in the IBM Spectrum LSF multicluster capability only.
- PEND_REASON_MAX_JOBS: Obsolete in LSF 10.1.
- TRACK_ELIGIBLE_PENDINFO: Set to Y to enable LSF to determine whether a pending job is eligible for scheduling, and to use eligible pending time instead of total pending time to determine job priorities for automatic job priority escalation and absolute priority scheduling.
- ELIGIBLE_PENDINFO_SNAPSHOT_INTERVAL: Specifies the time interval, in minutes, for **mbschd** to dump eligible and ineligible pending information to disk. The eligible and ineligible pending information is saved when **mbatchd** or **mbschd** restarts. The default value is 5 minutes
- JOB_SCHEDULING_INTERVAL: Now specifies the minimal interval between subsequent job scheduling sessions. Specify in seconds, or include the keyword `ms` to specify in milliseconds. A value of 0 means no minimum interval between subsequent sessions. Previously, this parameter specified the amount of time that **mbschd** sleeps before the next scheduling session starts.
- ESTIMATOR_MAX_JOBS_PREDICTION: Specifies the number of pending jobs that the estimator predicts, which is 1000 by default.
- ESTIMATOR_MAX_TIME_PREDICTION: Specifies the amount of time into the future, in minutes, that a job is predicted to start before the estimator stops the current round of estimation. By default, the estimator stops after a job is predicted to start in one week (10080 minutes).
- ESTIMATOR_MAX_RUNTIME_PREDICTION: Specifies the amount of time that the estimator runs, up to the value of the ESTIMATOR_SIM_START_INTERVAL parameter. By default, the estimator stops after it runs for 30 minutes or the amount of time as specified by the ESTIMATOR_SIM_START_INTERVAL parameter, whichever is smaller.
- EVALUATE_JOB_DEPENDENCY_TIMEOUT: In LSF Version 10.1 Fix Pack 2, this new parameter sets the maximum amount of time, in seconds or milliseconds, that the **mbatchd** daemon takes to evaluate job dependencies in one scheduling cycle. This parameter limits the amount of time that **mbatchd** spends on evaluating job dependencies in a scheduling cycle, which limits the amount of time the job dependency evaluation blocks services. If the EVALUATE_JOB_DEPENDENCY parameter is also defined, the EVALUATE_JOB_DEPENDENCY_TIMEOUT parameter takes effect.
- EVALUATE_WAIT_CONDITION_TIMEOUT: In LSF Version 10.1 Fix Pack 2, this new parameter specifies a limit to the amount of time that the **mbatchd** daemon spends on evaluating the **bwait** wait conditions in a scheduling session.
- DEFAULT_BWAIT_TIMEOUT: In LSF Version 10.1 Fix Pack 2, this new parameter specifies the default timeout interval to evaluate the wait conditions in a scheduling session, in minutes.
- MAX_PEND_JOBS: In LSF Version 10.1 Fix Pack 3, this parameter has been changed to specify pending "jobs" instead of pending "job slots" as in previous versions of LSF.
- MAX_PEND_SLOTS: In LSF Version 10.1 Fix Pack 3, this new parameter has been added to specify "job slots" and replaces the previous role of MAX_PEND_JOBS.
- FWD_JOB_FACTOR: In LSF Version 10.1 Fix Pack 4, this new parameter defines the forwarded job slots factor, which accounts for forwarded jobs when making the user priority calculation for the fair share policies.
- JOB_GROUP_CLEAN: In LSF Version 10.1 Fix Pack 4, a new option "all" has been provided for JOB_GROUP_CLEAN, to delete empty implicit job groups automatically even if they have limits.
- EADMIN_TRIGGER_INTERVAL: In LSF Version 10.1 Fix Pack 5 and Fix Pack 6, this parameter is introduced to invoke an eadmin script at a set interval even if there is no job exception. The default is 0, which disables this feature and only triggers an eadmin script when there is a job exception.
- PERSIST_LIVE_CONFIG: In LSF Version 10.1 Fix Pack 5 and Fix Pack 6, this parameter is introduced to allow update of configuration files for a live reconfiguration. This allows for job submission during a policy update or cluster restart. The default is Y, which enables this feature.

If PERSIST_LIVE_CONFIG=Y LSF will persist all live config request so that they take effect after mbatchd restart.

If PERSIST_LIVE_CONFIG=N LSF will not persist live config request and they will not take effect after mbatchd restart.

- ALLOCATION_PLANNER: In LSF Version 10.1 Fix Pack 5 and Fix Pack 6, this parameter is introduced to enable the plan-based scheduling and reservation feature.
- ESTIMATED_RUNTIME: In LSF Version 10.1 Fix Pack 5 and Fix Pack 6, this parameter is introduced to configure cluster-wide estimated runtimes for jobs and is meant to replace the existing RUNTIME parameter from the lsb.applications file. Can also be configured at the application level (lsb.applications file) and queue level (lsb.queues file).
- PLAN: In LSF Version 10.1 Fix Pack 5 and Fix Pack 6, this parameter is introduced for use when plan-based scheduling is enabled with the parameter ALLOCATION_PLANNER=Y. This parameter controls whether or not jobs are candidates for plan-based scheduling. Can also be configured at the application level (lsb.applications file) and queue level (lsb.queues file).
- DEFAULT_PROJECT: In LSF Version 10.1 Fix Pack 6, the project name can now be up to 511 characters long (previously, this limit was 59 characters).
- EGROUP_UPDATE_INTERVAL: In LSF Version 10.1 Fix Pack 7, this parameter also controls the time interval for which dynamic host group information is updated automatically, in addition to dynamic user group information. You can also specify the time interval in minutes by using the m keyword after the time interval.
- GPU_RUN_TIME_FACTOR: In LSF Version 10.1 Fix Pack 7, this new parameter defines the GPU run time factor, which accounts for the total GPU run time of a user's running GPU jobs.
- GPU_RUN_TIME_FACTOR: In LSF Version 10.1 Fix Pack 7, this new parameter defines the GPU run time factor, which accounts for the total GPU run time of a user's running GPU jobs when calculating fair share scheduling policy.
- ENABLE_GPU_HIST_RUN_TIME: In LSF Version 10.1 Fix Pack 7, this new parameter enables the use of historical GPU run time in the calculation of fair share scheduling policy.
- KILL_JOBS_OVER_RUNLIMIT: In LSF Version 10.1 Fix Pack 7, this new parameter enables the **mbatchd** daemon to kill jobs that are running over the defined RUNLIMIT value for a long period of time.
- CSM_VALID_SMT: In LSF Version 10.1 Fix Pack 8, this new parameter defines a space-delimited list of valid SMT mode values for CSM jobs. The first value in the list is the default value for CSM jobs if the SMT mode is not specified at the queue or job level.
- SECURE_INFODIR_USER_ACCESS: In LSF Version 10.1 Fix Pack 9, this parameter now has the new keyword G to provide full granularity over what information the **bhist** and **bacct** commands display for jobs for other users. Enable this feature by defining SECURE_INFODIR_USER_ACCESS=G.
- SECURE_JOB_INFO_LEVEL: In LSF Version 10.1 Fix Pack 9, this parameter now has an additional information level 5 to display summary information for the jobs that belong to other users. Enable this information level by defining SECURE_JOB_INFO_LEVEL=5.
- DOCKER_IMAGE_AFFINITY: In LSF Version 10.1 Fix Pack 9, this new parameter enables LSF to give preference for execution hosts that already have the requested Docker image when submitting or scheduling Docker jobs.
- GPU_REQ_MERGE: In LSF Version 10.1 Fix Pack 9, this new parameter enables all individual options in the GPU requirement string to be merged separately. Any specified options override the any options that are specified at the lower levels of precedence. If an individual option is not specified, but is explicitly specified at a lower level, then the highest level for which the option is specified takes precedence.
- SIMPLIFIED_GUARANTEE: In LSF Version 10.1 Fix Pack 10, this new parameter enables simplified scheduling algorithms for package and slot pools that are used by jobs with guarantee policies.
- ATTR_CREATE_USERS: In LSF Version 10.1 Fix Pack 10, this new parameter specifies the users who can create host attributes for attribute affinity scheduling.

- ATTR_MAX_NUM: In LSF Version 10.1 Fix Pack 10, this new parameter specifies the maximum number of host attributes that can exist simultaneously in the cluster.
- ATTR_TTL: In LSF Version 10.1 Fix Pack 10, this new parameter specifies the time-to-live (TTL) for newly-created host attributes.
- SAME_JOB_AFFINITY: In LSF Version 10.1 Fix Pack 10, this new parameter enables users to specify affinity preferences for jobs to run on the same host or compute unit as another job. That is, users can use the samehost and samecu keywords with the **bsub -jobaff** command option.
- GLOBAL_LIMITS: In LSF Version 10.1 Fix Pack 10, this new parameter enables global limit scheduling, which allows you to specify global resource allocation limits in the lsb.globalpolicies file. When using the LSF multicluster capability, global resource allocation limits apply to all clusters.
- RELAX_JOB_DISPATCH_ORDER: In LSF Version 10.1 Fix Pack 10, this parameter now has the SHARE[] keyword to relax additional constraints on the pending jobs that can reuse resource allocations for finished jobs.
- JOB_DISPATCH_PACK_SIZE: In LSF Version 10.1 Fix Pack 10, this new parameter specifies the maximum number of job decisions that can accumulate before LSF publishes the decisions in a decision package before the end of the job scheduling cycle.
- JOB_SCHEDULING_INTERVAL: In LSF Version 10.1 Fix Pack 10, you can now specify a maximum time for the job scheduling cycle. **mbschd** skips job scheduling if the scheduling cycle exceeds this time. To specify the maximum time, add a second number, in seconds.
- RESCHED_UPON_CSM_SETUP_ERROR: In LSF Version 10.1 Fix Pack 10, this new parameter enables LSF to reschedule IBM CSM jobs that are stage-in or non-transfer jobs that fail during CSM setup if they fail with the specified CSM API error codes.
- DEFAULT_RC_ACCOUNT_PER_PROJECT: In LSF Version 10.1 Fix Pack 11, this new parameter enables LSF to set the project name as the default account name on hosts that are borrowed through LSF resource connector.
- ENABLE_RC_ACCOUNT_REQUEST_BY_USER: In LSF Version 10.1 Fix Pack 11, this new parameter enables users to assign a specific account name at the job level on hosts that are borrowed through LSF resource connector. This allows users to use the `bsub -rcacct "rc_account_name"` command option to assign an account name.
- In LSF Version 10.1 Fix Pack 12, the following parameters are deprecated and will be removed in a future version:
  - CHUNK_JOB_DURATION
  - ENABLE_DEFAULT_EGO_SLA
  - MAX_PROTOCOL_INSTANCES
  - NETWORK_REQ
  - SIMPLIFIED_GUARANTEE: Now fixed to `Y`.
  - STRIPING_WITH_MINIMUM_NETWORK
- FAIRSHARE_JOB_COUNT parameter: In LSF 10.1 Fix Pack 13, this new parameter enables LSF to use the number of jobs instead of job slots in the fair share scheduling algorithm.
- JOB_GROUP_IDLE_TTL parameter: In LSF 10.1 Fix Pack 13, this new parameter defines the job group's time-to-live (TTL) when all jobs leave the job group.

lsb.queues

- RELAX_JOB_DISPATCH_ORDER: Allows LSF to deviate from standard job prioritization policies to improve cluster utilization by allowing multiple jobs with common resource requirements to run consecutively on the same allocation.
  By default, the same allocation can be reused for up to 30 minutes. You can also specify a custom allocation reuse time by specifying the `ALLOC_REUSE_DURATION` keyword with a maximum value and an optional minimum value.

- **ELIGIBLE_PEND_TIME_LIMIT** specifies the eligible pending time limit for jobs in the queue.

- **PEND_TIME_LIMIT** specifies the pending time limit for jobs in the queue.
- FWD_JOB_FACTOR: In LSF Version 10.1 Fix Pack 4, this new parameter defines the forwarded job slots factor, which accounts for forwarded jobs when making the user priority calculation for the fair share policies.
- #INCLUDE: In LSF Version 10.1 Fix Pack 5 and Fix Pack 6, you can use this new directive to insert the contents of a specified file into this configuration file.
- FWD_USERS: In LSF Version 10.1 Fix Pack 5 and Fix Pack 6, you can use this new parameter to specify a list of users or user groups that can forward jobs to remote clusters when using the LSF multicluster capability. This allows you to prevent jobs from certain users or user groups from being forwarded to an execution cluster, and to set limits on the submission cluster.
- EXTENDABLE_RUNLIMIT: In LSF Version 10.1 Fix Pack 5 and Fix Pack 6, you can use this new parameter to enable jobs to continue running past the original run limit if resources are not needed by other jobs.
- ESTIMATED_RUNTIME: In LSF Version 10.1 Fix Pack 5 and Fix Pack 6, this parameter is introduced to configure estimated runtimes for jobs in a queue, and is meant to replace the existing RUNTIME parameter from the lsb.applications file. Can also be configured at the application level (lsb.applications file) and cluster level (lsb.params file).
- PLAN: In LSF Version 10.1 Fix Pack 5 and Fix Pack 6, this parameter is introduced for use when plan-based scheduling is enabled with the parameter ALLOCATION_PLANNER=Y. This parameter controls whether or not jobs are candidates for plan-based scheduling. Can also be configured at the application level (lsb.applications file), cluster level (lsb.params file), and job level.
- CSM_REQ: In LSF Version 10.1 Fix Pack 5 and Fix Pack 6, this parameter specifies the required values for the IBM Cluster Systems Manager (CSM) **bsub** job submission command options. These settings override job level CSM options and append system level allocation flags to the job level allocation flags.
    - In LSF Version 10.1 Fix Pack 8, you can now use the `smt` keyword to specify the SMT mode.
- GPU_REQ: In LSF Version 10.1 Fix Pack 6, this new parameter specifies the GPU requirements for the queue.
- GPU_RUN_TIME_FACTOR: In LSF Version 10.1 Fix Pack 7, this new parameter defines the GPU run time factor, which accounts for the total GPU run time of a user's running GPU jobs when calculating fair share scheduling policy.
- ENABLE_GPU_HIST_RUN_TIME: In LSF Version 10.1 Fix Pack 7, this new parameter enables the use of historical GPU run time in the calculation of fair share scheduling policy.
- In LSF Version 10.1 Fix Pack 8, the maximum integer that you can specify for the following resource limits is increased from 32 bit (2^{31}) to 64 bit (2^{63}):
    - memory limit (MEMLIMIT parameter)
    - swap limit (SWAPLIMIT parameter)
    - core file size limit (CORELIMIT parameter)
    - stack limit (STACKLIMIT parameter)
    - data segment size limit (DATALIMIT parameter)
    - file size limit (FILELIMIT parameter)
- GPU_REQ: In LSF Version 10.1 Fix Pack 8, GPU_REQ has the following changes:
    - You can now specify `aff=no` in the GPU requirements to relax GPU affinity while maintaining CPU affinity. By default, `aff=no` is set to maintain strict GPU-CPU affinity binding.
    - You can now specify `mps=per_socket` in the GPU requirements to enable LSF to start one MPS daemon per socket per job on each GPU host.
    - You can now specify `mps=per_gpu` in the GPU requirements to enable LSF to start one MPS daemon per GPU per job on each GPU host.
- RES_REQ: In LSF Version 10.1 Fix Pack 9, you can now specify the resource reservation method (by task, by job, or by host) in the rusage string by using the /task, /job, or /host keyword after the numeric value in the rusage string. You can only specify resource reservation methods for consumable resources.
  In addition, you can now enable LSF to stripe tasks of a parallel job across the free resources of the candidate hosts by using the stripe keyword in the span string of the resource requirements.

- GPU_REQ: In LSF Version 10.1 Fix Pack 9, you can now specify `mps=yes,share`, `mps=per_socket,share`, and `mps=per_gpu,share` in the GPU requirements to enable LSF to share the MPS daemon on the host, socket, or GPU for jobs that are submitted by the same user with the same resource requirements.
  That is, you can now add "`,share`" to the mps value to enable MPS daemon sharing for the host, socket, or GPU.

  In addition, you can now assign the number of GPUs per task or per host by specifying `num=`*`number`*`/task` or `num=`*`number`*`/host`. By default, the number of GPUs is still assigned per host.

- RUN_WINDOW: In LSF Version 10.1 Fix Pack 9, this parameter now allows you to specify supported time zones when specifying the time window. You can specify multiple time windows, but all time window entries must be consistent in whether they set the time zones. That is, either all entries must set a time zone, or all entries must not set a time zone.
- DISPATCH_WINDOW: In LSF Version 10.1 Fix Pack 9, this parameter now allows you to specify supported time zones when specifying the time window. You can specify multiple time windows, but all time window entries must be consistent in whether they set the time zones. That is, either all entries must set a time zone, or all entries must not set a time zone.
- CONTAINER: In LSF Version 10.1 Fix Pack 9, this new parameter enables container jobs to run in the queue. The usage of this parameter is the same as the CONTAINER parameter in the lsb.applications file.
- EXEC_DRIVER: In LSF Version 10.1 Fix Pack 9, this new parameter specifies the execution driver framework for the queue. The usage of this parameter is the same as the EXEC_DRIVER parameter in the lsb.applications file.
- DOCKER_IMAGE_AFFINITY: In LSF Version 10.1 Fix Pack 9, this new parameter enables LSF to give preference for execution hosts that already have the requested Docker image when submitting or scheduling Docker jobs.
- GPU_REQ has the following changes in LSF Version 10.1 Fix Pack 10:
  - You can now add the "`,nocvd`" keyword to the existing `mps` value in the GPU resource requirements string to disable the CUDA_VISIBLE_DEVICES environment variable for MPS jobs.
  - You can now specify `block=yes` in the GPU resource requirements string to enable block distribution of allocated GPUs.
  - You can now specify `gpack=yes` in the GPU resource requirements string to enable pack scheduling for shared mode GPU jobs.
- USE_PAM_CREDS has the following changes in LSF Version 10.1 Fix Pack 10:
  - You can now specify the `session` keyword to enable LSF to open a PAM session when submitting jobs to Linux hosts using PAM..
  - You can now specify the `limits` keyword to apply the limits that are specified in the PAM configuration file to a queue. This is functionally identical to enabling USE_PAM_CREDS=y except that you can define `limits` together with the `session` keyword.
- MC_FORWARD_DELAY: In LSF Version 10.1 Fix Pack 10, this new parameter specifies the job forwarding behavior and the amount of time after job submission and scheduling for LSF revert to the default job forwarding behavior.
- RELAX_JOB_DISPATCH_ORDER: In LSF Version 10.1 Fix Pack 10, this parameter now has the SHARE[] keyword to relax additional constraints on the pending jobs that can reuse resource allocations for finished jobs.
- MAX_SBD_CONNS: In LSF Version 10.1 Fix Pack 10, the default value of this parameter is changed to 2 * numOfHosts + 300.
- DISPATCH_BY_QUEUE: In LSF Version 10.1 Fix Pack 10, this parameter is obsolete and replaced by the JOB_DISPATCH_PACK_SIZE parameter in the lsb.params file.
- CONTAINER: LSF Version 10.1 Fix Pack 11 enables the following new containers with this parameter:
  - Pod Manager (Podman) container jobs run in the application profile by using the docker[] keyword to run Podman jobs with the Docker execution driver. The options[] keyword specifies

Podman job run options for the **podman run** command, which are passed to the job container. Because Podman uses the Docker execution driver to run Podman container jobs, **podman** and **docker** command options are not compatible, and execution driver permissions are not the same, you cannot use LSF to run Docker container jobs if you are using LSF to run Podman container jobs.
    - Enroot container jobs run in the application profile by using the new enroot[] keyword to run Enroot jobs with the Enroot execution driver. The options[] keyword specifies Enroot job run options for the **enroot start** command, which are passed to the job container.
- EXEC_DRIVER: In LSF Version 10.1 Fix Pack 11, this parameter specifies the execution driver framework for the following container jobs:
    - Pod Manager (Podman) container jobs if LSF is configured to run Podman container jobs instead of Docker container jobs. The user keyword must be set to `default` for Podman jobs, and the starter and controller file permissions must be set to 0755. The monitor file is not required for podman jobs, but if it is used, the monitor file permission must also be set to 0755. Because Podman uses the Docker execution driver to run Podman container jobs, **podman** and **docker** command options are not compatible, and execution driver permissions are not the same, you cannot use LSF to run Docker container jobs if you are using LSF to run Podman container jobs.
    - Enroot container jobs. The starter file permission must be set to 0755. The context, monitor, and controller settings are ignored. This parameter is optional for Enroot container jobs, and is `context[user(default)] starter[`*`/path/to/serverdir`*`/enroot-starter.py]` by default.
- GPU_REQ has the following changes in LSF Version 10.1 Fix Pack 11:
    - The new gvendor keyword in the GPU requirements strings enables LSF to allocate GPUs with the specified vendor type. Specify `gvendor=nvidia` to request Nvidia GPUs and `gvendor=amd` to request AMD GPUs.
    - The nvlink=yes keyword in the GPU requirements string is deprecated. Replace `nvlink=yes` in the GPU requirements string with `glink=nvlink` instead.
    - The new glink keyword in the GPU requirements strings specifies the connections among GPUs. Specify `glink=nvlink` for the NVLink connection for Nvidia GPUs or `glink=xgmi` for the xGMI connection for AMD GPUs. Do not use glink with the nvlink keyword, which is now deprecated.
- In LSF Version 10.1 Fix Pack 12, the following parameters are deprecated and will be removed in a future version:
    - CHUNK_JOB_SIZE
    - HOSTS (`allremote` and `all@`*`cluster_name`* keywords only)
    - MAX_PROTOCOL_INSTANCES
    - MAX_SLOTS_IN_POOL
    - NETWORK_REQ
    - SLOT_POOL
    - SLOT_SHARE
    - STRIPING_WITH_MINIMUM_NETWORK
    - USE_PRIORITY_IN_POOL
- In LSF Version 10.1 Fix Pack 13, the new IMPT_JOBLIMIT and IMPT_TASKLIMIT parameters allow you to specify how many MultiCluster jobs or tasks, from remote clusters, that can be configured at the receive-jobs queue.
- In LSF Version 10.1 Fix Pack 13, the supported Podman version is 3.3.1. For both the lsb.applications and lsb.queues files, the CONTAINER parameter now supports podman configuration, and requires the EXEC_DRIVER parameter (with mandatory `[user(default)]` configuration for the Podman jobs to start, and `controller[]` configuration (such as `controller[/path/to/serverdir/docker-control.py]`) for Podman to work.
- In LSF Version 10.1 Fix Pack 13, the CONTAINER parameter supports Apptainer container jobs to run in the queue. Apptainer is the rebranded product name for Singularity. The usage of this parameter is the same as the CONTAINER parameter in the lsb.applications file.

lsb.reasons

lsb.reasons allows for individual configuration of pending reason messages. Administrators can make messages clear and can inform users on which action they can take to allow the job to run. Messages can be customized for one or more pending reasons and the priority that is given to particular resources.

This file is optional. It is installed by default in config/lsbatch/*<cluster_name>*/configdir/lsb.reasons.

After you change the lsb.reasons file, run **badmin reconfig**.

- #INCLUDE: In LSF Version 10.1 Fix Pack 5 and Fix Pack 6, you can use this new directive to insert the contents of a specified file into this configuration file.

lsb.resources

- LOAN_POLICIES: In LSF Version 10.1 Fix Pack 1, you can now enable queues to ignore the RETAIN and DURATION loan policies when LSF determines whether jobs in those queues can borrow unused guaranteed resources. To enable the queue to ignore the RETAIN and DURATION loan policies, specify an exclamation point (!) before the queue name in the LOAN_POLICIES parameter definition.
- #INCLUDE: In LSF Version 10.1 Fix Pack 5 and Fix Pack 6, you can use this new directive to insert the contents of a specified file into this configuration file.
- JOBS_PER_SCHED_CYCLE: In LSF Version 10.1 Fix Pack 5 and Fix Pack 6, you can use this new parameter to set limits on the maximum number of jobs that are dispatched in a scheduling cycle for users, user groups, and queues. You can only set job dispatch limits if the limit consumer types are USERS, PER_USER, QUEUES, or PER_QUEUE.
- PER_PROJECT: In LSF Version 10.1 Fix Pack 6, each project name can now be up to 511 characters long (previously, this limit was 59 characters).
- PROJECTS: In LSF Version 10.1 Fix Pack 6, each project name can now be up to 511 characters long (previously, this limit was 59 characters).
- In LSF Version 10.1 Fix Pack 7, the JOBS_PER_SCHED_CYCLE parameter is renamed to ELIGIBLE_PEND_JOBS. The old JOBS_PER_SCHED_CYCLE parameter is still kept for backwards compatibility.
- APPS: In LSF Version 10.1 Fix Pack 9, this new parameter specifies one or more application profiles on which limits are enforced. Limits are enforced on all application profiles listed.
- PER_APP: In LSF Version 10.1 Fix Pack 9, this new parameter specifies one or more application profiles on which limits are enforced. Limits are enforced on each application profile listed.
- LOAN_POLICIES: In LSF Version 10.1 Fix Pack 10, the RETAIN keyword is now deprecated and replaced with IDLE_BUFFER.
- HostExport section: In LSF Version 10.1 Fix Pack 12, the HostExport section is deprecated and will be removed in a future version.
- SharedResourceExport section: In LSF Version 10.1 Fix Pack 12, the SharedResourceExport section is deprecated and will be removed in a future version.

lsb.users

- FS_POLICY in the UserGroup section: This new parameter enables global fair share policy for the defined user group. FS_POLICY specifies which global fair share policy the share account will participate into.
- MAX_PEND_JOBS in the User section: In LSF Version 10.1 Fix Pack 3, this parameter has been changed to specify pending "jobs" instead of pending "job slots" as in previous versions of LSF.
- MAX_PEND_SLOTS in the User section: In LSF Version 10.1 Fix Pack 3, this new parameter has been added to specify "job slots" and replaces the previous role of MAX_PEND_JOBS.

- #INCLUDE: In LSF Version 10.1 Fix Pack 5 and Fix Pack 6, you can use this new directive to insert the contents of a specified file into this configuration file.
- PRIORITY in the User and UserGroup sections: In LSF Version 10.1 Fix Pack 5 and Fix Pack 6, this new parameter specifies a priority that is used as a factor when calculating the job priority for absolute priority scheduling (APS).

lsf.cluster.*cluster_name*

- HOSTNAME in the Host section now supports condensed notation for host names.
  Use square brackets (`[]`) to enclose the multiple numbers, and use a hyphen (`-`) or a colon (`:`) to specify a range of numbers. Use a comma (`,`) to separate multiple ranges of numbers or to separate individual numbers. You can also use leading zeros to specify host names.

  Use multiple sets of square brackets (with the supported special characters) to define multiple sets of non-negative integers anywhere in the host name. For example, `hostA[1,3]B[1-3]` includes `hostA1B1`, `hostA1B2`, `hostA1B3`, `hostA3B1`, `hostA3B2`, and `hostA3B3`.

lsf.conf

- The LSB_BJOBS_FORMAT parameter now has the following fields:
  - `effective_plimit`, `plimit_remain`, `effective_eplimit`, and `eplimit_remain` to display the job's pending time limit, remaining pending time, eligible pending time limit, and remaining eligible pending. You can use the -p option to show only information for pending jobs.
  - `"pend_reason"` shows the pending reason field of a job. If a job has no pending reason (for example, the job is running), then the `pend_reason` field of the job is NULL and shows a hyphen (`-`).
- LSB_BCONF_PROJECT_LIMITS: By default, LSF does not allow the **bconf** command to create project-based limits because LSF schedules jobs faster if the cluster has no project-based limits. If you need to use the **bconf** command to dynamically create project-based limits when the cluster is running, set the LSB_BCONF_PROJECT_LIMITS parameter to `Y`.
- LSB_BJOBS_PENDREASON_LEVEL: This new parameter sets the default behavior when a user enters the command **bjobs -p**, without specifying a level of 0 to 3. For upgraded clusters, if the LSB_BJOBS_PENDREASON_LEVEL parameter is not configured, the level for the **bjobs -p** command is 0 by default. For new clusters, the LSB_BJOBS_PENDREASON_LEVEL parameter is set to 1 in the installation template, showing the single key reason by default.
- LSB_BMGROUP_ALLREMOTE_EXPAND: When set to `N` or `n` in the IBM® Spectrum LSF multicluster capability resource leasing model, the **bmgroup** command displays leased-in hosts with a single keyword `allremote` instead of being displayed as a list. Otherwise, a list of leased-in hosts is displayed in the `HOSTS` column in the form *`host_name@cluster_name`* by default.
- LSB_DEBUG_GPD: This new parameter sets the debugging log class for **gpolicyd**. Only messages that belong to the specified log class are recorded.
- LSB_EXCLUDE_HOST_PERIOD: Specifies the amount of time, in the number of **mbatchd** sleep time units (MBD_SLEEP_TIME), that a host remains excluded from a job. When this time expires, the hosts are no longer excluded and the job can run on the host again.
  This parameter does not apply to the IBM Spectrum LSF multicluster capability job lease model.

- LSB_ESUB_SINGLE_QUOTE: When set to `Y` or `y`, the values in the LSB_SUB_PROJECT_NAME parameter that are written to the $LSB_SUB_PARM_FILE file for **esub** processes are enclosed in single quotation marks (`'`). The shell is prevented from processing certain meta characters in parameter values such as `$`.
  Otherwise, the LSB_SUB_PROJECT_NAME parameter values written to the $LSB_SUB_PARM_FILE file for **esub** processes are enclosed in double quotation marks (`"`) by default.

- LSB_GSLA_PREFER_ADRSV_HOST: When set to `Y` or `y`, the guaranteed SLA first tries to reserve hosts without advanced reservation. The LSB_GSLA_PREFER_ADRSV_HOST ensures that advance reservation does not interfere with guaranteed SLA job scheduling.
- LSB_GPD_CLUSTER: This new parameter defines the names of clusters whose master hosts start **gpolicyd** for global fair share policy among multiple clusters. The LSB_GPD_CLUSTER parameter must be configured for every cluster that participates in global fair share.
- LSB_GPD_PORT: This new parameter defines the TCP service port for communication with **gpolicyd**. The LSB_GPD_PORT parameter must be configured for every cluster that participates in global fair share.
- LSB_MBD_MAX_SIG_COUNT: Obsolete in LSF 10.1.
- LSB_SUPPRESS_CUSTOM_REASONS: This new parameter allows individual users to disable display of customized pending reasons for the new Single key reason feature (**bjobs -p1**) and Categorized Pending Reasons feature (**bjobs -p2** and **bjobs -p3**).
  By default, the value of the LSB_SUPPRESS_CUSTOM_REASONS parameter is set to N. This parameter applies to all **bjobs -p** levels except -p0. The command **bjobs -p0**, is used to display pending reasons in the style previous to version 10.1, without using the single key reason or the categorized pending reason features.

- LSB_TERMINAL_SERVICE_PORT: Specifies the terminal service port number for Remote Desktop Protocol (RDP). This port is used in **tssub** jobs.
- LSB_TIME_GPD: This new parameter sets a timing level for checking how long **gpolicyd** routines run. Time usage is logged in milliseconds.
- LSF_AUTH: You can now specify `LSF_AUTH=none` to disable authentication. Use the `LSF_AUTH=none` parameter only for performance benchmarking.
- LSF_CONNECTION_CHANGE: Windows only. When set to `Y` or `y`, enables **lsreghost** to register with LSF servers whenever it detects a change in the total number of connections (IP addresses) that are associated with the local host. This parameter is only valid if registration handling is enabled for LSF hosts (that is, `LSF_REG_FLOAT_HOSTS=Y` is set in the lsf.conf file on the LSF server).
- LSF_CRAY_RUR_ACCOUNTING: For LSF on Cray. Specify `N` to disable RUR job accounting if RUR is not enabled in your Cray environment, or to increase performance. Default value is `Y` (enabled).
- LSF_CRAY_RUR_DIR: Location of the Cray RUR data files, which is a shared file system that is accessible from any potential first execution host. Default value is LSF_SHARED_DIR/*<cluster_name>*/craylinux/*<cray_machine_name>*/rur.
- LSF_CRAY_RUR_PROLOG_PATH: File path to the RUR prolog script file. Default value is /opt/cray/rur/default/bin/rur_prologue.py.
- LSF_CRAY_RUR_EPILOG_PATH: File path to the RUR epilog script file. Default value is /opt/cray/rur/default/bin/rur_epilogue.py.
- LSF_DISCARD_LOG: Specifies the behavior of the **mbatchd** and **mbschd** logging threads if the logging queue is full.
  If set to `Y`, the logging thread discards all new messages at a level lower than LOG_WARNING when the logging queue is full. LSF logs a summary of the discarded messages later.

  If set to `N`, LSF automatically extends the size of the logging queue if the logging queue is full.

- LSF_LOG_QUEUE_SIZE: Specifies the maximum number of entries in the logging queues that the **mbatchd** and **mbschd** logging threads use before the logging queue is full.
- LSF_LOG_THREAD: If set to N, **mbatchd** and **mbschd** does not create dedicated threads to write messages to the log files.
- LSF_PLATFORM_COMPATIBILITY: Allows for compatibility with an earlier version of the *IBM Platform* name after LSF 10.1. Set it to `y`|`Y` in lsf.conf to enable **lsid** and the LSF command **-V** to display "IBM Platform LSF" instead of "IBM Spectrum LSF". The LSF_PLATFORM_COMPATIBILITY parameter solves compatibility issues between LSF 10.1 and older versions of IBM Platform Process Manager.
- LSF_REG_FLOAT_HOSTS: When set to `Y` or `y`, enables registration handling for LSF hosts so that LSF servers can resolve these hosts without requiring the use of DNS servers.

- LSF_REG_HOST_INTERVAL: Windows only. Specifies the interval, in minutes, in which **lsreghost** sends more registration messages to LSF servers. This parameter is only valid if registration handling is enabled for LSF hosts (that is, `LSF_REG_FLOAT_HOSTS=Y` is set in the lsf.conf file on the LSF server).
- LSF_REPLACE_PIM_WITH_LINUX_CGROUP: Minimizes the impact of PIM daemon processing load for parallel jobs. PIM collects job processes, the relationship among all processes, the memory usage of each process, and the CPU time of each process periodically. Those actions can influence the execution of parallel jobs (so-called *OS jitter*). To minimize OS jitter, you can configure the LSF cgroup feature. This parameter is only supported on Linux. The parameter is ignored on other operating systems. The LSF cgroup feature does not support PAM jobs, so you cannot disable PIM if you run PAM jobs.
- In LSF Version 10.1 Fix Pack 2, the LSB_BJOBS_FORMAT parameter now has the following fields:
  - `jobindex` shows the job array index.
  - `estimated_run_time` shows estimated run time of the job.
  - `ru_utime` and `ru_stime` show the user time used and the system time used from the resource usage information for the job.
  - `nthreads` shows the number of threads that the job used
  - `hrusage` shows the per-host resource usage information.
  - `plimit` and `eplimit` show the pending time limit and eligible time limit.
  - `licproject` shows the license project information.
  - `srcjobid`, `dstjobid`, and `source_cluster` show the submission cluster job ID, execution cluster job ID, and the name of the submission cluster when using the LSF multicluster capability.
- LSF_INTELLIGENT_CPU_BIND: In LSF Version 10.1 Fix Pack 2, this new parameter enables LSF to bind a defined set of LSF daemons to CPUs.
- LSB_BWAIT_REREG_INTERVAL: In LSF Version 10.1 Fix Pack 2, this new parameter specifies the default time interval to reregister the wait condition from the **bwait** command to the **mbatchd** daemon, in minutes.
- LSF_HOST_CACHE_NTTL: In LSF Version 10.1 Fix Pack 2, the default value of this parameter is increased from 20s to 60s, which is the maximum valid value.
- LSB_QUERY_PORT: In LSF Version 10.1 Fix Pack 2, the value of this parameter is now set to `6891` at the time of installation, which enables the multithread **mbatchd** job query daemon and specifies the port number that the **mbatchd** daemon uses for LSF query requests.
- LSB_QUERY_ENH: In LSF Version 10.1 Fix Pack 2, the value of this parameter is now set to `Y` at the time of installation, which extends multithreaded query support to batch query requests (in addition to **bjobs** query requests).
- LSF_DCGM_PORT: In LSF Version 10.1 Fix Pack 2, this new parameter enables the NVIDIA Data Center GPU Manager (DCGM) features and specifies the port number that LSF uses to communicate with the DCGM daemon.
- LSF_ENABLE_TMP_UNIT: In LSF Version 10.1 Fix Pack 2, this new parameter allows units that are defined by the LSF_UNIT_FOR_LIMITS parameter to also apply cluster-wide to the tmp resource.
- LSB_RC_MAX_INSTANCES_PER_TEMPLATE: In LSF Version 10.1 Fix Pack 2, this new parameter for the LSF resource connector specifies the maximum number of resource instances that can be launched for any template for any resource provider in the cluster. The default value is 50.
- LSB_BHOSTS_FORMAT: In LSF Version 10.1 Fix Pack 2, this new parameter customizes specific fields that the **bhosts** command displays.
- LSB_BQUEUES_FORMAT: In LSF Version 10.1 Fix Pack 2, this new parameter customizes specific fields that the **bqueues** command displays.
- LSB_HMS_TIME_FORMAT: In LSF Version 10.1 Fix Pack 2, this new parameter displays times from the customized **bjobs -o** command output in `hh:mm:ss` format. This parameter setting only applies to **bjobs -o** or **bjobs -o -json** command output.
- LSB_PROFILE_MBD: In LSF Version 10.1 Fix Pack 3, this new parameter configures the **mbatchd** daemon profiler to track the time that **mbatchd** spends on key functions.
- LSB_PROFILE_SCH: In LSF Version 10.1 Fix Pack 3, this new parameter configures the **mbschd** daemon profiler to track the time that **mbschd** spends on key functions.

- In LSF Version 10.1 Fix Pack 3, the LSB_BJOBS_FORMAT option now has the following fields:
  - `rsvid` shows the reservation ID, if the job is associated with an advance reservation.
- LSF_LSLOAD_FORMAT: In LSF Version 10.1 Fix Pack 3, this new parameter customizes specific fields that the **lsload** command displays.
- LSB_GPU_NEW_SYNTAX: In LSF Version 10.1 Fix Pack 3, this new parameter enables the **bsub -gpu** option to submit jobs that require GPU resources.
- LSF_ENABLE_BEAT_SERVICE: In LSF Version 10.1 Fix Pack 4, this new parameter enables the lsfbeat tool which integrates energy accounting into LSF. IBM Spectrum LSF Explorer uses Elasticsearch in the collection of the energy data of each host using Beats. With this tool enabled, LSF can query the data from IBM Spectrum LSF Explorer and the **bjobs** and **bhosts** display the job level or host level energy to users.
- LSF_QUERY_ES_SERVERS: In LSF Version 10.1 Fix Pack 4, this new parameter specifies LSF Explorer servers to retrieve log records. Use this parameter to enable the supported commands (as defined by the LSF_QUERY_ES_FUNCTIONS parameter) to use LSF Explorer to get log records instead of parsing the log files to get data.
- LSF_QUERY_ES_FUNCTIONS: In LSF Version 10.1 Fix Pack 4, this new parameter specifies the commands and functions that use LSF Explorer to retrieve job records.
- LSF_LSLOAD_FORMAT: In LSF Version 10.1 Fix Pack 4, this parameter now has the following fields:
  - `gpu_status*` shows the status of the GPU (`ok`, `error`, or `warning`). If more than 1 GPU is reported, an index is appended to the resource name, starting at 0. For example, `gpu_status0` and `gpu_status1`.
  - `gpu_error*` shows the detailed error or warning message if the `gpu_status*` field is not `ok`. If more than 1 GPU is reported, an index is appended to the resource name, starting at 0. For example, `gpu_status0` and `gpu_status1`.
- LSB_GPU_AUTOBOOST: In LSF Version 10.1 Fix Pack 4, this parameter is now obsolete because LSF synchronizes the GPU auto-boost to resolve any issues that previously required disabling the auto-boost.
- LSF_HWLOC_DYNAMIC: In LSF Version 10.1 Fix Pack 4, this new parameter enables LSF to dynamically load the hardware locality (**hwloc**) library from system library paths whenever it is needed. If LSF fails to load the library, LSF defaults to using the **hwloc** functions in the static library.
- LSB_ESWITCH_METHOD: In LSF Version 10.1 Fix Pack 4, this new parameter specifies a mandatory **eswitch** executable file that applies to all job switch requests.
- LSF_MC_FORWARD_FAIRSHARE_CHARGE_DURATION: In LSF Version 10.1 Fix Pack 4, this new parameter specifies the duration of time after which LSF removes the forwarded jobs from the user priority calculation for fair share scheduling. This parameter is used if global fair share scheduling is enabled for the LSF multicluster capability job forwarding model.
- LSB_START_EBROKERD: In LSF Version 10.1 Fix Pack 4, this new parameter enables the **mbatchd** daemon to start the **ebrokerd** daemon whenever **mbatchd** starts up, is reconfigured, or when it detects that the old **ebrokerd** daemon exits. This is required to use advance reservation prescripts and post-scripts. The **ebrokerd** daemon also starts automatically if the LSF resource connector is configured and in use.
- LSF_MQ_BROKER_HOSTS: In LSF Version 10.1 Fix Pack 4, this new parameter for LSF resource connector enables support for the **bhosts -rc** and **bhosts -rconly** command options to get LSF resource connector provider host information.
- MQTT_BROKER_HOST: In LSF Version 10.1 Fix Pack 4, new parameter for LSF resource connector. If you do not use the MQTT message broker daemon (**mosquitto**) that is provided with LSF, specifies the host name that **mosquitto** runs on. The MQTT message broker receives provider host information from **ebrokerd** and publishes that information for the **bhosts -rc** and **bhosts -rconly** command options to display.
- LSF_MQ_BROKER_PORT: In LSF Version 10.1 Fix Pack 4, new parameter for LSF resource connector. If you do not use the MQTT message broker daemon (**mosquitto**) that is provided with LSF, specifies an optional TCP port for the MQTT message broker daemon (**mosquitto**). The MQTT message broker

receives provider host information from **ebrokerd** and publishes that information for the **bhosts -rc** and **bhosts -rconly** command options to display.

- EBROKERD_HOST_CLEAN_DELAY: In LSF Version 10.1 Fix Pack 4, this new parameter for LSF resource connector specifies the delay, in minutes, after which the **ebrokerd** daemon removes information about relinquished or reclaimed hosts. This parameter allows the **bhosts -rc** and **bhosts -rconly** command options to get LSF resource connector provider host information for some time after they are deprovisioned.
- LSF_UGROUP_TRANSFER: In LSF Version 10.1 Fix Pack 5, this new parameter transfers secondary user group IDs from the submission host to the execution host for job execution, thereby overcoming an NFS limitation of 16 user groups.
- LSF_UDP_PORT_RANGE: In LSF Version 10.1 Fix Pack 5 and Fix Pack 6, this new parameter defines the UDP port range to be used by the LSF daemons. If defined, the UDP socket for the LSF daemons binds to one port in the specified range.
- LSF Version 10.1 Fix Pack 5 and Fix Pack 6, now have the following parameters for running jobs with IBM Cluster Systems Manager (CSM):
  - LSB_JSM_DEFAULT specifies the default value for the **bsub -jsm** option for CSM jobs.
  - LSB_STAGE_IN_EXEC specifies the stage in script for direct data staging (for example, IBM CAST burst buffer).
  - LSB_STAGE_OUT_EXEC specifies the stage out script for direct data staging.
  - LSB_STAGE_STORAGE specifies the resource name to report available storage space for direct data staging.
  - LSB_STAGE_TRANSFER_RATE specifies the estimated data transfer rate for the burst buffer. LSF uses this value to calculate the predicted duration for data stage in.
  - LSB_STEP_CGROUP_DEFAULT specifies the default value for the **bsub -step_cgroup** option for CSM jobs.
- In LSF Version 10.1 Fix Pack 6, the LSB_BJOBS_FORMAT parameter now has the following fields:
  - `gpfsio` shows job usage (I/O) data on IBM Spectrum Scale if IBM Spectrum Scale I/O accounting with IBM Spectrum LSF Explorer is enabled by setting `LSF_QUERY_ES_FUNCTIONS="gpfsio"` or `"all"` in the lsf.conf file.
- LSF_QUERY_ES_FUNCTIONS: In LSF Version 10.1 Fix Pack 6, this parameter now allows you to specify the `gpfsio` function, which enables IBM Spectrum Scale I/O accounting with IBM Spectrum LSF Explorer.
- LSF_GPU_AUTOCONFIG: In LSF Version 10.1 Fix Pack 6, this new parameter controls whether LSF enables use of GPU resources automatically.
- LSB_GPU_NEW_SYNTAX: In LSF Version 10.1 Fix Pack 6, this parameter now has `extend` as a new keyword. If LSB_GPU_NEW_SYNTAX=extend is set, you can specify the `gmem`, `gmodel`, `gtile`, and `nvlink` GPU requirements with the **bsub -gpu** option, the GPU_REQ parameter in the lsb.queues file, the GPU_REQ parameter in the lsb.applications file, or the LSB_GPU_REQ parameter in the lsf.conf file.
- LSB_GPU_REQ: In LSF Version 10.1 Fix Pack 6, this new parameter specifies the default GPU requirements for the cluster.
- LSB_GSLA_DISPLAY_ALLOC_HOSTS: In LSF Version 10.1 Fix Pack 7, this new parameter enables the **bsla** command to display information on guarantee hosts that are being used (allocated) from each guarantee pool for the guarantee SLA.
- LSB_BSUB_PARSE_SCRIPT: In LSF Version 10.1 Fix Pack 7, this new parameter enables the **bsub** command to load, parse, and run job scripts from the command line.
- LSF_LSHOSTS_FORMAT: In LSF Version 10.1 Fix Pack 7, this new parameter customizes specific fields that the **lshosts** command displays.
- LSB_STAGE_MAX_STAGE_IN: In LSF Version 10.1 Fix Pack 7, this new parameter specifies the maximum number of concurrent stage-in process that run on a host, which prevents LSF from launching too many stage-in processes to transfer files to the host.
- LSF_STAGE_STORAGE: In LSF Version 10.1 Fix Pack 7, this parameter now allows you to specify a resource that reports the total storage space in addition to a resource that reports the available storage space. This prevents LSF from assigning more storage than is available because the resource

information might be out of date. This can occur for direct data staging jobs where the job handles the file transfer instead of LSF because LSF cannot reliably predict the storage usage for these jobs.

- LSB_PLAN_KEEP_RESERVE: In LSF Version 10.1 Fix Pack 7, this new parameter enables LSF to keep the resource reservations for jobs with plans, even if the plan is no longer valid, until LSF creates new plans based on updated resource availability.
- In LSF Version 10.1 Fix Pack 7, the LSB_BJOBS_FORMAT parameter now has the following fields:
  - `nreq_slot` shows the calculated number of requested slots for jobs.
  - `gpu_num` shows the number of physical GPUs that the job is using.
  - `gpu_mode` shows the GPU compute mode that the job is using (`shared` or `exclusive_process`).
  - `gpu_alloc` shows the job-based GPU allocation information.
  - `j_exclusive` shows whether the job requested exclusive allocated GPUs (that is, if the GPUs cannot be shared with other jobs).
  - `kill_reason` shows the user-specified reason for killing the job.
- LSF_IMAGE_INFO_PUBLISH_INTERVAL: In LSF Version 10.1 Fix Pack 7, this new parameter specifies the interval for the **lim** process to fork a new process to collect host Docker container image information.
- LSF_IMAGE_INFO_EXPIRE_INTERVAL: In LSF Version 10.1 Fix Pack 7, this new parameter specifies how long the host image information is available in **mosquitto** before the information expires.
- LSF_EXT_SERVERDIR: In LSF Version 10.1 Fix Pack 7, this new parameter specifies a secure directory in which the eauth and esub.*application* binary files are located.
- LSF_ENV_OVERRIDE: In LSF Version 10.1 Fix Pack 7, this new parameter specifies whether environment variable values and the $LSF_ENVIDR/lsf.conf file parameters can override the parameter settings in the /etc/lsf.conf file.
- LSB_GPU_REQ has the following changes in LSF Version 10.1 Fix Pack 8:
  - You can now specify `aff=no` in the GPU requirements to relax GPU affinity while maintaining CPU affinity. By default, `aff=no` is set to maintain strict GPU-CPU affinity binding.
  - You can now specify `mps=per_socket` in the GPU requirements to enable LSF to start one MPS daemon per socket per job.
  - You can now specify `mps=per_gpu` in the GPU requirements to enable LSF to start one MPS daemon per GPU per job.
- LSF_AC_PNC_URL: In LSF Version 10.1 Fix Pack 8, this new parameter specifies the URL and listen port of the LSF Application Center Notifications server for sending notifications. If the listen port is not specified, the default port number is 80.
- LSB_RC_TEMPLATE_REQUEST_DELAY: In LSF Version 10.1 Fix Pack 8, this new parameter for LSF resource connector specifies the amount of time that LSF waits before repeating a request for a template, in minutes, if the **ebrokerd** daemon encountered certain provider errors.
- LSB_RC_MQTT_ERROR_LIMIT: In LSF Version 10.1 Fix Pack 8, this new parameter for LSF resource connector specifies the maximum number of API error messages that are stored in Mosquitto per host provider. This parameter specifies the maximum number of messages that the **badmin rc error** command displays for each host provider.
- LSB_GPU_REQ: In LSF Version 10.1 Fix Pack 9, you can now specify `mps=yes,share`, `mps=per_socket,share`, and `mps=per_gpu,share` in the GPU requirements to enable LSF to share the MPS daemon on the host, socket, or GPU for jobs that are submitted by the same user with the same resource requirements.
  That is, you can now add "`,share`" to the mps value to enable MPS daemon sharing for the host, socket, or GPU.

  In addition, you can now assign the number of GPUs per task or per host by specifying `num=`*`number`*`/task` or `num=`*`number`*`/host`. By default, the number of GPUs is still assigned per host.

- LSB_BUSERS_FORMAT: In LSF Version 10.1 Fix Pack 9, this new parameter customizes specific fields that the **busers** command displays.

- LSF_DATA_BSUB_CHKSUM: In LSF Version 10.1 Fix Pack 9, this new parameter enables the **bsub** and **bmod** commands to perform a full sanity check on the files and folders for jobs with a data requirement, and to generate the hash for each file and folder. If not specified, these operations occur at the transfer job.
- LSB_JOB_REPORT_MAIL: In LSF Version 10.1 Fix Pack 9, you can now specify `ERROR` for the **sbatchd** daemon to send mail only when the job exits (that is, when the job is under `Exit` status). This ensures than an email notification is only sent on a job error.
- In LSF Version 10.1 Fix Pack 9, the LSB_BJOBS_FORMAT parameter now has the `ask_hosts` field, which shows the list of requested hosts as specified by the **bsub -m** command option.
- LSB_MEMLIMIT_ENF_CONTROL: In LSF Version 10.1 Fix Pack 9, you can now exclude the swap threshold from memory limit enforcement and specify only the memory threshold. To exclude swap threshold, specify a value of 0 for the swap threshold.
- LSF_DATA_NO_SSH_CHK_HOSTS: In LSF Version 10.1 Fix Pack 9, this new parameter specifies a list of data hosts for which **ssh** is not needed. If the host specified with the data specification of a submitted job matches one of the hosts in this list, LSF assumes that the submission host can directly access the file within the data specification.
- In LSF Version 10.1 Fix Pack 10, the LSB_BJOBS_FORMAT parameter now has the following fields:
  - `suspend_reason` shows the user-specified reason for suspending (stopping) the job.
  - `resume_reason` shows the user-specified reason for resuming the job.
  - `kill_issue_host` shows the host that issued the job kill request.
  - `suspend_issue_host` shows the host that issued the job suspend (stop) request.
  - `resume_issue_host` shows the host that issued the job resume request.
- LSB_SUBK_SHOW_JOBID: In LSF Version 10.1 Fix Pack 10, this new parameter enables the **bsub -K** command option to display the job ID of a job after it is finished.
- LSB_GPU_REQ has the following changes in LSF Version 10.1 Fix Pack 10:
  - You can now add the "`,nocvd`" keyword to the existing `mps` value in the GPU resource requirements string to disable the CUDA_VISIBLE_DEVICES environment variable for MPS jobs.
  - You can now specify `block=yes` in the GPU resource requirements string to enable block distribution of allocated GPUs.
  - You can now specify `gpack=yes` in the GPU resource requirements string to enable pack scheduling for shared mode GPU jobs.
- LSB_NCPU_ENFORCE: In LSF Version 10.1 Fix Pack 10, this parameter is now enabled (that is, set to 1) at the time of installation for new LSF installations.
- LSB_MAX_JOB_DISPATCH_PER_SESSION: In LSF Version 10.1 Fix Pack 10, the default value of this parameter is changed to 15000.
- LSF_ACCEPT_NUMCLIENTS: In LSF Version 10.1 Fix Pack 10, this new parameter specifies the maximum number of new client connections to the **mbatchd** port that **mbatchd** accepts during each scheduling cycle. Previously, this value was fixed at 1.
- LSF_GPU_RESOURCE_IGNORE: In LSF Version 10.1 Fix Pack 10, this new parameter enables the **mbatchd** and **mbschd** daemons to ignore GPU resources. This means that the **lsload -s**, **lsload -l**, and **bhosts -l** commands, which display LSF resources, no longer display information about GPU resources. That is, these options do not display `gpu_<`*num*`>n` resources.
- LSF_ROOT_REX: In LSF Version 10.1 Fix Pack 10, this parameter is obsolete and no longer allows root execution privileges for jobs from local and remote hosts. Any actions that were performed as root must instead be performed as the LSF administrator.
- LSF_ROOT_USER: In LSF Version 10.1 Fix Pack 10, this new parameter enables the root user to perform actions as a valid user from the LSF command line.
  Important: Only enable `LSF_ROOT_USER=Y` as a temporary configuration setting. When you are done, you must disable this parameter to ensure that your cluster remains secure.
- LS_ROOT_USER: In LSF Version 10.1 Fix Pack 10, this new parameter enables the root user to run LSF License Scheduler commands (**bladmin**, **blkill**, **globauth**, and **taskman**) as a valid user from the LSF command line.

Important: Only enable `LS_ROOT_USER=Y` as a temporary configuration setting. When you are done, you must disable this parameter to ensure that your cluster remains secure.

- LSF_ADDON_HOSTS: In LSF Version 10.1 Fix Pack 10, this new parameter specifies a list of hosts for LSF Application Center, LSF RTM, or LSF Explorer that require root privileges to remotely execute commands.
- LSB_BJOBS_FORMAT: In LSF Version 10.1 Fix Pack 11, this parameter now allows you to display a unit prefix for the following resource fields: `mem`, `max_mem`, `avg_mem`, `memlimit`, `swap`, `swaplimit`, `corelimit`, `stacklimit`, and `hrusage` (for `hrusage`, the unit prefix is for `mem` and `swap` resources only).
  In addition, the default width for these resource fields except `hrusage` are increased from 10 to 15. That is, the following output fields now have a default width that is increased from 10 to 15:`mem`, `max_mem`, `avg_mem`, `memlimit`, `swap`, `swaplimit`, `corelimit`, and `stacklimit`.

- LSF_DISABLE_LSRUN: In LSF Version 10.1 Fix Pack 11, this parameter now also enables RES to refuse remote connections from the **lsmake** command, in addition to refusing remote connections from the **lsrun** and **lsgrun** commands.
- LSF_GPU_RESOURCE_IGNORE: In LSF Version 10.1 Fix Pack 11, if LSF_GPU_AUTOCONFIG is set to `Y` and LSB_GPU_NEW_SYNTAX is set to `Y` or `extend`, setting LSF_GPU_RESOURCE_IGNORE to `Y` also enables LSF to remove all built-in GPU resources (`gpu_<`*num*`>n`) from the *management* host LIM. LSF uses a different method for the *management* host LIM to collect GPU information in the cluster.
- LSB_GPU_NEW_SYNTAX: In LSF Version 10.1 Fix Pack 11, if GPU preemption is enabled (that is, PREEMPTABLE_RESOURCES parameter in the lsb.params file includes the `ngpus_physical` resource), setting LSB_GPU_NEW_SYNTAX=extend removes several restrictions to GPU preemption:
  - Non-GPU jobs can now preempt lower priority GPU jobs.
  - GPU jobs no longer have to be configured for automatic job migration and rerun to be preemptable by higher priority jobs. That is, the MIG parameter no longer has to be defined and the RERUNNABLE parameter no longer has to be set to `yes` in the lsb.queues or lsb.applications file. Ensure that you properly configure the MIG, RERUNNABLE, or REQUEUE parameters to ensure that GPU resources are properly released after the job is preempted.
  - GPU jobs no longer have to have either `mode=exclusive_process` or `j_exclusive=yes` set to be preempted by other GPU jobs. GPU jobs can also use `mode=shared` if the GPU is used by only one shared-mode job.
    Higher priority GPU jobs cannot preempt shared-mode GPU jobs if there are multiple jobs running on the GPU.

  Setting LSB_GPU_NEW_SYNTAX=Y enables GPU preemption with the previous restrictions (as introduced in LSF Version 10.1 Fix Pack 7).

- LSB_KRB_IMPERSONATE: In LSF Version 10.1 Fix Pack 11, this new parameter enables Kerberos user impersonation if external authentication is enabled (LSF_AUTH=eauth in the file lsf.conf).
- LSF_STRICT_CHECKING: In LSF Version 10.1 Fix Pack 11, you can now set this parameter to the `ENHANCED`, which enables LSF to also add a checksum to each authorization request in addition to enabling more strict checking of communications between LSF daemons and between LSF commands and daemons.
- LSF_AUTH_QUERY_COMMANDS: In LSF Version 10.1 Fix Pack 11, this new parameter enables query command authentication.
- LSF_MANAGE_MIG: In LSF Version 10.1 Fix Pack 11, this new parameter enables dynamic MIG scheduling.
- LSB_BHOSTS_FORMAT: In LSF Version 10.1 Fix Pack 11, this parameter has a new mig_alloc keyword to display the MIG allocation information in the **bhosts** customized output.
- LSB_BHOSTS_FORMAT: In LSF Version 10.1 Fix Pack 11, this parameter has a new mig_alloc keyword to display the MIG allocation information in the **bhosts** customized output.
- LSF_ENV_OVERRIDE: In LSF Version 10.1 Fix Pack 12, the default value of this parameter is changed to `N`.

- In LSF Version 10.1 Fix Pack 12, the following parameters are deprecated and will be removed in a future version:
  - LSB_CHUNK_RUSAGE
  - LSB_CPUSET_BESTCPUS
  - LSB_CPUSET_DISPLAY_CPULIST
  - LSB_GPU_NEW_SYNTAX: Now fixed to `extend`.
  - LSF_CPUSETLIB
  - LSF_GPU_AUTOCONFIG: Now fixed to `Y`.
  - LSF_GPU_RESOURCE_IGNORE: Now fixed to `Y`.
  - LSF_PAM_APPL_CHKPNT
  - LSF_PAM_CLEAN_JOB_DELAY
  - LSF_PAM_HOSTLIST_USE
  - LSF_PAM_PLUGINDIR
  - LSF_PAM_USE_ASH
  - LSF_PE_NETWORK_NUM
  - LSF_PE_NETWORK_UPDATE_INTERVAL
  - LSF_SHELL_AT_USERS
  - LSF_STRICT_RESREQ: Now fixed to `Y`.
  - LSF_TOPD_PORT
  - LSF_TOPD_TIMEOUT
- LSF_STRICT_CHECKING: In LSF Version 10.1 Fix Pack 12, the default value of this parameter is now `ENHANCED`.
- LSF_AUTH_QUERY_COMMANDS: In LSF Version 10.1 Fix Pack 12, the default value of this parameter is now `Y`.
- LSF_ADDON_HOSTS: In LSF Version 10.1 Fix Pack 12, this parameter is now required if you are running LSF Application Center, LSF Explorer, LSF Process Manager, or LSF RTM.
- The LSB_BQUEUES_FORMAT parameter now has the following fields for limits and resources:
  - `max_corelimit`, `max_cpulimit`, `default_cpulimit`, `max_datalimit`, `default_datalimit`, `max_filelimit`, `max_memlimit`, `default_memlimit`, `max_processlimit`, `max_runlimit`, `default_runlimit`, `max_stacklimit`, `max_swaplimit`, `max_tasklimit`, `min_tasklimit`, `default_tasklimit`, `max_threadlimit`, `default_threadlimit`, `res_req`, `hosts`.
  - The following resource limit fields show the same content as their corresponding maximum resource limit fields: `corelimit`, `cpulimit`, `datalimit`, `filelimit`, `memlimit`, `processlimit`, `runlimit`, `stacklimit`, `swaplimit`, `tasklimit`, `threadlimit`.
    For example, `corelimit` is the same as `max_corelimit`.
- LSB_BWAIT_IN_JOBS: In LSF 10.1 Fix Pack 13, this new parameter specifies whether LSF can use the **bwait** command within a job.
- LSF_GPU_AUTOCONFIG: In LSF 10.1 Fix Pack 13, the default value of this parameter changed from N to Y.
- LSB_GPU_NEW_SYNTAX: In LSF 10.1 Fix Pack 13, the default value of this parameter changed from not defined to `extend`.
- LSF_GPU_RESOURCE_IGNORE: In LSF 10.1 Fix Pack 13, the default value of this parameter changed from N to Y.
- Starting in LSF Version 10.1 Fix Pack 13, existing LSF commands that support host names as a parameter option now also accept host groups. For details on the affected commands, see **battr**, **bresume**, **brvs**, **lshosts**, and **lsload**.

lsf.datamanager

- CACHE_REFRESH_INTERVAL: In LSF Version 10.1 Fix Pack 9, this parameter is added to the Parameters section to limit the number of transfer jobs to the data manager by setting a refresh interval for the file cache. This is due to a change to the default behavior of jobs submitted to the data manager. The sanity check for the existence of files or folders and whether the user can access them, discovery of the size and modification of the files or folders, and generation of the hash from the **bsub** and **bmod** commands is moved to the transfer job. This equalizes the performance of submitting and modifying jobs with and without data requirements.

lsf.licensescheduler

- LM_RESERVATION in the Parameters and Feature section: In LSF Version 10.1 Fix Pack 3, this new parameter enables LSF License Scheduler to support the FlexNet Manager reservation keyword (RESERVE). LSF License Scheduler treats the RESERVE value in the FlexNet Manager license option file as OTHERS tokens instead of FREE tokens. The RESERVE value is now included in the OTHERS value in the **blstat** command output and is no longer included in the FREE value.
- In LSF 11.1, the following parameters are deprecated and removed:
  - ACCINUSE_INCLUDES_OWNERSHIP
  - FAST_DISPATCH: Now fixed to `Y`.
  - GROUP
  - LOCAL_TO
  - LS_ACTIVE_PERCENTAGE
- Clusters section: In LSF 10.1 Fix Pack 13, this new section specifies the licenses to share as global resources and the clusters that will share these licenses.

lsf.sudoers

- In LSF Version 10.1 Fix Pack 10, you must enable the setuid bit for the LSF administration commands to use the lsf.sudoers file. Run the **hostsetup --setuid** command option on the LSF master and candidate hosts. Since this allows LSF administration commands to run with root privileges, do not enable the setuid bit if you do not want these LSF commands to run with root privileges.
- LSF_EAUTH_OLDKEY: In LSF Version 10.1 Fix Pack 12, this parameter specifies the previous key that **eauth** used to encrypt and decrypt user authentication data after you specify a new **eauth** key. To use this parameter, you must also define the LSF_EAUTH_OLDKEY_EXPIRY parameter to specify an expiry date for the old key.
- LSF_EAUTH_OLDKEY_EXPIRY: In LSF Version 10.1 Fix Pack 12, this parameter specifies the expiry date for the previous **eauth** key (LSF_EAUTH_OLDKEY_EXPIRY parameter), after which the previous key no longer works and only the new LSF_EAUTH_KEY parameter works.

lsf.task

In LSF Version 10.1 Fix Pack 12, this file is deprecated and will be removed in a future version.

lsf.usermapping (New)

In LSF Version 10.1 Fix Pack 11, the lsf.usermapping file defines the user mapping policy for the new **bsubmit** command. The lsf.usermapping file allows you to map several job execution users and user groups to a single submission user or user group. Create the lsf.usermapping file in the $LSF_ENVDIR directory.

awsprov_templates.json

- interfaceType: In LSF Version 10.1 Fix Pack 10, this new parameter specifies whether the Elastic Fabric Adapter (EFA) network interface is attached to the instance.
- launchTemplateId: In LSF Version 10.1 Fix Pack Fix Pack 10, this new parameter specifies the AWS launch template.
- launchTemplateVersion: In Fix Pack 10, this new parameter specifies the specific version of AWS launch template to select.

azureccprov_templates.json (New)

In LSF Version 10.1 Fix Pack 9, the azureccprov_templates.json file defines the mapping between LSF resource demand requests and Microsoft Azure CycleCloud instances for LSF resource connector.

- imageName: In LSF Version 10.1 Fix Pack 10, this new parameter specifies that a cluster node uses a private Custom Azure image or a Marketplace image. You can find this ID for custom images in the Azure portal as the Resource ID for the image.
- interruptible: In LSF Version 10.1 Fix Pack 11, this new parameter enables the use of spot VMs.
- maxPrice: In LSF Version 10.1 Fix Pack 11, this new parameter defines the maximum allowed price of a spot VM before Azure reclaims the VM.

googleprov_config.json

- GCLOUD_REGION: In LSF Version 10.1 Fix Pack 12, this new parameter specifies the default region for LSF resource connector to use with the bulk API endpoint. The region that is defined in the googleprov_templates.json file overrides the region that is defined here.

googleprov_templates.json

- hostProject: In LSF Version 10.1 Fix Pack 10, this new parameter specifies the host project ID to generate the VPN and subnet values instead of the Google Cloud Project ID (that is, the GCLOUD_PROJECT_ID parameter in the googleprov_config.json file). If not specified, the LSF resource connector uses the Google Cloud Project ID to generate the VPN and subnet.
- launchTemplateId: In LSF Version 10.1 Fix Pack 12, this new parameter specifies the launch template ID. Specify this parameter and the zone parameter to enable launch instance templates.

hostProviders.json

- preProvPath: In LSF Version 10.1 Fix Pack 2, this new parameter specifies the absolute path file to the pre-provisioning script that the LSF resource connector runs after the instance is created and started successfully but before it is marked allocated to the LSF cluster.
- postProvPath: In LSF Version 10.1 Fix Pack 2, this new parameter specifies the absolute file path to the post-provisioning script that the LSF resource connector runs after the instance is terminated successfully but before it is removed from the LSF cluster.
- provTimeOut: In LSF Version 10.1 Fix Pack 2, this new parameter specifies the maximum amount of time, in minutes, for a pre- or post-provisioning script to run before it ends. Use this parameter to avoid the pre- or post-provisioning program from running for an unlimited time. The default value is 10 minutes. If set to 0, pre- and post-provisioning is disabled for the LSF resource connector .

ibmcloudgen2_config.json (New)

In LSF Version 10.1 Fix Pack 11, the ibmcloudgen2_config.json file manages remote administrative functions that the resource connector must perform on IBM Cloud Virtual Servers for Virtual Private Cloud Gen 2 (Cloud VPC Gen 2).

ibmcloudgen2_templates.json (New)

In LSF Version 10.1 Fix Pack 11, the ibmcloudgen2_templates.json file defines the mapping between LSF resource demand requests and Cloud VPC Gen 2 instances for LSF resource connector.

policy_config.json (New)

In LSF Version 10.1 Fix Pack 2, the policy_config.json file configures custom policies for resources providers for the LSF resource connector. The resource policy plug-in reads this file.

The default location for the file is *<LSF_TOP>*/conf/resource_connector/policy_config.json.

The policy_config.json file contains a JSON list of named policies. Each policy contains a name, a consumer, a maximum number of instances that can be launched for the consumer, and maximum number of instances that can be launched in a specified period.

Environment variables

- LSB_BMGROUP_ALLREMOTE_EXPAND: In the IBM Spectrum LSF multicluster capability resource leasing model, the **bmgroup** command now displays a list of leased-in hosts in the `HOSTS` column in the form *`host_name@cluster_name`* by default.
  If LSB_BMGROUP_ALLREMOTE_EXPAND=N is configured, leased-in hosts are represented by a single keyword `allremote` instead of being displayed as a list.

- The **epsub** environment variable LSB_SUB_JOB_ID indicates the ID of a submitted job that is assigned by LSF, as shown by **bjobs**. A value of -1 indicates that **mbatchd** rejected the job submission.
- The **epsub** environment variable LSB_SUB_JOB_QUEUE indicates the name of the final queue from which the submitted job is dispatched, which includes any queue modifications that are made by **esub**.
- The **epsub** environment variable LSB_SUB_JOB_ERR indicates the error number of a submitted job if the job submission failed, and is used by the **epsub** to determine the reason for job submission failure. If the job was submitted or modified successfully, the value of this environment variable is LSB_NO_ERROR (or 0)
- LSB_BHOSTS_FORMAT: In LSF Version 10.1 Fix Pack 2, this new environment variable customizes specific fields that the **bhosts** command displays.
- LSB_BQUEUES_FORMAT: In LSF Version 10.1 Fix Pack 2, this new environment variable customizes specific fields that the **bqueues** command displays.
- LSB_HMS_TIME_FORMAT: In LSF Version 10.1 Fix Pack 2, this new environment variable displays times from the customized **bjobs -o** command output in `hh:mm:ss` format. This environment variable setting only applies to **bjobs -o** or **bjobs -o -json** command output.
- NOCHECKVIEW_POSTEXEC: In LSF Version 10.1 Fix Pack 3, this environment variable for the LSF Integration for Rational ClearCase is obsolete because the daemon wrappers no longer run the **checkView** function to the check the ClearCase view, which means that this environment variable is no longer needed.
- LSB_DATA_PROVENANCE: In LSF Version 10.1 Fix Pack 4, this new environment variable enables data provenance tools for tracing data output files for jobs.
- LSB_DEFAULTPROJECT: In LSF Version 10.1 Fix Pack 6, the project name can now be up to 511 characters long (previously, this limit was 59 characters).

- LSB_PROJECT_NAME: In LSF Version 10.1 Fix Pack 6, the project name can now be up to 511 characters long (previously, this limit was 59 characters).
- LSB_DOCKER_IMAGE_AFFINITY: In LSF Version 10.1 Fix Pack 9, this new environment variable enables LSF to give preference for execution hosts that already have the requested Docker image when submitting or scheduling Docker jobs.
- LSB_BUSERS_FORMAT: In LSF Version 10.1 Fix Pack 9, this new environment customizes specific fields that the **busers** command displays.
- LSF_AC_JOB_NOTIFICATION: In LSF Version 10.1 Fix Pack 9, this new environment requests that the user be notified when the job reaches any of the specified states.

New and changed accounting and job event fields

The following job event fields are added or changed for LSF 10.1.

Note: Starting in Fix Pack 14, instead of reading this topic, refer to the What's new information for each fix pack (which contain changed and new details, including direct references to affected topics).

lsb.acct

- JOB_FINISH record: The **ineligiblePendTime(%d)** field is added. **ineligiblePendTime(%d)** is the time in seconds that the job is in the ineligible pending state.
- JOB_FINISH record: The **indexRangeCnt (%d)** field is added, for the number of element ranges that indicate successful signals. The **indexRangeStart1 (%d)** field is added, for the start of the first index range. The **indexRangeEnd1 (%d)** field is added, for the end of the first index range. The **indexRangeStep1 (%d)** field is added, for the step of the first index range.
- JOB_FINISH record: In LSF 10.1, Fx Pack 7, the **userGroup (%s)** field is added for the user group.
- JOB_FINISH record: In LSF 10.1, Fx Pack 11, the **schedulingOverhead (%f)** field is added for the scheduler overhead, in milliseconds.

lsb.events

- JOB_CLEAN record:
  - If JOB_ARRAY_EVENTS_COMBINE in lsb.params is enabled (set to `Y`), job IDs are recorded in a range format. Range fields consist of multiple segments that define groups of array indexes. The following new fields are added:
    - **indexRangeCnt (%d)**: The number of element ranges that indicate successful signals.
    - **indexRangeStart1 (%d)**: The start of the first index range.
    - **indexRangeEnd1 (%d)**: The end of the first index range.
    - **indexRangeStep1 (%d)**: The step of the first index range.
    - **indexRangeStartN (%d)**: The start of the last index range.
    - **indexRangeEndN (%d)**: The end of the last index range.
    - **indexRangeStepN (%d)**: The step of the last index range.
- JOB_MODIFY2 record:
  - The **pendTimeLimit(%d)** field is added. **pendTimeLimit(%d)** is the job-level pending time limit of the job, in seconds.
  - The **eligiblePendTimeLimit(%d)** field is added. **eligiblePendTimeLimit(%d)** is the job-level eligible pending time limit of the job, in seconds.

    - If JOB_ARRAY_EVENTS_COMBINE in lsb.params is enabled (set to `Y`), job IDs are recorded in a range format for successfully modified array jobs. Range fields consist of multiple segments that define groups of array indexes. The following new fields are added:
        - **`numRmtCtrlResult2 (%s):`** Number of remote job modification sessions that are generated by this modify request. JobIDs are recorded in an array index range format.
        - **`rmtCtrlResult2 (%s):`** Remote job modification records.
        - **`indexRangeCnt (%d)`**: The number of element ranges that indicate successful signals.
        - **`indexRangeStart1 (%d)`**: The start of the first index range.
        - **`indexRangeEnd1 (%d)`**: The end of the first index range.
        - **`indexRangeStep1 (%d)`**: The step of the first index range.
        - **`indexRangeStartN (%d)`**: The start of the last index range.
        - **`indexRangeEndN (%d)`**: The end of the last index range.
        - **`indexRangeStepN (%d)`**: The step of the last index range.
- JOB_MOVE record:
    - If JOB_ARRAY_EVENTS_COMBINE in lsb.params is enabled (set to `Y`), job IDs are recorded in a range format for successfully modified array jobs. Range fields consist of multiple segments that define groups of array indexes. The following new fields are added:
        - **`jobArrayIndex arrayIndex`**: The job array index.
        - **`numRmtCtrlResult2 (%d)`**: The number of records for remote job move handling.
        - **`rmtJobCtrlRecord2`**: Remote job move result.
- JOB_RESIZE_NOTIFY_START record:
    - **`GPU_ALLOC_COMPAT (%s)`**: In LSF 10.1, Fix Pack 13, this new string describes resized portions of the GPU allocation.
    - **`GPU_MEM_RSV (%s)`**: In LSF 10.1, Fix Pack 13, this new string describes GPU memory that is reserved by resized tasks on execution hosts.
- JOB_NEW record:
    - The **`pendTimeLimit(%d)`** field is added. **`pendTimeLimit(%d)`** is the job-level pending time limit of the job, in seconds.
    - The **`eligiblePendTimeLimit(%d)`** field is added. **`eligiblePendTimeLimit(%d)`** is the job-level eligible pending time limit of the job, in seconds.
- JOB_SIGNAL record:
    - If JOB_ARRAY_EVENTS_COMBINE in lsb.params is enabled (set to `Y`), job IDs are recorded in a range format for successfully modified array jobs. Range fields consist of multiple segments that define groups of array indexes. The following new fields are added:
        - **`indexRangeCnt (%d)`**: The number of element ranges that indicate successful signals.
        - **`indexRangeStart1 (%d)`**: The start of the first index range.
        - **`indexRangeEnd1 (%d)`**: The end of the first index range.
        - **`indexRangeStep1 (%d)`**: The step of the first index range.
        - **`indexRangeStartN (%d)`**: The start of the last index range.
        - **`indexRangeEndN (%d)`**: The end of the last index range.
        - **`indexRangeStepN (%d)`**: The step of the last index range.
- JOB_START record: The **`ineligiblePendTime(%d)`** field is added. **`ineligiblePendTime(%d)`** is the time in seconds that the job is in the ineligible pending state.
- JOB_STATUS record:
    - The **`ineligiblePendTime(%d)`** field is added. **`ineligiblePendTime(%d)`** is the time in seconds that the job is in the ineligible pending state.
    - If JOB_ARRAY_EVENTS_COMBINE in lsb.params is enabled (set to `Y`), job IDs are recorded in a range format for successfully modified array jobs. Range fields consist of multiple segments that define groups of array indexes. The following new fields are added:
        - **`indexRangeCnt (%d)`**: The number of element ranges that indicate successful signals.
        - **`indexRangeStart1 (%d)`**: The start of the first index range.
        - **`indexRangeEnd1 (%d)`**: The end of the first index range.
        - **`indexRangeStep1 (%d)`**: The step of the first index range.

        - **indexRangeStartN (%d)**: The start of the last index range.
        - **indexRangeEndN (%d)**: The end of the last index range.
        - **indexRangeStepN (%d)**: The step of the last index range.
- JOB_SWITCH record:
    - If JOB_ARRAY_EVENTS_COMBINE in lsb.params is enabled (set to Y), job IDs are recorded in a range format for successfully modified array jobs. Range fields consist of multiple segments that define groups of array indexes. The following new fields are added:
        - **idx (%d)**: Job array index, if it is -1, the **indexRangeCnt** will take effect
        - **indexRangeCnt (%d)**: The number of element ranges that indicate successfully signaled.
        - **indexRangeStart1 (%d)**: The start of the first index range.
        - **indexRangeEnd1 (%d)**: The end of the first index range.
        - **indexRangeStep1 (%d)**: The step of the first index range.
        - **indexRangeStartN (%d)**: The start of the last index range.
        - **indexRangeEndN (%d)**: The end of the last index range.
        - **indexRangeStepN (%d)**: The step of the last index range.
- HOST_CLOSURE_LOCK_ID_CTRL record: In LSF Version 10.1 Fix Pack 10, this new record stores batch server host closure lock IDs.
- ATTR_CREATE record: In LSF Version 10.1 Fix Pack 10, this new record is logged when creating host attributes for attribute affinity scheduling.
- ATTR_DELETE record: In LSF Version 10.1 Fix Pack 10, this new record is logged when deleting host attributes for attribute affinity scheduling.
- ATTR_DELETE record: In LSF Version 10.1 Fix Pack 10, this new record is a snapshot of information on a host attribute at the time of an event switch. Each record represent one host attribute.
- JOB_START record: In LSF 10.1, Fx Pack 11, the **SCHEDULING_OVERHEAD (%f)** field is added for the scheduler overhead, in milliseconds.

lsb.jobinfo.events

The job information event file stores the contents of the job information cache. It is located by default at $LSB_SHAREDIR/cluster_name/logdir/lsb.jobinfo.events. The location of the file can be changed with the parameter LSB_JOBINFO_DIR in lsf.conf. The job information cache can reduce the load on the work directory file server by caching job information such as the job's environment variables, command-line and **eexec** data in memory, in a compressed format. The job information cache can improve job submission and job dispatch performance, especially when the work directory's shared file system is slow or at the limits of its performance.

The amount of memory that is dedicated to the cache is controlled by the lsb.params parameter JOB_INFO_MEMORY_CACHE_SIZE.

As jobs are cleaned from the system, the job information event file needs to be periodically rewritten to discard the unneeded data. By default, the job information event file is rewritten every 15 minutes. This interval can be changed with the lsb.params parameter JOB_INFO_EVENT_DUMP_INTERVAL.

lsb.status

- JOB_STATUS2 record: The **indexRangeCnt (%d)** field is added, for the number of element ranges that indicate successful signals. The **indexRangeStart (%d)** field is added, for the start of the index range. The **indexRangeEnd (%d)** field is added, for the end of the index range. The **indexRangeStep (%d)** field is added, for the step of the index range.

- JOB_FINISH2 record: The **indexRangeCnt (%d)** field is added, for the number of element ranges that indicate successful signals. The **indexRangeStart (%d)** field is added, for the start of the index range. The **indexRangeEnd (%d)** field is added, for the end of the index range. The **indexRangeStep (%d)** field is added, for the step of the index range.
- EVENT_HOST_CTRL record: This record is logged if the host is closed with a lock ID for the first time. This record is only captured if HOST_CTRL is configured in the ALLOW_EVENT_TYPE parameter in the lsb.params file.