

---

# Platform LSF Configuration Reference

Platform LSF  
Version 8.0  
January 2011



Copyright

© 1994-2011 Platform Computing Corporation.

Although the information in this document has been carefully reviewed, Platform Computing Corporation ("Platform") does not warrant it to be free of errors or omissions. Platform reserves the right to make corrections, updates, revisions or changes to the information in this document.

UNLESS OTHERWISE EXPRESSLY STATED BY PLATFORM, THE PROGRAM DESCRIBED IN THIS DOCUMENT IS PROVIDED "AS IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT WILL PLATFORM COMPUTING BE LIABLE TO ANYONE FOR SPECIAL, COLLATERAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, INCLUDING WITHOUT LIMITATION ANY LOST PROFITS, DATA, OR SAVINGS, ARISING OUT OF THE USE OF OR INABILITY TO USE THIS PROGRAM.

We'd like to hear from you

You can help us make this document better by telling us what you think of the content, organization, and usefulness of the information. If you find an error, or just want to make a suggestion for improving this document, please address your comments to [doc@platform.com](mailto:doc@platform.com).

Your comments should pertain only to Platform documentation. For product support, contact [support@platform.com](mailto:support@platform.com).

Document redistribution and translation

This document is protected by copyright and you may not redistribute or translate it into another language, in part or in whole.

Internal redistribution

You may only redistribute this document internally within your organization (for example, on an intranet) provided that you continue to check the Platform Web site for updates and update your version of the documentation. You may not make it available to your organization over the Internet.

Trademarks

LSF is a registered trademark of Platform Computing Corporation in the United States and in other jurisdictions.

ACCELERATING INTELLIGENCE, PLATFORM COMPUTING, PLATFORM SYMPHONY, PLATFORM JOB SCHEDULER, PLATFORM ISF, PLATFORM ENTERPRISE GRID ORCHESTRATOR, PLATFORM EGO, and the PLATFORM and PLATFORM LSF logos are trademarks of Platform Computing Corporation in the United States and in other jurisdictions.

UNIX is a registered trademark of The Open Group in the United States and in other jurisdictions.

Linux is the registered trademark of Linus Torvalds in the U.S. and other countries.

Microsoft is either a registered trademark or a trademark of Microsoft Corporation in the United States and/or other countries.

Windows is a registered trademark of Microsoft Corporation in the United States and other countries.

Intel, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Other products or services mentioned in this document are identified by the trademarks or service marks of their respective owners.

Third-party license agreements

<http://www.platform.com/Company/third.part.license.htm>

---

# Contents

---

## Part I: Configuration Files ... 5

bld.license.acct .....	7
cshrc.lsf and profile.lsf .....	9
hosts .....	18
install.config .....	21
lim.acct .....	30
lsb.acct .....	31
lsb.applications .....	41
lsb.events .....	74
lsb.hosts .....	111
lsb.modules .....	127
lsb.params .....	132
lsb.queues .....	188
lsb.resources .....	239
lsb.serviceclasses .....	272
lsb.users .....	283
lsf.acct .....	291
lsf.cluster .....	294
lsf.cluster_name.license.acct .....	314
lsf.conf .....	316
lsf.licensescheduler .....	436
lsf.shared .....	465
lsf.sudoers .....	471
lsf.task .....	477
setup.config .....	480
slave.config .....	483

---

## Part II: Environment Variables ... 489

Environment variables set for job execution .....	491
Environment variables for resize notification command .....	492
Environment variables for session scheduler (ssched) .....	493
Environment variable reference .....	495





# Configuration Files

---

**Important:**

Specify any domain names in all uppercase letters in all configuration files.



# bld.license.acct

The `bld.license.acct` file is the license and accounting file for Platform License Scheduler.

## bld.license.acct structure

The license accounting log file is an ASCII file with one record per line. The fields of a record are separated by blanks. Platform License Scheduler adds a new record to the file every hour.

## File properties

### Location

The default location of this file is `LSF_SHAREDIR/db`. Use `LSF_LICENSE_ACCT_PATH` in `lsf.conf` to specify another location.

### Owner

The primary Platform License Scheduler admin is the owner of this file.

### Permissions

```
-rw-r--r--
```

## Records and fields

The fields in order of occurrence are as follows:

### timestamp (%d)

Time stamp of the logged event (in seconds since the epoch).

### type (%s)

The LSF product type. For Platform License Scheduler, this is `LICENSE_SCHEDULER`.

### version (%s)

The version of the Platform License Scheduler product.

### value (%d)

The total number of tokens that Platform License Scheduler is using.

### status (%s)

The results of the license usage check. The valid values are as follows:

- OK  
Token usage is less than the currently licensed amount
- OVERUSE  
Token usage is more than the currently licensed amount

### hash (%s)

Line encryption used to authenticate the record.

## Example record format

```
1107961731 LICENSE_SCHEDULER 7.0 0 OK 335a33c2bd9c9428140a61e57bd06da02b623a42
1107961792 LICENSE_SCHEDULER 7.0 2 OK 58e45b891f371811edfcceb6f5270059a74ee31a
1126639979 LICENSE_SCHEDULER 7.0 0 5 OK b3efd43ee28346f2d125b445fd16aa96875da35
1126640028 LICENSE_SCHEDULER 7.0 6 5 OVERUSE 2865775920372225fa7f8ed4b9a8eb2b15
```

## See also

- `LSF_LOGDIR` in `lsf.conf`
- `LSF_LICENSE_ACCT_PATH` in `lsf.conf`
- `lsf.cluster_name.license.acct`

# cshrc.lsf and profile.lsf

## About cshrc.lsf and profile.lsf

The user environment shell files `cshrc.lsf` and `profile.lsf` set the LSF operating environment on an LSF host. They define machine-dependent paths to LSF commands and libraries as environment variables:

- `cshrc.lsf` sets the C shell (`csh` or `tcsh`) user environment for LSF commands and libraries
- `profile.lsf` sets and exports the Bourne shell/Korn shell (`sh`, `ksh`, or `bash`) user environment for LSF commands and libraries

---

### Tip:

LSF Administrators should make sure that `cshrc.lsf` or `profile.lsf` are available for users to set the LSF environment variables correctly for the host type running LSF.

---

## Location

`cshrc.lsf` and `profile.lsf` are created by `lsfinstall` during installation. After installation, they are located in `LSF_CONFDIR` (`LSF_TOP/conf/`).

## Format

`cshrc.lsf` and `profile.lsf` are conventional UNIX shell scripts:

- `cshrc.lsf` runs under `/bin/csh`
- `profile.lsf` runs under `/bin/sh`

## What cshrc.lsf and profile.lsf do

`cshrc.lsf` and `profile.lsf` determine the binary type (`BINARY_TYPE`) of the host and set environment variables for the paths to the following machine-dependent LSF directories, according to the LSF version (`LSF_VERSION`) and the location of the top-level installation directory (`LSF_TOP`) defined at installation:

- `LSF_BINDIR`
- `LSF_SERVERDIR`
- `LSF_LIBDIR`
- `XLSF_UIDDIR`

`cshrc.lsf` and `profile.lsf` also set the following user environment variables:

- `LSF_ENVDIR`
- `LD_LIBRARY_PATH`
- `PATH` to include the paths to:
  - `LSF_BINDIR`
  - `LSF_SERVERDIR`
- `MANPATH` to include the path to the LSF man pages

## If Platform EGO is enabled

If Platform EGO is enabled in the LSF cluster (`LSF_ENABLE_EGO=Y` and `LSF_EGO_ENVDIR` are defined in `lsf.conf`), `cshrc.lsf` and `profile.lsf` set the following environment variables.

- `EGO_BINDIR`
- `EGO_CONFDIR`
- `EGO_ESRVDIR`
- `EGO_LIBDIR`
- `EGO_LOCAL_CONFDIR`
- `EGO_SERVERDIR`
- `EGO_TOP`

See the *Platform EGO Reference* for more information about these variables.

## Setting the LSF environment with cshrc.lsf and profile.lsf

Before using LSF, you must set the LSF execution environment.

After logging on to an LSF host, use one of the following shell environment files to set your LSF environment:

- For example, in `csh` or `tcsh`:  
`source /usr/lsf/lsf_8/conf/cshrc.lsf`
- For example, in `sh`, `ksh`, or `bash`:  
`./usr/lsf/lsf_8/conf/profile.lsf`

## Making your cluster available to users with cshrc.lsf and profile.lsf

To set the LSF user environment, run one of the following two shell files:

- `LSF_CONFDIR/cshrc.lsf` (for `csh`, `tcsh`)
- `LSF_CONFDIR/profile.lsf` (for `sh`, `ksh`, or `bash`)

---

### Tip:

LSF administrators should make sure all LSF users include one of these files at the end of their own `.cshrc` or `.profile` file, or run one of these two files before using LSF.

---

## For csh or tcsh

Add `cshrc.lsf` to the end of the `.cshrc` file for all users:

- Copy the `cshrc.lsf` file into `.cshrc`, or
- Add a line similar to the following to the end of `.cshrc`:  
`source /usr/lsf/lsf_8/conf/cshrc.lsf`

After running `cshrc.lsf`, use `setenv` to see the environment variable settings. For example:

#### setenv

```
PATH=/usr/lsf/lsf_8/8.0/linux2.6-glibc2.3-x86/bin
...
MANPATH=/usr/lsf/lsf_8/8.0/man
...
LSF_BINDIR=/usr/lsf/lsf_8/8.0/linux2.6-glibc2.3-x86/bin
LSF_SERVERDIR=/usr/lsf/lsf_8/8.0/linux2.6-glibc2.3-x86/etc
LSF_LIBDIR=/usr/lsf/lsf_8/8.0/linux2.6-glibc2.3-x86/lib
LD_LIBRARY_PATH=/usr/lsf/lsf_8/8.0/linux2.6-glibc2.3-x86/lib
XLSF_UIDDIR=/usr/lsf/lsf_8/8.0/linux2.6-glibc2.3-x86/lib/ui d
LSF_ENVDIR=/usr/lsf/lsf_8/conf
```

#### Note:

These variable settings are an example only. Your system may set additional variables.

## For sh, ksh, or bash

Add `profile.lsf` to the end of the `.profile` file for all users:

- Copy the `profile.lsf` file into `.profile`, or
- Add a line similar to following to the end of `.profile`:  

```
. /usr/lsf/lsf_8/conf/profile.lsf
```

After running `profile.lsf`, use the `set` command to see the environment variable settings. For example:

#### set

```
...
LD_LIBRARY_PATH=/usr/lsf/lsf_8/8.0/linux2.6-glibc2.3-x86/lib
LSF_BINDIR=/usr/lsf/lsf_8/8.0/linux2.6-glibc2.3-x86/bin
LSF_ENVDIR=/usr/lsf/lsf_8/conf
LSF_LIBDIR=/usr/lsf/lsf_8/8.0/linux2.6-glibc2.3-x86/lib
LSF_SERVERDIR=/usr/lsf/lsf_8/8.0/linux2.6-glibc2.3-x86/etc
MANPATH=/usr/lsf/lsf_8/8.0/man
PATH=/usr/lsf/lsf_8/8.0/linux2.6-glibc2.3-x86/bin
...
XLSF_UIDDIR=/usr/lsf/lsf_8/8.0/linux2.6-glibc2.3-x86/lib/ui d
...
```

#### Note:

These variable settings are an example only. Your system may set additional variables.

## cshrc.lsf and profile.lsf on dynamically added LSF slave hosts

Dynamically added LSF hosts that will not be master candidates are *slave hosts*. Each dynamic slave host has its own LSF binaries and local `lsf.conf` and shell environment scripts (`cshrc.lsf` and `profile.lsf`).

## LSF environment variables set by cshrc.lsf and profile.lsf

### LSF\_BINDIR

#### Syntax

```
LSF_BINDIR=dir
```

#### Description

Directory where LSF user commands are installed.

#### Examples

- Set in `csh` and `tcsh` by `cshrc.lsf`:  

```
setenv LSF_BINDIR /usr/lsf/lsf_8/8.0/linux2.6-glibc2.3-x86/bin
```
- Set and exported in `sh`, `ksh`, or `bash` by `profile.lsf`:  

```
LSF_BINDIR=/usr/lsf/lsf_8/8.0/linux2.6-glibc2.3-x86/bin
```

#### Values

- In `cshrc.lsf` for `csh` and `tcsh`:  

```
setenv LSF_BINDIR $LSF_TOP/$LSF_VERSION/$BINARY_TYPE/bin
```
- Set and exported in `profile.lsf` for `sh`, `ksh`, or `bash`:  

```
LSF_BINDIR=$LSF_TOP/$LSF_VERSION/$BINARY_TYPE/bin
```

### LSF\_ENVDIR

#### Syntax

```
LSF_ENVDIR=dir
```

#### Description

Directory containing the `lsf.conf` file.

By default, `lsf.conf` is installed by creating a shared copy in `LSF_CONFDIR` and adding a symbolic link from `/etc/lsf.conf` to the shared copy. If `LSF_ENVDIR` is set, the symbolic link is installed in `LSF_ENVDIR/lsf.conf`.

The `lsf.conf` file is a global environment configuration file for all LSF services and applications. The LSF default installation places the file in `LSF_CONFDIR`.

#### Examples

- Set in `csh` and `tcsh` by `cshrc.lsf`:  

```
setenv LSF_ENVDIR /usr/lsf/lsf_8/conf
```

- Set and exported in sh, ksh, or bash by profile.lsf:  
LSF\_ENVDIR=/usr/lsf/lsf\_8/conf

## Values

- In cshrc.lsf for csh and tcsh:  
setenv LSF\_ENVDIR \$LSF\_TOP/conf
- Set and exported in profile.lsf for sh, ksh, or bash:  
LSF\_ENVDIR=\$LSF\_TOP/conf

## LSF\_LIBDIR

### Syntax

**LSF\_LIBDIR=dir**

### Description

Directory where LSF libraries are installed. Library files are shared by all hosts of the same type.

### Examples

- Set in csh and tcsh by cshrc.lsf:  
setenv LSF\_LIBDIR /usr/lsf/lsf\_8/0/linux2.6-glibc2.3-x86/lib
- Set and exported in sh, ksh, or bash by profile.lsf:  
LSF\_LIBDIR=/usr/lsf/lsf\_8/0/linux2.6-glibc2.3-x86/lib

## Values

- In cshrc.lsf for csh and tcsh:  
setenv LSF\_LIBDIR \$LSF\_TOP/\$LSF\_VERSION/\$BINARY\_TYPE/lib
- Set and exported in profile.lsf for sh, ksh, or bash:  
LSF\_LIBDIR=\$LSF\_TOP/\$LSF\_VERSION/\$BINARY\_TYPE/lib

## LSF\_SERVERDIR

### Syntax

**LSF\_SERVERDIR=dir**

### Description

Directory where LSF server binaries and shell scripts are installed.

These include lim, res, nios, sbatchd, mbatchd, and mbschd. If you use elim, eauth, eexec, esub, etc, they are also installed in this directory.

### Examples

- Set in csh and tcsh by cshrc.lsf:  
setenv LSF\_SERVERDIR /usr/lsf/lsf\_8/0/linux2.6-glibc2.3-x86/etc
- Set and exported in sh, ksh, or bash by profile.lsf:  
LSF\_SERVERDIR=/usr/lsf/lsf\_8/0/linux2.6-glibc2.3-x86/etc

## Values

- In `cshrc.lsf` for `csh` and `tcsh`:  
`setenv LSF_SERVERDIR $LSF_TOP/$LSF_VERSION/$BINARY_TYPE/etc`
- Set and exported in `profile.lsf` for `sh`, `ksh`, or `bash`:  
`LSF_SERVERDIR=$LSF_TOP/$LSF_VERSION/$BINARY_TYPE/etc`

## XLSF\_UIDDIR

### Syntax

**XLSF\_UIDDIR**=*dir*

### Description

(UNIX and Linux only) Directory where Motif User Interface Definition files are stored. These files are platform-specific.

### Examples

- Set in `csh` and `tcsh` by `cshrc.lsf`:  
`setenv XLSF_UIDDIR /usr/l sf/lsf_8/8.0/linux2.6-glibc2.3-x86/lib/ui d`
- Set and exported in `sh`, `ksh`, or `bash` by `profile.lsf`:  
`XLSF_UIDDIR=/usr/l sf/lsf_8/8.0/linux2.6-glibc2.3-x86/lib/ui d`

## Values

- In `cshrc.lsf` for `csh` and `tcsh`:  
`setenv XLSF_UIDDIR $LSF_TOP/$LSF_VERSION/$BINARY_TYPE/lib/ui d`
- Set and exported in `profile.lsf` for `sh`, `ksh`, or `bash`:  
`XLSF_UIDDIR=$LSF_TOP/$LSF_VERSION/$BINARY_TYPE/lib/ui d`

## Platform EGO environment variables set by cshrc.lsf and profile.lsf

See the *Platform EGO Reference* for more information about these variables.

## EGO\_BINDIR

### Syntax

**EGO\_BINDIR**=*dir*

### Description

Directory where Platform EGO user commands are installed.

### Examples

- Set in `csh` and `tcsh` by `cshrc.lsf`:  
`setenv EGO_BINDIR /usr/l sf/lsf_8/8.0/linux2.6-glibc2.3-x86/bin`
- Set and exported in `sh`, `ksh`, or `bash` by `profile.lsf`:  
`EGO_BINDIR=/usr/l sf/lsf_8/8.0/linux2.6-glibc2.3-x86/bin`

## Values

- In `cshrc.lsf` for `cs`h and `tc`sh:  

```
setenv EGO_BINDIR $LSF_BINDIR
```
- Set and exported in `profile.lsf` for `sh`, `ksh`, or `bash`:  

```
EGO_BINDIR=$LSF_BINDIR
```

## EGO\_CONFDIR

### Syntax

```
EGO_CONFDIR=dir
```

### Description

Directory containing the `ego.conf` file.

### Examples

- Set in `cs`h and `tc`sh by `cshrc.lsf`:  

```
setenv EGO_CONFDIR /usr/lsf/lsf_8/conf/ego/lsf1.2.3/kernel
```
- Set and exported in `sh`, `ksh`, or `bash` by `profile.lsf`:  

```
EGO_CONFDIR=/usr/lsf/lsf_8/conf/ego/lsf1.2.3/kernel
```

## Values

- In `cshrc.lsf` for `cs`h and `tc`sh:  

```
setenv EGO_CONFDIR /usr/lsf/lsf_8/conf/ego/lsf1.2.3/kernel
```
- Set and exported in `profile.lsf` for `sh`, `ksh`, or `bash`:  

```
EGO_CONFDIR=/usr/lsf/lsf_8/conf/ego/lsf1.2.3/kernel
```

## EGO\_ESRVDIR

### Syntax

```
EGO_ESRVDIR=dir
```

### Description

Directory where the EGO the service controller configuration files are stored.

### Examples

- Set in `cs`h and `tc`sh by `cshrc.lsf`:  

```
setenv EGO_ESRVDIR /usr/lsf/lsf_8/conf/ego/lsf702/eservice
```
- Set and exported in `sh`, `ksh`, or `bash` by `profile.lsf`:  

```
EGO_ESRVDIR=/usr/lsf/lsf_8/conf/ego/lsf702/eservice
```

## Values

- In `cshrc.lsf` for `cs`h and `tc`sh:  

```
setenv EGO_ESRVDIR /usr/lsf/lsf_8/conf/ego/lsf702/eservice
```
- Set and exported in `profile.lsf` for `sh`, `ksh`, or `bash`:  

```
EGO_ESRVDIR=/usr/lsf/lsf_8/conf/ego/lsf702/eservice
```

## EGO\_LIBDIR

### Syntax

**EGO\_LIBDIR=dir**

### Description

Directory where EGO libraries are installed. Library files are shared by all hosts of the same type.

### Examples

- Set in csh and tcsh by cshrc.lsf:  

```
setenv EGO_LIBDIR /usr/l sf/l sf_8/8. 0/lin ux2. 6- gl i bc2. 3- x86/1 i b
```
- Set and exported in sh, ksh, or bash by profile.lsf:  

```
EGO_LIBDIR=/usr/l sf/l sf_8/8. 0/lin ux2. 6- gl i bc2. 3- x86/1 i b
```

### Values

- In cshrc.lsf for csh and tcsh:  

```
setenv EGO_LIBDIR $LSF_LIBDIR
```
- Set and exported in profile.lsf for sh, ksh, or bash:  

```
EGO_LIBDIR=$LSF_LIBDIR
```

## EGO\_LOCAL\_CONFDIR

### Syntax

**EGO\_LOCAL\_CONFDIR=dir**

### Description

The local EGO configuration directory containing the ego.conf file.

### Examples

- Set in csh and tcsh by cshrc.lsf:  

```
setenv EGO_LOCAL_CONFDIR /usr/l sf/l sf_8/conf/ego/l sf1. 2. 3/kernel
```
- Set and exported in sh, ksh, or bash by profile.lsf:  

```
EGO_LOCAL_CONFDIR=/usr/l sf/l sf_8/conf/ego/l sf1. 2. 3/kernel
```

### Values

- In cshrc.lsf for csh and tcsh:  

```
setenv EGO_LOCAL_CONFDIR /usr/l sf/l sf_8/conf/ego/l sf1. 2. 3/kernel
```
- Set and exported in profile.lsf for sh, ksh, or bash:  

```
EGO_LOCAL_CONFDIR=/usr/l sf/l sf_8/conf/ego/l sf1. 2. 3/kernel
```

## EGO\_SERVERDIR

### Syntax

**EGO\_SERVERDIR=dir**

## Description

Directory where EGO server binaries and shell scripts are installed. These include `venkd`, `pem`, `egosc`, and shell scripts for EGO startup and shutdown.

## Examples

- Set in `csh` and `tcsh` by `cshrc.lsf`:  

```
setenv EGO_SERVERDIR /usr/lsf/lsf_8/8.0/linux2.6-glibc2.3-x86/etc
```
- Set and exported in `sh`, `ksh`, or `bash` by `profile.lsf`:  

```
EGO_SERVERDIR=/usr/lsf/lsf_8/8.0/linux2.6-glibc2.3-x86/etc
```

## Values

- In `cshrc.lsf` for `csh` and `tcsh`:  

```
setenv EGO_SERVERDIR $LSF_SERVERDIR
```
- Set and exported in `profile.lsf` for `sh`, `ksh`, or `bash`:  

```
EGO_SERVERDIR=$LSF_SERVERDIR
```

## EGO\_TOP

### Syntax

**EGO\_TOP**=*dir*

### Description

The the top-level installation directory. The path to `EGO_TOP` must be shared and accessible to all hosts in the cluster. Equivalent to `LSF_TOP`.

### Examples

- Set in `csh` and `tcsh` by `cshrc.lsf`:  

```
setenv EGO_TOP /usr/lsf/lsf_8
```
- Set and exported in `sh`, `ksh`, or `bash` by `profile.lsf`:  

```
EGO_TOP=/usr/lsf/lsf_8
```

### Values

- In `cshrc.lsf` for `csh` and `tcsh`:  

```
setenv EGO_TOP /usr/lsf/lsf_8
```
- Set and exported in `profile.lsf` for `sh`, `ksh`, or `bash`:  

```
EGO_TOP=/usr/lsf/lsf_8
```

## hosts

For hosts with multiple IP addresses and different official host names configured at the system level, this file associates the host names and IP addresses in LSF.

By default, LSF assumes each host in the cluster:

- Has a unique “official” host name
- Can resolve its IP address from its name
- Can resolve its official name from its IP address

Hosts with only one IP address, or hosts with multiple IP addresses that already resolve to a unique official host name should not be configured in this file: they are resolved using the default method for your system (for example, local configuration files like `/etc/hosts` or through DNS.)

The LSF `hosts` file is used in environments where:

- Machines in cluster have multiple network interfaces and cannot be set up in the system with a unique official host name
- DNS is slow or not configured properly
- Machines have special topology requirements; for example, in HPC systems where it is desirable to map multiple actual hosts to a single “head end” host

The LSF `hosts` file is not installed by default. It is usually located in the directory specified by `LSF_CONFDIR`. The format of `LSF_CONFDIR/hosts` is similar to the format of the `/etc/hosts` file on UNIX machines.

## hosts file structure

One line for each IP address, consisting of the IP address, followed by the official host name, optionally followed by host aliases, all separated by spaces or tabs. Each line has the form:

```
ip_address official_name [alias [alias ...]]
```

IP addresses can have either a dotted quad notation (IPv4) or IP Next Generation (IPv6) format. You can use IPv6 addresses if you define the parameter `LSF_ENABLE_SUPPORT_IPV6` in `lsf.conf`; you do not have to map IPv4 addresses to an IPv6 format.

Use consecutive lines for IP addresses belonging to the same host. You can assign different aliases to different addresses.

Use a pound sign (#) to indicate a comment (the rest of the line is not read by LSF). Do not use `#if` as this is reserved syntax for time-based configuration.

## IP address

Written using an IPv4 or IPv6 format. LSF supports both formats; you do not have to map IPv4 addresses to an IPv6 format (if you define the parameter `LSF_ENABLE_SUPPORT_IPV6` in `lsf.conf`).

- IPv4 format: `nnn.nnn.nnn.nnn`
- IPv6 format: `nnnn:nnnn:nnnn:nnnn:nnnn:nnnn:nnnn:nnnn`

## Official host name

The official host name. Single character names are not allowed.

Specify `-GATEWAY` or `-GW` as part of the host name if the host serves as a GATEWAY.

Specify `-TAC` as the last part of the host name if the host is a TAC and is a DoD host.

Specify the host name in the format defined in Internet RFC 952, which states:

A “name” (Net, Host, Gateway, or Domain name) is a text string up to 24 characters drawn from the alphabet (A-Z), digits (0-9), minus sign (-), and period (.). Periods are only allowed when they serve to delimit components of “domain style names”. (See RFC 921, “Domain Name System Implementation Schedule”, for background). No blank or space characters are permitted as part of a name. No distinction is made between upper and lower case. The first character must be an alpha character. The last character must not be a minus sign or a period.

RFC 952 has been modified by RFC 1123 to relax the restriction on the first character being a digit.

For maximum interoperability with the Internet, you should use host names no longer than 24 characters for the host portion (exclusive of the domain component).

## Aliases

Optional. Aliases to the host name.

The default host file syntax

```
ip_address official_name [ alias [ alias ... ] ]
```

is powerful and flexible, but it is difficult to configure in systems where a single host name has many aliases, and in multihomed host environments.

In these cases, the `hosts` file can become very large and unmanageable, and configuration is prone to error.

The syntax of the LSF `hosts` file supports host name ranges as aliases for an IP address. This simplifies the host name alias specification.

To use host name ranges as aliases, the host names must consist of a fixed node group name prefix and node indices, specified in a form like:

```
host_name[ index_x- index_y, index_m, index_a- index_b ]
```

For example:

```
atlasD0[ 0- 3, 4, 5- 6, ... ]
```

is equivalent to:

```
atlasD0[ 0- 6, ... ]
```

The node list does not need to be a continuous range (some nodes can be configured out). Node indices can be numbers or letters (both upper case and lower case).

For example, some systems map internal compute nodes to single LSF host names. A host file might contain 64 lines, each specifying an LSF host name and 32 node names that correspond to each LSF host:

```
...
177.16.1.1 atlasD0 atlas0 atlas1 atlas2 atlas3 atlas4 ... atlas31
177.16.1.2 atlasD1 atlas32 atlas33 atlas34 atlas35 atlas36 ... atlas63
...
```

In the new format, you still map the nodes to the LSF hosts, so the number of lines remains the same, but the format is simplified because you only have to specify ranges for the nodes, not each node individually as an alias:

```
...
177.16.1.1 atlasD0 atlas[ 0- 31 ]
177.16.1.2 atlasD1 atlas[ 32- 63 ]
...
```

hosts

You can use either an IPv4 or an IPv6 format for the IP address (if you define the parameter `LSF_ENABLE_SUPPORT_IPV6` in `lsf.conf`).

## IPv4 Example

```
192.168.1.1 hostA hostB  
192.168.2.2 hostA hostC host-C
```

In this example, `hostA` has 2 IP addresses and 3 aliases. The alias `hostB` specifies the first address, and the aliases `hostC` and `host-C` specify the second address. LSF uses the official host name, `hostA`, to identify that both IP addresses belong to the same host.

## IPv6 Example

```
3ffe:b80:3:1a91::2 hostA hostB 3ffe:b80:3:1a91::3 hostA hostC host-C
```

In this example, `hostA` has 2 IP addresses and 3 aliases. The alias `hostB` specifies the first address, and the aliases `hostC` and `host-C` specify the second address. LSF uses the official host name, `hostA`, to identify that both IP addresses belong to the same host.

# install.config

## About install.config

The `install.config` file contains options for LSF installation and configuration. Use `lsfinstall -f install.config` to install LSF using the options specified in `install.config`.

## Template location

A template `install.config` is included in the installation script tar file `lsf8_1sfinstall.tar.Z` and is located in the `lsf8_1sfinstall` directory created when you uncompress and extract installation script tar file. Edit the file and uncomment the options you want in the template file. Replace the example values with your own settings to specify the options for your new installation.

---

### Important:

The sample values in the `install.config` template file are examples only. They are not default installation values.

---

After installation, the `install.config` containing the options you specified is located in `LSF_TOP/8/install/`.

## Format

Each entry in `install.config` has the form:

```
NAME="STRING1 STRING2 ..."
```

The equal sign `=` must follow each `NAME` even if no value follows and there should be no spaces around the equal sign.

A value that contains multiple strings separated by spaces must be enclosed in quotation marks.

Blank lines and lines starting with a pound sign (`#`) are ignored.

## Parameters

- EGO\_DAEMON\_CONTROL
- ENABLE\_DYNAMIC\_HOSTS
- ENABLE\_EGO
- ENABLE\_HPC\_CONFIG
- EP\_BACKUP
- LSF\_ADD\_SERVERS
- LSF\_ADD\_CLIENTS
- LSF\_ADMINS
- LSF\_CLUSTER\_NAME
- LSF\_DYNAMIC\_HOST\_WAIT\_TIME
- LSF\_LICENSE
- LSF\_MASTER\_LIST
- LSF\_QUIET\_INST
- LSF\_TARDIR
- LSF\_TOP
- PATCH\_BACKUP\_DIR

install.config

- PATCH\_HISTORY\_DIR

## EGO\_DAEMON\_CONTROL

### Syntax

```
EGO_DAEMON_CONTROL="Y" | "N"
```

### Description

Enables EGO to control LSF `res` and `sbat chd`. Set the value to "Y" if you want EGO Service Controller to start `res` and `sbat chd`, and restart if they fail. To avoid conflicts, leave this parameter undefined if you use a script to start up LSF daemons.

---

**Note:**

If you specify `EGO_ENABLE="N"`, this parameter is ignored.

---

### Example

```
EGO_DAEMON_CONTROL="N"
```

### Default

N (`res` and `sbat chd` are started manually)

## ENABLE\_DYNAMIC\_HOSTS

### Syntax

```
ENABLE_DYNAMIC_HOSTS="Y" | "N"
```

### Description

Enables dynamically adding and removing hosts. Set the value to "Y" if you want to allow dynamically added hosts.

If you enable dynamic hosts, any host can connect to cluster. To enable security, configure `LSF_HOST_ADDR_RANGE` in `lsf.cluster.cluster_name` after installation and restrict the hosts that can connect to your cluster.

### Example

```
ENABLE_DYNAMIC_HOSTS="N"
```

### Default

N (dynamic hosts not allowed)

## ENABLE\_EGO

### Syntax

```
ENABLE_EGO="Y" | "N"
```

## Description

Enables Platform EGO functionality in the LSF cluster.

`ENABLE_EGO="Y"` causes `lsfi nstall` to uncomment `LSF_EGO_ENVDIR` and sets `LSF_ENABLE_EGO="Y"` in `lsf.conf`.

`ENABLE_EGO="N"` causes `lsfi nstall` to comment out `LSF_EGO_ENVDIR` and sets `LSF_ENABLE_EGO="N"` in `lsf.conf`.

Set the value to "Y" if you want to take advantage of the following LSF features that depend on EGO:

- LSF daemon control by EGO Service Controller
- EGO-enabled SLA scheduling

## Default

N (EGO is disabled in the LSF cluster)

## ENABLE\_HPC\_CONFIG

### Syntax

`ENABLE_HPC_CONFIG="Y" | "N"`

### Description

Set the value to "Y" to enable LSF HPC features and add HPC configuration parameters to the cluster.

### Default

N (HPC features are disabled.)

## EP\_BACKUP

### Syntax

`EP_BACKUP="Y" | "N"`

### Description

Enables backup and rollback for enhancement packs. Set the value to "N" to disable backups when installing enhancement packs (you will not be able to roll back to the previous patch level after installing an EP, but you will still be able to roll back any fixes installed on the new EP).

You may disable backups to speed up install time, to save disk space, or because you have your own methods to back up the cluster.

### Default

Y (backup and rollback are fully enabled)

## LSF\_ADD\_SERVERS

### Syntax

`LSF_ADD_SERVERS="host_name [ host_name...]"`

## Description

List of additional LSF server hosts.

The hosts in `LSF_MASTER_LIST` are always LSF servers. You can specify additional server hosts. Specify a list of host names two ways:

- Host names separated by spaces
- Name of a file containing a list of host names, one host per line.

## Valid Values

Any valid LSF host name.

## Example 1

List of host names:

```
LSF_ADD_SERVERS="hosta hostb hostc hostd"
```

## Example 2

Host list file:

```
LSF_ADD_SERVERS=: lsf_server_hosts
```

The file `lsf_server_hosts` contains a list of hosts:

```
hosta  
hostb  
hostc  
hostd
```

## Default

Only hosts in `LSF_MASTER_LIST` are LSF servers.

# LSF\_ADD\_CLIENTS

## Syntax

```
LSF_ADD_CLIENTS="host_name [host_name...]"
```

## Description

List of LSF client-only hosts.

---

### Tip:

After installation, you must manually edit `lsf.cluster.cluster_name` to include the host model and type of each client listed in `LSF_ADD_CLIENTS`.

---

## Valid Values

Any valid LSF host name.

## Example 1

List of host names:

```
LSF_ADD_CLIENTS="hoste hostf"
```

## Example 2

Host list file:

```
LSF_ADD_CLIENTS=:lsf_client_hosts
```

The file `lsf_client_hosts` contains a list of hosts:

```
hoste
hostf
```

## Default

No client hosts installed.

## LSF\_ADMINS

### Syntax

```
LSF_ADMINS="user_name [ user_name ... ]"
```

### Description

Required. List of LSF administrators.

The first user account name in the list is the primary LSF administrator. It cannot be the root user account.

Typically this account is named `lsfadmin`. It owns the LSF configuration files and log files for job events. It also has permission to reconfigure LSF and to control batch jobs submitted by other users. It typically does not have authority to start LSF daemons. Usually, only root has permission to start LSF daemons.

All the LSF administrator accounts must exist on all hosts in the cluster before you install LSF. Secondary LSF administrators are optional.

---

#### Caution:

You should *not* configure the root account as the primary LSF administrator.

---

### Valid Values

Existing user accounts

### Example

```
LSF_ADMINS="lsfadmin user1 user2"
```

### Default

None—required variable

## LSF\_CLUSTER\_NAME

### Syntax

```
LSF_CLUSTER_NAME="cluster_name"
```

### Description

Required. The name of the LSF cluster.

install.config

## Example

```
LSF_CLUSTER_NAME="cluster1"
```

## Valid Values

Any alphanumeric string containing no more than 39 characters. The name cannot contain white spaces.

---

### Important:

Do not use the name of any host, user, or user group as the name of your cluster.

---

## Default

None—required variable

## LSF\_DYNAMIC\_HOST\_WAIT\_TIME

### Syntax

```
LSF_DYNAMIC_HOST_WAIT_TIME=seconds
```

### Description

Time in seconds slave LIM waits after startup before calling master LIM to add the slave host dynamically.

This parameter only takes effect if you set `ENABLE_DYNAMIC_HOSTS="Y"` in this file. If the slave LIM receives the master announcement while it is waiting, it does not call the master LIM to add itself.

### Recommended value

Up to 60 seconds for every 1000 hosts in the cluster, for a maximum of 15 minutes. Selecting a smaller value will result in a quicker response time for new hosts at the expense of an increased load on the master LIM.

## Example

```
LSF_DYNAMIC_HOST_WAIT_TIME=60
```

Hosts will wait 60 seconds from startup to receive an acknowledgement from the master LIM. If it does not receive the acknowledgement within the 60 seconds, it will send a request for the master LIM to add it to the cluster.

## Default

Slave LIM waits forever

## LSF\_LICENSE

### Syntax

```
LSF_LICENSE="/path/license_file"
```

### Description

Full path to the name of the LSF license file, `license.dat`.

You must have a valid license file to install LSF.

---

**Caution:**

If you do not specify `LSF_LICENSE`, or `lsfi nstal l` cannot find a valid license file in the default location, `lsfi nstal l` exits.

---

## Example

```
LSF_LICENSE="/usr/share/lsf_distrib/license.dat"
```

## Default

The parent directory of the current working directory. For example, if `lsfi nstal l` is running under `usr/share/lsf_distrib/lsf_instal l` the `LSF_LICENSE` default value is `usr/share/lsf_distrib/license.dat`.

## LSF\_MASTER\_LIST

### Syntax

```
LSF_MASTER_LIST="host_name [ host_name ...]"
```

### Description

Required for a first-time installation. List of LSF server hosts to be master or master candidates in the cluster.

You must specify at least one valid server host to start the cluster. The first host listed is the LSF master host.

During upgrade, specify the existing value.

### Valid Values

LSF server host names

### Example

```
LSF_MASTER_LIST="hosta hostb hostc hostd"
```

### Default

None — required variable

## LSF\_QUIET\_INST

### Syntax

```
LSF_QUIET_INST="Y" | "N"
```

### Description

Enables quiet installation.

Set the value to Y if you want to hide the LSF installation messages.

install.config

## Example

```
LSF_QUIET_INSTALL="Y"
```

## Default

N (installer displays messages during installation)

## LSF\_TARDIR

### Syntax

```
LSF_TARDIR="/path"
```

### Description

Full path to the directory containing the LSF distribution tar files.

### Example

```
LSF_TARDIR="/usr/share/lfs_distribution"
```

### Default

The parent directory of the current working directory. For example, if `lfsinstall` is running under `usr/share/lfs_distribution/lfsinstall` the LSF\_TARDIR default value is `usr/share/lfs_distribution`.

## LSF\_TOP

### Syntax

```
LSF_TOP="/path"
```

### Description

Required. Full path to the top-level LSF installation directory.

### Valid Value

The path to LSF\_TOP must be shared and accessible to all hosts in the cluster. It cannot be the root directory (/). The file system containing LSF\_TOP must have enough disk space for all host types (approximately 300 MB per host type).

### Example

```
LSF_TOP="/usr/share/lfs"
```

### Default

None — required variable

## PATCH\_BACKUP\_DIR

### Syntax

```
PATCH_BACKUP_DIR="/path"
```

## Description

Full path to the patch backup directory. This parameter is used when you install a new cluster for the first time, and is ignored for all other cases.

The file system containing the patch backup directory must have sufficient disk space to back up your files (approximately 400 MB per binary type if you want to be able to install and roll back one enhancement pack and a few additional fixes). It cannot be the root directory (/).

If the directory already exists, it must be writable by the cluster administrator (`lsfadmin`).

If you need to change the directory after installation, edit `PATCH_BACKUP_DIR` in `LSF_TOP/patch.conf` and move the saved backup files to the new directory manually.

## Example

```
PATCH_BACKUP_DIR="/usr/share/lsf/patch/backup"
```

## Default

```
LSF_TOP/patch/backup
```

## PATCH\_HISTORY\_DIR

### Syntax

```
PATCH_HISTORY_DIR="/path"
```

## Description

Full path to the patch history directory. This parameter is used when you install a new cluster for the first time, and is ignored for all other cases.

It cannot be the root directory (/). If the directory already exists, it must be writable by `lsfadmin`.

The location is saved as `PATCH_HISTORY_DIR` in `LSF_TOP/patch.conf`. Do not change the directory after installation.

## Example

```
PATCH_HISTORY_DIR="/usr/share/lsf/patch"
```

## Default

```
LSF_TOP/patch
```

## lim.acct

The `lim.acct` file is the log file for Load Information Manager (LIM). Produced by `lsmmon`, `lim.acct` contains host load information collected and distributed by LIM.

## lim.acct structure

The first line of `lim.acct` contains a list of load index names separated by spaces. This list of load index names can be specified in the `lsmmon` command line. The default list is "r15s r1m r15m ut pg l s i t swp mem tmp". Subsequent lines in the file contain the host's load information at the time the information was recorded.

## Fields

Fields are ordered in the following sequence:

### **time (%ld)**

The time when the load information is written to the log file

### **host name (%s)**

The name of the host.

### **status of host (%d)**

An array of integers. The first integer marks the operation status of the host. Additional integers are used as a bit map to indicate load status of the host. An integer can be used for 32 load indices. If the number of user defined load indices is not more than 21, only one integer is used for both built-in load indices and external load indices. See the `hostload` structure in `ls_load(3)` for the description of these fields.

### **indexvalue (%f)**

A sequence of load index values. Each value corresponds to the index name in the first line of `lim.acct`. The order in which the index values are listed is the same as the order of the index names.

## lsb.acct

The `lsb.acct` file is the batch job log file of LSF. The master batch daemon (see `mbatchd(8)`) generates a record for each job completion or failure. The record is appended to the job log file `lsb.acct`.

The file is located in `LSB_SHAREDIR/cluster_name/logdir`, where `LSB_SHAREDIR` must be defined in `lsf.conf(5)` and `cluster_name` is the name of the LSF cluster, as returned by `lsid(1)`. See `mbatchd(8)` for the description of `LSB_SHAREDIR`.

The `bacct` command uses the current `lsb.acct` file for its output.

## lsb.acct structure

The job log file is an ASCII file with one record per line. The fields of a record are separated by blanks. If the value of some field is unavailable, a pair of double quotation marks (" ") is logged for character string, 0 for time and number, and -1 for resource usage.

## Configuring automatic archiving

The following parameters in `lsb.params` affect how records are logged to `lsb.acct`:

### **ACCT\_ARCHIVE\_AGE=*days***

Enables automatic archiving of LSF accounting log files, and specifies the archive interval. LSF archives the current log file if the length of time from its creation date exceeds the specified number of days.

By default there is no limit to the age of `lsb.acct`.

### **ACCT\_ARCHIVE\_SIZE=*kilobytes***

Enables automatic archiving of LSF accounting log files, and specifies the archive threshold. LSF archives the current log file if its size exceeds the specified number of kilobytes.

By default, there is no limit to the size of `lsb.acct`.

### **ACCT\_ARCHIVE\_TIME=*hh:mm***

Enables automatic archiving of LSF accounting log file `lsb.acct`, and specifies the time of day to archive the current log file.

By default, no time is set for archiving `lsb.acct`.

### **MAX\_ACCT\_ARCHIVE\_FILE=*integer***

Enables automatic deletion of archived LSF accounting log files and specifies the archive limit.

By default, `lsb.acct.n` files are not automatically deleted.

## Records and fields

The fields of a record are separated by blanks. The first string of an event record indicates its type. The following types of events are recorded:

- `JOB_FINISH`

- EVENT\_ADRSV\_FINISH
- JOB\_RESIZE

## JOB\_FINISH

A job has finished.

If LSF\_HPC\_EXTENSIONS="SHORT\_EVENTFILE" is specified in lsf.conf, older daemons and commands (pre-LSF Version 6.0) cannot recognize the lsb.acct file format.

The fields in order of occurrence are:

**Event type (%s)**

Which is "JOB\_FINISH"

**Version Number (%s)**

Version number of the log file format

**Event Time (%d)**

Time the event was logged (in seconds since the epoch)

**jobId (%d)**

ID for the job

**userId (%d)**

UNIX user ID of the submitter

**options (%d)**

Bit flags for job processing

**numProcessors (%d)**

Number of processors initially requested for execution

**submitTime (%d)**

Job submission time

**beginTime (%d)**

Job start time – the job should be started at or after this time

**termTime (%d)**

Job termination deadline – the job should be terminated by this time

**startTime (%d)**

Job dispatch time – time job was dispatched for execution

**userName (%s)**

User name of the submitter

**queue (%s)**

Name of the job queue to which the job was submitted

**resReq (%s)**

Resource requirement specified by the user

**dependCond (%s)**

Job dependency condition specified by the user

**preExecCmd (%s)**

Pre-execution command specified by the user

**fromHost (%s)**

Submission host name

**cwd (%s)**

Current working directory (up to 4094 characters for UNIX or 512 characters for Windows)

**inFile (%s)**

Input file name (up to 4094 characters for UNIX or 512 characters for Windows)

**outFile (%s)**

output file name (up to 4094 characters for UNIX or 512 characters for Windows)

**errFile (%s)**

Error output file name (up to 4094 characters for UNIX or 512 characters for Windows)

**jobFile (%s)**

Job script file name

**numAskedHosts (%d)**

Number of host names to which job dispatching will be limited

**askedHosts (%s)**

List of host names to which job dispatching will be limited (%s for each); nothing is logged to the record for this value if the last field value is 0. If there is more than one host name, then each additional host name will be returned in its own field

**numExHosts (%d)**

Number of processors used for execution

If LSF\_HPC\_EXTENSIONS="SHORT\_EVENTFILE" is specified in `lsf.conf`, the value of this field is the number of `.hosts` listed in the `execHosts` field.

Logged value reflects the allocation at job finish time.

**execHosts (%s)**

List of execution host names (%s for each); nothing is logged to the record for this value if the last field value is 0.

If LSF\_HPC\_EXTENSIONS="SHORT\_EVENTFILE" is specified in `lsf.conf`, the value of this field is logged in a shortened format.

The logged value reflects the allocation at job finish time.

**jStatus (%d)**

Job status. The number 32 represents EXIT, 64 represents DONE

**hostFactor (%f)**

CPU factor of the first execution host.

**jobName (%s)**

Job name (up to 4094 characters).

**command (%s)**

Complete batch job command specified by the user (up to 4094 characters for UNIX or 512 characters for Windows).

**lsfRusage (%f)**

The following fields contain resource usage information for the job (see `getrusage(2)`). If the value of some field is unavailable (due to job exit or the difference among the operating systems), -1 will be logged. Times are measured in seconds, and sizes are measured in KB.

**ru\_utime (%f)**

User time used

**ru\_stime (%f)**

System time used

**ru\_maxrss (%f)**

Maximum shared text size

**ru\_ixrss (%f)**

Integral of the shared text size over time (in KB seconds)

**ru\_ismrss (%f)**

Integral of the shared memory size over time (valid only on Ultrix)

**ru\_idrss (%f)**

Integral of the unshared data size over time

**ru\_isrss (%f)**

Integral of the unshared stack size over time

**ru\_minflt (%f)**

Number of page reclaims

**ru\_majflt (%f)**

Number of page faults

**ru\_nswap (%f)**

Number of times the process was swapped out

<b>ru_inblock (%f)</b>	Number of block input operations
<b>ru_oublock (%f)</b>	Number of block output operations
<b>ru_ioch (%f)</b>	Number of characters read and written (valid only on HP-UX)
<b>ru_msgsnd (%f)</b>	Number of System V IPC messages sent
<b>ru_msgrcv (%f)</b>	Number of messages received
<b>ru_nsignals (%f)</b>	Number of signals received
<b>ru_nvcsw (%f)</b>	Number of voluntary context switches
<b>ru_nivcsw (%f)</b>	Number of involuntary context switches
<b>ru_exutime (%f)</b>	Exact user time used (valid only on ConvexOS)
<b>mailUser (%s)</b>	Name of the user to whom job related mail was sent
<b>projectName (%s)</b>	LSF project name
<b>exitStatus (%d)</b>	UNIX exit status of the job
<b>maxNumProcessors (%d)</b>	Maximum number of processors specified for the job
<b>loginShell (%s)</b>	Login shell used for the job
<b>timeEvent (%s)</b>	Time event string for the job - JobScheduler only
<b>idx (%d)</b>	Job array index
<b>maxRMem (%d)</b>	

Maximum resident memory usage in the unit specified by LSF\_UNIT\_FOR\_LIMITS in `lsf.conf` of all processes in the job

**maxRSwap (%d)**

Maximum virtual memory usage in the unit specified by LSF\_UNIT\_FOR\_LIMITS in `lsf.conf` of all processes in the job

**inFileSpool (%s)**

Spool input file (up to 4094 characters for UNIX or 512 characters for Windows)

**commandSpool (%s)**

Spool command file (up to 4094 characters for UNIX or 512 characters for Windows)

**rsvld %s**

Advance reservation ID for a user group name less than 120 characters long; for example, "user2#0"

If the advance reservation user group name is longer than 120 characters, the rsvld field output appears last.

**sla (%s)**

SLA service class name under which the job runs

**exceptMask (%d)**

Job exception handling

Values:

- J\_EXCEPT\_OVERRUN 0x02
- J\_EXCEPT\_UNDERUN 0x04
- J\_EXCEPT\_IDLE 0x80

**additionalInfo (%s)**

Placement information of HPC jobs

**exitInfo (%d)**

Job termination reason, mapped to corresponding termination keyword displayed by `bacct`.

**warningAction (%s)**

Job warning action

**warningTimePeriod (%d)**

Job warning time period in seconds

**chargedSAAP (%s)**

SAAP charged to a job

**licenseProject (%s)**

Platform License Scheduler project name

**app (%s)**

Application profile name

**postExecCmd (%s)**

Post-execution command to run on the execution host after the job finishes

**runtimeEstimation (%d)**

Estimated run time for the job, calculated as the CPU factor of the submission host multiplied by the runtime estimate (in seconds).

**jobGroupName (%s)**

Job group name

**requeueEvalues (%s)**

Requeue exit value

**options2 (%d)**

Bit flags for job processing

**resizeNotifyCmd (%s)**

Resize notification command to be invoked on the first execution host upon a resize request.

**lastResizeTime (%d)**

Last resize time. The latest wall clock time when a job allocation is changed.

**rsvld %s**

Advance reservation ID for a user group name more than 120 characters long.

If the advance reservation user group name is longer than 120 characters, the rsvld field output appears last.

**jobDescription (%s)**

Job description (up to 4094 characters).

**submitEXT**

Submission extension field, reserved for internal use.

**Num (%d)**

Number of elements (key-value pairs) in the structure.

**key (%s)**

Reserved for internal use.

**value (%s)**

Reserved for internal use.

**numHostRusage(%d)**

The number of host-based resource usage entries (hostRusage) that follow. 0 unless HPC\_EXTENSIONS="HOST\_RUSAGE" is set in lsf.conf.

**hostRusage**

The following fields contain host-based resource usage information for the job., and only appear for parallel jobs when HPC\_EXTENSIONS="HOST\_RUSAGE" is set in lsf.conf.

**hostname (%s)**

Name of the host.

**mem(%d)**

Total resident memory usage of all processes in the job running on this host.

**swap(%d)**

The total virtual memory usage of all processes in the job running on this host.

**utime(%d)**

User time used on this host.

**stime(%d)**

System time used on this host.

**hHostExtendInfo(%d)**

Number of following key-value pairs containing extended host information (PGIDs and PIDs). Set to 0 in lsb.events, lsb.acct, and lsb.stream files.

## EVENT\_ADRSV\_FINISH

An advance reservation has expired. The fields in order of occurrence are:

**Event type (%s)**

Which is "EVENT\_ADRSV\_FINISH"

**Version Number (%s)**

Version number of the log file format

**Event Logging Time (%d)**

Time the event was logged (in seconds since the epoch); for example, "1038942015"

**Reservation Creation Time (%d)**

Time the advance reservation was created (in seconds since the epoch); for example, "1038938898"

**Reservation Type (%d)**

Type of advance reservation request:

- User reservation (RSV\_OPTION\_USER, defined as 0x001)
- User group reservation (RSV\_OPTION\_GROUP, defined as 0x002)
- System reservation (RSV\_OPTION\_SYSTEM, defined as 0x004)
- Recurring reservation (RSV\_OPTION\_RECUR, defined as 0x008)

For example, "9" is a recurring reservation created for a user.

**Creator ID (%d)**

UNIX user ID of the reservation creator; for example, "30408"

**Reservation ID (rsvid %s)**

For example, "user2#0"

**User Name (%s)**

User name of the reservation user; for example, "user2"

**Time Window (%s)**

Time window of the reservation:

- One-time reservation in seconds since the epoch; for example, "1033761000- 1033761600"
- Recurring reservation; for example, "17: 50- 18: 00"

**Creator Name (%s)**

User name of the reservation creator; for example, "user1"

**Duration (%d)**

Duration of the reservation, in hours, minutes, seconds; for example, "600" is 6 hours, 0 minutes, 0 seconds

**Number of Resources (%d)**

Number of reserved resource pairs in the resource list; for example "2" indicates 2 resource pairs ("hostA 1 hostB 1")

**Host Name (%s)**

Reservation host name; for example, "hostA"

**Number of CPUs (%d)**

Number of reserved CPUs; for example "1"

## JOB\_RESIZE

When there is an allocation change, LSF logs the event after mbatchd receives "JOB\_RESIZE\_NOTIFY\_DONE" event. From lastResizeTime and eventTime, people can easily calculate the duration of previous job allocation. The fields in order of occurrence are:

**Version number (%s)**

The version number.

**Event Time (%d)**

Time the event was logged (in seconds since the epoch).

**jobId (%d)**

ID for the job.

**tdx (%d)**

Job array index.

**startTime (%d)**

The start time of the running job.

**userId (%d)**

UNIX user ID of the user invoking the command

**userName (%s)**

User name of the submitter

**resizeType (%d)**

Resize event type, 0, grow, 1 shrink.

**lastResizeTime(%d)**

The wall clock time when job allocation is changed previously. The first lastResizeTime is the job start time.

**numExecHosts (%d)**

The number of execution hosts before allocation is changed. Support LSF\_HPC\_EXTENSIONS="SHORT\_EVENTFILE".

**execHosts (%s)**

Execution host list before allocation is changed. Support LSF\_HPC\_EXTENSIONS="SHORT\_EVENTFILE".

**numResizeHosts (%d)**

Number of processors used for execution during resize. If LSF\_HPC\_EXTENSIONS="SHORT\_EVENTFILE" is specified in `lsf.conf`, the value of this field is the number of hosts listed in short format.

**resizeHosts (%s)**

List of execution host names during resize. If LSF\_HPC\_EXTENSIONS="SHORT\_EVENTFILE" is specified in `lsf.conf`, the value of this field is logged in a shortened format.

# lsb.applications

The `lsb.applications` file defines application profiles. Use application profiles to define common parameters for the same type of jobs, including the execution requirements of the applications, the resources they require, and how they should be run and managed.

This file is optional. Use the `DEFAULT_APPLICATION` parameter in `lsb.params` to specify a default application profile for all jobs. LSF does not automatically assign a default application profile.

This file is installed by default in `LSB_CONFDIR/cluster_name/confidir`.

## Changing lsb.applications configuration

After making any changes to `lsb.applications`, run `badminton reconfig` to reconfigure `mbatchd`. Configuration changes apply to pending jobs only. Running jobs are not affected.

## lsb.applications structure

Each application profile definition begins with the line `Begin Application` and ends with the line `End Application`. The application name must be specified. All other parameters are optional.

## Example

```
Begin Application
NAME = catia
DESCRIPTION = CATIA V5
CPULIMIT = 24:0/hostA # 24 hours of host hostA
FILELIMIT = 20000
DATALIMIT = 20000 # jobs data segment limit
CORELIMIT = 20000
PROCLIMIT = 5 # job processor limit
REQUEUE_EXIT_VALUES = 55 34 78
End Application
```

See the `lsb.applications` template file for additional application profile examples.

## ABS\_RUNLIMIT

### Syntax

**ABS\_RUNLIMIT=y | Y**

### Description

If set, absolute (wall-clock) run time is used instead of normalized run time for all jobs submitted with the following values:

- Run time limit specified by the `-W` option of `bsub`
- `RUNLIMIT` queue-level parameter in `lsb.queues`
- `RUNLIMIT` application-level parameter in `lsb.applications`
- `RUNTIME` parameter in `lsb.applications`

The runtime estimates and limits are not normalized by the host CPU factor.

## Default

Not defined. Run limit and runtime estimate are normalized.

# BIND\_JOB

## Syntax

**BIND\_JOB=NONE | BALANCE | PACK | ANY | USER | USER\_CPU\_LIST**

## Description

Specifies the processor binding policy for sequential and parallel job processes that run on a single host. On Linux execution hosts that support this feature, job processes are hard bound to selected processors.

If processor binding feature is not configured with the BIND\_JOB parameter in an application profile in `lsb.applications`, the `lsf.conf` configuration setting takes effect. The application profile configuration for processor binding overrides the `lsf.conf` configuration.

For backwards compatibility:

- BIND\_JOB=Y is interpreted as BIND\_JOB=BALANCE
- BIND\_JOB=N is interpreted as BIND\_JOB=NONE

## Supported platforms

Linux with kernel version 2.6 or higher

## Default

Not defined. Processor binding is disabled.

# CHKPNT\_DIR

## Syntax

**CHKPNT\_DIR=*chkpnt\_dir***

## Description

Specifies the checkpoint directory for automatic checkpointing for the application. To enable automatic checkpoint for the application profile, administrators must specify a checkpoint directory in the configuration of the application profile.

If CHKPNT\_PERIOD, CHKPNT\_INITPERIOD or CHKPNT\_METHOD was set in an application profile but CHKPNT\_DIR was not set, a warning message is issued and those settings are ignored.

The checkpoint directory is the directory where the checkpoint files are created. Specify an absolute path or a path relative to the current working directory for the job. Do not use environment variables in the directory path.

If checkpoint-related configuration is specified in both the queue and an application profile, the application profile setting overrides queue level configuration.

If checkpoint-related configuration is specified in the queue, application profile, and at job level:

- Application-level and job-level parameters are merged. If the same parameter is defined at both job-level and in the application profile, the job-level value overrides the application profile value.

- The merged result of job-level and application profile settings override queue-level configuration.

To enable checkpointing of MultiCluster jobs, define a checkpoint directory in an application profile (CHKPNT\_DIR, CHKPNT\_PERIOD, CHKPNT\_INITPERIOD, CHKPNT\_METHOD in `lsb.applications`) of both submission cluster and execution cluster. LSF uses the directory specified in the execution cluster.

Checkpointing is not supported if a job runs on a leased host.

The file path of the checkpoint directory can contain up to 4000 characters for UNIX and Linux, or up to 255 characters for Windows, including the directory and file name.

## Default

Not defined

# CHKPNT\_INITPERIOD

## Syntax

**CHKPNT\_INITPERIOD**=*init\_chkpt\_period*

## Description

Specifies the initial checkpoint period in minutes. CHKPNT\_DIR must be set in the application profile for this parameter to take effect. The periodic checkpoint specified by CHKPNT\_PERIOD does not happen until the initial period has elapsed.

Specify a positive integer.

Job-level command line values override the application profile configuration.

If administrators specify an initial checkpoint period and do not specify a checkpoint period (CHKPNT\_PERIOD), the job will only checkpoint once.

If the initial checkpoint period of a job is specified, and you run `bchkpnt` to checkpoint the job at a time before the initial checkpoint period, the initial checkpoint period is not changed by `bchkpnt`. The first automatic checkpoint still happens after the specified number of minutes.

## Default

Not defined

# CHKPNT\_PERIOD

## Syntax

**CHKPNT\_PERIOD**=*chkpnt\_period*

## Description

Specifies the checkpoint period for the application in minutes. CHKPNT\_DIR must be set in the application profile for this parameter to take effect. The running job is checkpointed automatically every checkpoint period.

Specify a positive integer.

Job-level command line values override the application profile and queue level configurations. Application profile level configuration overrides the queue level configuration.

## Default

Not defined

# CHKPNT\_METHOD

## Syntax

**CHKPNT\_METHOD**=*chkpnt\_method*

## Description

Specifies the checkpoint method. CHKPNT\_DIR must be set in the application profile for this parameter to take effect. Job-level command line values override the application profile configuration.

## Default

Not defined

# CHUNK\_JOB\_SIZE

## Syntax

**CHUNK\_JOB\_SIZE**=*integer*

## Description

Chunk jobs only. Allows jobs submitted to the same application profile to be chunked together and specifies the maximum number of jobs allowed to be dispatched together in a chunk. Specify a positive integer greater than or equal to 1.

All of the jobs in the chunk are scheduled and dispatched as a unit, rather than individually.

Specify **CHUNK\_JOB\_SIZE**=1 to disable job chunking for the application. This value overrides chunk job dispatch configured in the queue.

Use the **CHUNK\_JOB\_SIZE** parameter to configure application profiles that chunk small, short-running jobs. The ideal candidates for job chunking are jobs that have the same host and resource requirements and typically take 1 to 2 minutes to run.

The ideal candidates for job chunking are jobs that have the same host and resource requirements and typically take 1 to 2 minutes to run.

Job chunking can have the following advantages:

- Reduces communication between `sbat chd` and `mbat chd` and reduces scheduling overhead in `mbschd`.
- Increases job throughput in `mbat chd` and CPU utilization on the execution hosts.

However, throughput can deteriorate if the chunk job size is too big. Performance may decrease on profiles with **CHUNK\_JOB\_SIZE** greater than 30. You should evaluate the chunk job size on your own systems for best performance.

With MultiCluster job forwarding model, this parameter does not affect MultiCluster jobs that are forwarded to a remote cluster.

## Compatibility

This parameter is ignored and jobs are not chunked under the following conditions:

- CPU limit greater than 30 minutes (CPULIMIT parameter in lsb. queues or lsb. applications)
- Run limit greater than 30 minutes (RUNLIMIT parameter in lsb. queues or lsb. applications)
- Runtime estimate greater than 30 minutes (RUNTIME parameter in lsb. applications)

If CHUNK\_JOB\_DURATION is set in lsb. params, chunk jobs are accepted regardless of the value of CPULIMIT, RUNLIMIT or RUNTIME.

## Default

Not defined

# CORELIMIT

## Syntax

**CORELIMIT**=*integer*

## Description

The per-process (soft) core file size limit for all of the processes belonging to a job from this application profile (see `getrlimit(2)`). Application-level limits override any default limit specified in the queue, but must be less than the hard limit of the submission queue. Job-level core limit (`bsub -C`) overrides queue-level and application-level limits.

By default, the limit is specified in KB. Use `LSF_UNIT_FOR_LIMITS` in `lsf.conf` to specify a larger unit for the the limit (MB, GB, TB, PB, or EB).

## Default

Unlimited

# CPULIMIT

## Syntax

**CPULIMIT**=[*[hour:]minute[/host\_name | /host\_model]*

## Description

Normalized CPU time allowed for all processes of a job running in the application profile. The name of a host or host model specifies the CPU time normalization host to use.

Limits the total CPU time the job can use. This parameter is useful for preventing runaway jobs or jobs that use up too many resources.

When the total CPU time for the whole job has reached the limit, a SIGXCPU signal is sent to all processes belonging to the job. If the job has no signal handler for SIGXCPU, the job is killed immediately. If the SIGXCPU signal is handled, blocked, or ignored by the application, then after the grace period expires, LSF sends SIGINT, SIGTERM, and SIGKILL to the job to kill it.

If a job dynamically spawns processes, the CPU time used by these processes is accumulated over the life of the job.

Processes that exist for fewer than 30 seconds may be ignored.

By default, jobs submitted to the application profile without a job-level CPU limit (`bsub -c`) are killed when the CPU limit is reached. Application-level limits override any default limit specified in the queue.

The number of minutes may be greater than 59. For example, three and a half hours can be specified either as 3:30 or 210.

If no host or host model is given with the CPU time, LSF uses the default CPU time normalization host defined at the queue level (`DEFAULT_HOST_SPEC` in `lsb.queues`) if it has been configured, otherwise uses the default CPU time normalization host defined at the cluster level (`DEFAULT_HOST_SPEC` in `lsb.params`) if it has been configured, otherwise uses the host with the largest CPU factor (the fastest host in the cluster).

On Windows, a job that runs under a CPU time limit may exceed that limit by up to `SBD_SLEEP_TIME`. This is because `sbat chd` periodically checks if the limit has been exceeded.

On UNIX systems, the CPU limit can be enforced by the operating system at the process level.

You can define whether the CPU limit is a per-process limit enforced by the OS or a per-job limit enforced by LSF with `LSB_JOB_CPULIMIT` in `lsf.conf`.

## Default

Unlimited

# DATALIMIT

## Syntax

**DATALIMIT**=*integer*

## Description

The per-process (soft) data segment size limit (in KB) for all of the processes belonging to a job running in the application profile (see `getrlimit(2)`).

By default, jobs submitted to the application profile without a job-level data limit (`bsub -D`) are killed when the data limit is reached. Application-level limits override any default limit specified in the queue, but must be less than the hard limit of the submission queue.

## Default

Unlimited

# DESCRIPTION

## Syntax

**DESCRIPTION**=*text*

## Description

Description of the application profile. The description is displayed by `bapp -l`.

The description should clearly describe the service features of the application profile to help users select the proper profile for each job.

The text can include any characters, including white space. The text can be extended to multiple lines by ending the preceding line with a backslash (`\`). The maximum length for the text is 512 characters.

## DJOB\_COMMFAIL\_ACTION

### Syntax

**DJOB\_COMMFAIL\_ACTION="KILL\_TASKS"**

### Description

Defines the action LSF should take if it detects a communication failure with one or more remote parallel or distributed tasks. If defined, LSF tries to kill all the current tasks of a parallel or distributed job associated with the communication failure. If not defined, LSF terminates all tasks and shuts down the entire job.

This parameter only applies to the bl aunch distributed application framework.

When defined in an application profile, the LSB\_DJOB\_COMMFAIL\_ACTION variable is set when running `bsub -app` for the specified application.

### Default

Not defined. Terminate all tasks, and shut down the entire job.

## DJOB\_DISABLED

### Syntax

**DJOB\_DISABLED=Y | N**

### Description

Disables the bl aunch distributed application framework.

### Default

Not defined. Distributed application framework is enabled.

## DJOB\_ENV\_SCRIPT

### Syntax

**DJOB\_ENV\_SCRIPT=*script\_name***

### Description

Defines the name of a user-defined script for setting and cleaning up the parallel or distributed job environment.

The specified script must support a `setup` argument and a `cleanup` argument. The script is executed by LSF with the `setup` argument before launching a parallel or distributed job, and with argument `cleanup` after the job is finished.

The script runs as the user, and is part of the job.

If a full path is specified, LSF uses the path name for the execution. Otherwise, LSF looks for the executable from `$LSF_BINDIR`.

This parameter only applies to the bl aunch distributed application framework.

When defined in an application profile, the `LSB_DJOB_ENV_SCRIPT` variable is set when running `bsub -app` for the specified application.

The command path can contain up to 4094 characters for UNIX and Linux, or up to 255 characters for Windows, including the directory, file name, and expanded values for `%J` (*job\_ID*) and `%I` (*index\_ID*).

## Default

Not defined.

# DJOB\_HB\_INTERVAL

## Syntax

**DJOB\_HB\_INTERVAL=seconds**

## Description

Value in seconds used to calculate the heartbeat interval between the task RES and job RES of a parallel or distributed job.

This parameter only applies to the `blaunch` distributed application framework.

When `DJOB_HB_INTERVAL` is specified, the interval is scaled according to the number of tasks in the job:

$\max(\text{DJOB\_HB\_INTERVAL}, 10) + \text{host\_factor}$

where

$\text{host\_factor} = 0.01 * \text{number of hosts allocated for the job}$

## Default

Not defined. Interval is equal to `SBD_SLEEP_TIME` in `lsb.params`, where the default value of `SBD_SLEEP_TIME` is 30 seconds.

# DJOB\_RESIZE\_GRACE\_PERIOD

## Syntax

**DJOB\_RESIZE\_GRACE\_PERIOD = seconds**

## Description

When a resizable job releases resources, the LSF distributed parallel job framework terminates running tasks if a host has been completely removed. A `DJOB_RESIZE_GRACE_PERIOD` defines a grace period in seconds for the application to clean up tasks itself before LSF forcibly terminates them.

## Default

No grace period.

# DJOB\_RU\_INTERVAL

## Syntax

**DJOB\_RU\_INTERVAL=seconds**

## Description

Value in seconds used to calculate the resource usage update interval for the tasks of a parallel or distributed job.

This parameter only applies to the bl aunch distributed application framework.

When DJOB\_RU\_INTERVAL is specified, the interval is scaled according to the number of tasks in the job:

$$\max(\text{DJOB\_RU\_INTERVAL}, 10) + \text{host\_factor}$$

where

$$\text{host\_factor} = 0.01 * \text{number of hosts allocated for the job}$$

## Default

Not defined. Interval is equal to SBD\_SLEEP\_TIME in l sb. params, where the default value of SBD\_SLEEP\_TIME is 30 seconds.

# JOB\_INCLUDE\_POSTPROC

## Syntax

**JOB\_INCLUDE\_POSTPROC=Y | N**

## Description

Specifies whether LSF includes the post-execution processing of the job as part of the job. When set to Y:

- Prevents a new job from starting on a host until post-execution processing is finished on that host
- Includes the CPU and run times of post-execution processing with the job CPU and run times
- sbat chd sends both job finish status (DONE or EXIT) and post-execution processing status (POST\_DONE or POST\_ERR) to mbat chd at the same time

The variable LSB\_JOB\_INCLUDE\_POSTPROC in the user environment overrides the value of JOB\_INCLUDE\_POSTPROC in an application profile in l sb. appl i cat i ons.

JOB\_INCLUDE\_POSTPROC in an application profile in l sb. appl i cat i ons overrides the value of JOB\_INCLUDE\_POSTPROC in l sb. params.

For SGI cpusets, if JOB\_INCLUDE\_POSTPROC=Y, LSF does not release the cpuset until post-execution processing has finished, even though post-execution processes are not attached to the cpuset.

## Default

N. Post-execution processing is not included as part of the job, and a new job can start on the execution host before post-execution processing finishes.

# JOB\_POSTPROC\_TIMEOUT

## Syntax

**JOB\_POSTPROC\_TIMEOUT=*minutes***

## Description

Specifies a timeout in minutes for job post-execution processing. The specified timeout must be greater than zero

If post-execution processing takes longer than the timeout, `sbatchd` reports that post-execution has failed (POST\_ERR status), and kills the process group of the job's post-execution processes. Only the parent process of the post-execution command is killed when the timeout expires. The child processes of the post-execution command are not killed.

If `JOB_INCLUDE_POSTPROC=Y`, and `sbatchd` kills the post-execution processes because the timeout has been reached, the CPU time of the post-execution processing is set to 0, and the job's CPU time does not include the CPU time of post-execution processing.

`JOB_POSTPROC_TIMEOUT` defined in an application profile in `lsb.applications` overrides the value in `lsb.params`. `JOB_POSTPROC_TIMEOUT` cannot be defined in user environment.

## Default

Not defined. Post-execution processing does not time out.

# FILELIMIT

## Syntax

**FILELIMIT**=*integer*

## Description

The per-process (soft) file size limit (in KB) for all of the processes belonging to a job running in the application profile (see `getrlimit(2)`). Application-level limits override any default limit specified in the queue, but must be less than the hard limit of the submission queue.

## Default

Unlimited

# JOB\_STARTER

## Syntax

**JOB\_STARTER**=*starter* [*starter*] ["%USRCMD"] [*starter*]

## Description

Creates a specific environment for submitted jobs prior to execution. An application-level job starter overrides a queue-level job starter.

*starter* is any executable that can be used to start the job (i.e., can accept the job as an input argument). Optionally, additional strings can be specified.

By default, the user commands run after the job starter. A special string, `%USRCMD`, can be used to represent the position of the user's job in the job starter command line. The `%USRCMD` string and any additional commands must be enclosed in quotation marks (" ").

## Example

```
JOB_STARTER=csh -c "%USRCMD; sleep 10"
```

In this case, if a user submits a job

```
bsub myjob arguments
```

the command that actually runs is:  
`csh -c "myjob arguments; sleep 10"`

## Default

Not defined. No job starter is used,

# LOCAL\_MAX\_PREEEXEC\_RETRY

## Syntax

**LOCAL\_MAX\_PREEEXEC\_RETRY**=*integer*

## Description

The maximum number of times to attempt the pre-execution command of a job on the local cluster.

## Valid values

$0 < \text{MAX\_PREEEXEC\_RETRY} < \text{INFINIT\_INT}$   
 INFINIT\_INT is defined in `lsf.h`.

## Default

Not defined. The number of preexec retry times is unlimited

# MAX\_JOB\_PREEMPT

## Syntax

**MAX\_JOB\_PREEMPT**=*integer*

## Description

The maximum number of times a job can be preempted. Applies to queue-based preemption only.

## Valid values

$0 < \text{MAX\_JOB\_PREEMPT} < \text{INFINIT\_INT}$   
 INFINIT\_INT is defined in `lsf.h`.

## Default

Not defined. The number of preemption times is unlimited.

# MAX\_JOB\_REQUEUE

## Syntax

**MAX\_JOB\_REQUEUE**=*integer*

## Description

The maximum number of times to requeue a job automatically.

## Valid values

$0 < \text{MAX\_JOB\_REQUEUE} < \text{INFINIT\_INT}$

`INFINIT_INT` is defined in `lsf.h`.

## Default

Not defined. The number of requeue times is unlimited

# MAX\_PREEEXEC\_RETRY

## Syntax

`MAX_PREEEXEC_RETRY=integer`

## Description

Use `REMOTE_MAX_PREEEXEC_RETRY` instead. This parameter is only maintained for backwards compatibility.

MultiCluster job forwarding model only. The maximum number of times to attempt the pre-execution command of a job from a remote cluster.

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

## Valid values

$0 < \text{MAX\_PREEEXEC\_RETRY} < \text{INFINIT\_INT}$

`INFINIT_INT` is defined in `lsf.h`.

## Default

5

# MAX\_TOTAL\_TIME\_PREEMPT

## Syntax

`MAX_TOTAL_TIME_PREEMPT=integer`

## Description

The accumulated preemption time in minutes after which a job cannot be preempted again, where *minutes* is wall-clock time, not normalized time.

Setting this parameter in `lsb.applications` overrides the parameter of the same name in `lsb.queues` and in `lsb.params`.

## Valid values

Any positive integer greater than or equal to one (1)

## Default

Unlimited

# MEMLIMIT

## Syntax

**MEMLIMIT**=*integer*

## Description

The per-process (soft) process resident set size limit for all of the processes belonging to a job running in the application profile.

Sets the maximum amount of physical memory (resident set size, RSS) that may be allocated to a process.

By default, the limit is specified in KB. Use `LSF_UNIT_FOR_LIMITS` in `lsf.conf` to specify a larger unit for the limit (MB, GB, TB, PB, or EB).

By default, jobs submitted to the application profile without a job-level memory limit are killed when the memory limit is reached. Application-level limits override any default limit specified in the queue, but must be less than the hard limit of the submission queue.

LSF has two methods of enforcing memory usage:

- OS Memory Limit Enforcement
- LSF Memory Limit Enforcement

## OS memory limit enforcement

OS memory limit enforcement is the default MEMLIMIT behavior and does not require further configuration. OS enforcement usually allows the process to eventually run to completion. LSF passes MEMLIMIT to the OS, which uses it as a guide for the system scheduler and memory allocator. The system may allocate more memory to a process if there is a surplus. When memory is low, the system takes memory from and lowers the scheduling priority (re-nice) of a process that has exceeded its declared MEMLIMIT. Only available on systems that support `RLIMIT_RSS` for `setrlimit()`.

Not supported on:

- Sun Solaris 2.x
- Windows

## LSF memory limit enforcement

To enable LSF memory limit enforcement, set `LSB_MEMLIMIT_ENFORCE` in `lsf.conf` to `y`. LSF memory limit enforcement explicitly sends a signal to kill a running process once it has allocated memory past MEMLIMIT.

You can also enable LSF memory limit enforcement by setting `LSB_JOB_MEMLIMIT` in `lsf.conf` to `y`. The difference between `LSB_JOB_MEMLIMIT` set to `y` and `LSB_MEMLIMIT_ENFORCE` set to `y` is that with `LSB_JOB_MEMLIMIT`, only the per-job memory limit enforced by LSF is enabled. The per-process memory limit enforced by the OS is disabled. With `LSB_MEMLIMIT_ENFORCE` set to `y`, both the per-job memory limit enforced by LSF and the per-process memory limit enforced by the OS are enabled.

Available for all systems on which LSF collects total memory usage.

## Default

Unlimited

# MEMLIMIT\_TYPE

## Syntax

**MEMLIMIT\_TYPE=JOB** [PROCESS] [TASK]

**MEMLIMIT\_TYPE=PROCESS** [JOB] [TASK]

**MEMLIMIT\_TYPE=TASK** [PROCESS] [JOB]

## Description

A memory limit is the maximum amount of memory a job is allowed to consume. Jobs that exceed the level are killed. You can specify different types of memory limits to enforce. Use any combination of JOB, PROCESS, and TASK.

By specifying a value in the application profile, you overwrite these three parameters: LSB\_JOB\_MEMLIMIT, LSB\_MEMLIMIT\_ENFORCE, LSF\_HPC\_EXTENSIONS (TASK\_MEMLIMIT).

---

### Note:

A task list is a list in LSF that keeps track of the default resource requirements for different applications and task eligibility for remote execution.

---

- **PROCESS:** Applies a memory limit by OS process, which is enforced by the OS on the slave machine (where the job is running). When the memory allocated to one process of the job exceeds the memory limit, LSF kills the job.
- **TASK:** Applies a memory limit based on the task list file. It is enforced by LSF. LSF terminates the entire parallel job if any single task exceeds the limit setting for memory and swap limits.
- **JOB:** Applies a memory limit identified in a job and enforced by LSF. When the sum of the memory allocated to all processes of the job exceeds the memory limit, LSF kills the job.
- **PROCESS TASK:** Enables both process-level memory limit enforced by OS and task-level memory limit enforced by LSF.
- **PROCESS JOB:** Enables both process-level memory limit enforced by OS and job-level memory limit enforced by LSF.
- **TASK JOB:** Enables both task-level memory limit enforced by LSF and job-level memory limit enforced by LSF.
- **PROCESS TASK JOB:** Enables process-level memory limit enforced by OS, task-level memory limit enforced by LSF, and job-level memory limit enforced by LSF.

## Default

Not defined. The memory limit-level is still controlled by LSF\_HPC\_EXTENSIONS=TASK\_MEMLIMIT, LSB\_JOB\_MEMLIMIT, LSB\_MEMLIMIT\_ENFORCE

## MIG

### Syntax

**MIG=minutes**

## Description

Enables automatic job migration and specifies the migration threshold for checkpointable or rerunnable jobs, in minutes.

LSF automatically migrates jobs that have been in the SSUSP state for more than the specified number of minutes. A value of 0 specifies that a suspended job is migrated immediately. The migration threshold applies to all jobs running on the host.

Job-level command line migration threshold overrides threshold configuration in application profile and queue. Application profile configuration overrides queue level configuration.

When a host migration threshold is specified, and is lower than the value for the job, the queue, or the application, the host value is used.

Members of a chunk job can be migrated. Chunk jobs in WAIT state are removed from the job chunk and put into PEND state.

Does not affect MultiCluster jobs that are forwarded to a remote cluster.

## Default

Not defined. LSF does not migrate checkpointable or rerunnable jobs automatically.

## NAME

## Syntax

**NAME**=*string*

## Description

*Required.* Unique name for the application profile.

Specify any ASCII string up to 60 characters long. You can use letters, digits, underscores (\_), dashes (-), periods (.) or spaces in the name. The application profile name must be unique within the cluster.

---

### Note:

If you want to specify the ApplicationVersion in a JSDL file, include the version when you define the application profile name. Separate the name and version by a space, as shown in the following example:

```
NAME=myapp 1.0
```

## Default

You must specify this parameter to define an application profile. LSF does not automatically assign a default application profile name.

## NICE

## Syntax

**NICE**=*integer*

## Description

Adjusts the UNIX scheduling priority at which jobs from the application execute.

A value of 0 (zero) maintains the default scheduling priority for UNIX interactive jobs. This value adjusts the run-time priorities for batch jobs to control their effect on other batch or interactive jobs. See the `nic(1)` manual page for more details.

On Windows, this value is mapped to Windows process priority classes as follows:

- `nic>=0` corresponds to a priority class of IDLE
- `nic<0` corresponds to a priority class of NORMAL

Platform LSF on Windows does not support HIGH or REAL-TIME priority classes.

When set, this value overrides NICE set at the queue level in `lsb.queues`.

## Default

Not defined.

# NO\_PREEMPT\_INTERVAL

## Syntax

**NO\_PREEMPT\_INTERVAL**=*minutes*

## Description

Prevents preemption of jobs for the specified number of minutes of uninterrupted run time, where *minutes* is wall-clock time, not normalized time. **NO\_PREEMPT\_INTERVAL=0** allows immediate preemption of jobs as soon as they start or resume running.

Setting this parameter in `lsb.appl icat ions` overrides the parameter of the same name in `lsb.queues` and in `lsb.params`.

## Default

0

# NO\_PREEMPT\_FINISH\_TIME

## Syntax

**NO\_PREEMPT\_FINISH\_TIME**=*minutes* | *percentage*

## Description

Prevents preemption of jobs that will finish within the specified number of minutes or the specified percentage of the estimated run time or run limit.

Specifies that jobs due to finish within the specified number of minutes or percentage of job duration should not be preempted, where *minutes* is wall-clock time, not normalized time. Percentage must be greater than 0 or less than 100% (between 1% and 99%).

For example, if the job run limit is 60 minutes and **NO\_PREEMPT\_FINISH\_TIME=10%**, the job cannot be preempted after it runs 54 minutes or longer.

If you specify percentage for **NO\_PREEMPT\_FINISH\_TIME**, requires a run time (`bsub -We` or **RUNTIME** in `lsb.appl icat ions`), or run limit to be specified for the job (`bsub -W`, or **RUNLIMIT** in `lsb.queues`, or **RUNLIMIT** in `lsb.appl icat ions`)

## NO\_PREEMPT\_RUN\_TIME

### Syntax

**NO\_PREEMPT\_RUN\_TIME**=*minutes* | *percentage*

### Description

Prevents preemption of jobs that have been running for the specified number of minutes or the specified percentage of the estimated run time or run limit.

Specifies that jobs that have been running for the specified number of minutes or longer should not be preempted, where *minutes* is wall-clock time, not normalized time. Percentage must be greater than 0 or less than 100% (between 1% and 99%).

For example, if the job run limit is 60 minutes and **NO\_PREEMPT\_RUN\_TIME**=50%, the job cannot be preempted after it running 30 minutes or longer.

If you specify percentage for **NO\_PREEMPT\_RUN\_TIME**, requires a run time (`bsub -We` or `RUNTIME` in `lsb. applications`), or run limit to be specified for the job (`bsub -W`, or `RUNLIMIT` in `lsb. queues`, or `RUNLIMIT` in `lsb. applications`)

## PERSISTENT\_HOST\_ORDER

### Syntax

**PERSISTENT\_HOST\_ORDER**=Y | yes | N | no

### Description

Applies when migrating parallel jobs in a multicluster environment. Setting **PERSISTENT\_HOST\_ORDER**=Y ensures that jobs are restarted on hosts based on alphabetical names of the hosts, preventing them from being restarted on the same hosts that they ran on before migration.

### Default

**PERSISTENT\_HOST\_ORDER**=N. Migrated jobs in a multicluster environment could run on the same hosts that they ran on before.

## POST\_EXEC

### Syntax

**POST\_EXEC**=*command*

### Description

Enables post-execution processing at the application level. The **POST\_EXEC** command runs on the execution host after the job finishes. Post-execution commands can be configured at the job, application, and queue levels.

If both application-level (`POST_EXEC` in `lsb. applications`) and job-level post-execution commands are specified, job level post-execution overrides application-level post-execution commands. Queue-level post-execution commands (`POST_EXEC` in `lsb. queues`) run after application-level post-execution and job-level post-execution commands.

The `POST_EXEC` command uses the same environment variable values as the job, and runs under the user account of the user who submits the job.

When a job exits with one of the application profile's `REQUEUE_EXIT_VALUES`, LSF requeues the job and sets the environment variable `LSB_JOBPEND`. The post-execution command runs after the requeued job finishes.

When the post-execution command is run, the environment variable `LSB_JOBEXIT_STAT` is set to the exit status of the job. If the execution environment for the job cannot be set up, `LSB_JOBEXIT_STAT` is set to 0 (zero).

The command path can contain up to 4094 characters for UNIX and Linux, or up to 255 characters for Windows, including the directory, file name, and expanded values for `%J` (*job\_ID*) and `%I` (*index\_ID*).

For UNIX:

- The pre- and post-execution commands run in the `/tmp` directory under `/bin/sh -c`, which allows the use of shell features in the commands. The following example shows valid configuration lines:  

```
PRE_EXEC= /usr/share/lsf/misc/testq_pre >> /tmp/pre.out
```

```
POST_EXEC= /usr/share/lsf/misc/testq_post | grep -v "Hey!"
```
- LSF sets the `PATH` environment variable to  

```
PATH=' /bin /usr/bin /sbin /usr/sbin'
```
- The `stdin`, `stdout`, and `stderr` are set to `/dev/null`
- To allow UNIX users to define their own post-execution commands, an LSF administrator specifies the environment variable `$USER_POSTEXEC` as the `POST_EXEC` command. A user then defines the post-execution command:  

```
setenv USER_POSTEXEC /path_name
```

---

**Note:**

The path name for the post-execution command must be an absolute path.

---

For Windows:

- The pre- and post-execution commands run under `cmd.exe /c`
- The standard input, standard output, and standard error are set to `NULL`
- The `PATH` is determined by the setup of the LSF Service

---

**Note:**

For post-execution commands that execute on a Windows Server 2003, x64 Edition platform, users must have read and execute privileges for `cmd.exe`.

---

## Default

Not defined. No post-execution commands are associated with the application profile.

## PRE\_EXEC

### Syntax

**PRE\_EXEC=***command*

## Description

Enables pre-execution processing at the application level. The `PRE_EXEC` command runs on the execution host before the job starts. If the `PRE_EXEC` command exits with a non-zero exit code, LSF requeues the job to the front of the queue.

Pre-execution commands can be configured at the application, queue, and job levels and run in the following order:

1. The queue-level command
2. The application-level or job-level command. If you specify a command at both the application and job levels, the job-level command overrides the application-level command; the application-level command is ignored.

The `PRE_EXEC` command uses the same environment variable values as the job, and runs under the user account of the user who submits the job.

The command path can contain up to 4094 characters for UNIX and Linux, or up to 255 characters for Windows, including the directory, file name, and expanded values for `%J` (*job\_ID*) and `%I` (*index\_ID*).

For UNIX:

- The pre- and post-execution commands run in the `/tmp` directory under `/bin/sh -c`, which allows the use of shell features in the commands. The following example shows valid configuration lines:
 

```
PRE_EXEC= /usr/share/lsf/misc/testq_pre >> /tmp/pre.out
POST_EXEC= /usr/share/lsf/misc/testq_post | grep -v "Hey!"
```
- LSF sets the `PATH` environment variable to
 

```
PATH= '/bin /usr/bin /sbin /usr/sbin'
```
- The `stdin`, `stdout`, and `stderr` are set to `/dev/null`

For Windows:

- The pre- and post-execution commands run under `cmd. exe /c`
- The standard input, standard output, and standard error are set to `NULL`
- The `PATH` is determined by the setup of the LSF Service

---

### Note:

For pre-execution commands that execute on a Windows Server 2003, x64 Edition platform, users must have read and execute privileges for `cmd. exe`.

---

## Default

Not defined. No pre-execution commands are associated with the application profile.

# PROCESSLIMIT

## Syntax

`PROCESSLIMIT=integer`

## Description

Limits the number of concurrent processes that can be part of a job.

By default, jobs submitted to the application profile without a job-level process limit are killed when the process limit is reached. Application-level limits override any default limit specified in the queue.

SIGINT, SIGTERM, and SIGKILL are sent to the job in sequence when the limit is reached.

## Default

Unlimited

# PROCLIMIT

## Syntax

**PROCLIMIT**=[*minimum\_limit*] [*default\_limit*] *maximum\_limit*

## Description

Maximum number of slots that can be allocated to a job. For parallel jobs, the maximum number of processors that can be allocated to the job.

Optionally specifies the minimum and default number of job slots. All limits must be positive integers greater than or equal to 1 that satisfy the following relationship:

$1 \leq \textit{minimum} \leq \textit{default} \leq \textit{maximum}$

Job-level processor limits (bsub -n) override application-level PROCLIMIT, which overrides queue-level PROCLIMIT. Job-level limits must fall within the maximum and minimum limits of the application profile and the queue.

You can specify up to three limits in the PROCLIMIT parameter:

- One limit—Is the maximum processor limit. The minimum and default limits are set to 1.
- Two limits—The first is the minimum processor limit, and the second one is the maximum. The default is set equal to the minimum. The minimum must be less than or equal to the maximum.
- Three limits—The first is the minimum processor limit, the second is the default processor limit, and the third is the maximum. The minimum must be less than the default and the maximum.

Jobs that request fewer slots than the minimum PROCLIMIT or more slots than the maximum PROCLIMIT cannot use the application profile and are rejected. If the job requests minimum and maximum job slots, the maximum slots requested cannot be less than the minimum PROCLIMIT, and the minimum slots requested cannot be more than the maximum PROCLIMIT.

## Default

Unlimited, the default number of slots is 1

# REMOTE\_MAX\_PREEEXEC\_RETRY

## Syntax

**REMOTE\_MAX\_PREEEXEC\_RETRY**=*integer*

## Description

MultiCluster job forwarding model only. The maximum number of times to attempt the pre-execution command of a job from a remote cluster.

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

## Valid values

up to INFINIT\_INT defined in lsf.h.

## Default

5

# QUEUE\_EXIT\_VALUES

## Syntax

**QUEUE\_EXIT\_VALUES**=[*exit\_code* ...] [EXCLUDE(*exit\_code* ...)]

## Description

Enables automatic job requeue and sets the LSB\_EXIT\_QUEUE environment variable. Use spaces to separate multiple exit code values. Application-level exit values override queue-level values. Job-level exit values (bsub -Q) override application-level and queue-level values.

*exit\_code* has the following form:

```
"[all] [~number ...] | [number ...]"
```

The reserved keyword `all` specifies all exit codes. Exit codes are typically between 0 and 255. Use a tilde (~) to exclude specified exit codes from the list.

Jobs running the same applications generally shared the same exit values under the same conditions. Setting QUEUE\_EXIT\_VALUES in an application profile instead of in the queue allows different applications with different exit values to share the same queue.

Jobs are requeued to the head of the queue. The output from the failed run is not saved, and the user is not notified by LSF.

Define an exit code as EXCLUDE(*exit\_code*) to enable exclusive job requeue. Exclusive job requeue does not work for parallel jobs.

If `mbat chd` is restarted, it does not remember the previous hosts from which the job exited with an exclusive requeue exit code. In this situation, it is possible for a job to be dispatched to hosts on which the job has previously exited with an exclusive exit code.

## Example

```
QUEUE_EXIT_VALUES=30 EXCLUDE(20)
```

means that jobs with exit code 30 are requeued, jobs with exit code 20 are requeued exclusively, and jobs with any other exit code are not requeued.

## Default

Not defined. Jobs in the application profile are not requeued.

# RERUNNABLE

## Syntax

**RERUNNABLE**=yes | no

## Description

If `yes`, enables automatic job rerun (restart) for any job associated with the application profile.

Rerun is disabled when `RERUNNABLE` is set to `no`. The `yes` and `no` arguments are not case-sensitive.

Members of a chunk job can be rerunnable. If the execution host becomes unavailable, rerunnable chunk job members are removed from the job chunk and dispatched to a different execution host.

Job level rerun (`bsub -r`) overrides the `RERUNNABLE` value specified in the application profile, which overrides the queue specification. `bmod -rn` to make rerunnable jobs non-rerunnable overrides both the application profile and the queue.

## Default

Not defined.

## RES\_REQ

## Syntax

**RES\_REQ**=*res\_req*

## Description

Resource requirements used to determine eligible hosts. Specify a resource requirement string as usual. The resource requirement string lets you specify conditions in a more flexible manner than using the load thresholds.

Resource requirement strings can be simple (applying to the entire job) or compound (applying to the specified number of slots). When a compound resource requirement is set at the application-level, it will be ignored if any job-level resource requirements (simple or compound) are defined.

In the event no job-level resource requirements are set, the compound application-level requirements interact with queue resource requirement strings in the following ways:

In the event no job-level resource requirements are set, the compound application-level requirements interact with queue-level resource requirement strings in the following ways:

- If no queue-level resource requirement is defined or a compound queue-level resource requirement is defined, the compound application-level requirement is used.
- If a simple queue-level requirement is defined, the application-level and queue-level requirements combine as follows:

section	compound application and simple queue behavior
select	both levels satisfied; queue requirement applies to all compound terms
same	queue level ignored
order span	application-level section overwrites queue-level section (if a given level is present); queue requirement (if used) applies to all compound terms

section	compound application and simple queue behavior
rusage	<ul style="list-style-type: none"> <li>• both levels merge</li> <li>• queue requirement if a job-based resource is applied to the first compound term, otherwise applies to all compound terms</li> <li>• if conflicts occur the application-level section overwrites the queue-level section.</li> </ul> <p>For example: if the application-level requirement is <math>\text{num1}*\{\text{rusage}[\text{R1}]\} + \text{num2}*\{\text{rusage}[\text{R2}]\}</math> and the queue-level requirement is <math>\text{rusage}[\text{RQ}]</math> where <math>\text{RQ}</math> is a job resource, the merged requirement is <math>\text{num1}*\{\text{rusage}[\text{merge}(\text{R1}, \text{RQ})]\} + \text{num2}*\{\text{rusage}[\text{R2}]\}</math></p>

The following resource requirement sections are supported:

- select
- rusage
- order
- span
- same
- cu

Compound resource requirements do not support the `cu` section, multiple `-R` options, or the `||` operator within the `rusage` section.

Multiple `-R` strings cannot be used with multi-phase `rusage` resource requirements.

For internal load indices and duration, jobs are rejected if they specify resource reservation requirements at the job or application level that exceed the requirements specified in the queue.

If `RES_REQ` is defined at the queue level and there are no load thresholds defined, the pending reasons for each individual load index are not be displayed by `bj obs`.

By default, memory (`mem`) and swap (`swp`) limits in `select [ ]` and `rusage [ ]` sections are specified in MB. Use `LSF_UNIT_FOR_LIMITS` in `lsf.conf` to specify a larger unit for these limits (GB, TB, PB, or EB).

When `LSF_STRICT_RESREQ=Y` is configured in `lsf.conf`, resource requirement strings in `select` sections must conform to a more strict syntax. The strict resource requirement syntax only applies to the `select` section. It does not apply to the other resource requirement sections (`order`, `rusage`, `same`, `span`, or `cu`). When `LSF_STRICT_RESREQ=Y` in `lsf.conf`, LSF rejects resource requirement strings where an `rusage` section contains a non-consumable resource.

## select section

For simple resource requirements, the `select` section defined at the application, queue, and job level must all be satisfied.

## rusage section

The `rusage` section can specify additional requests. To do this, use the OR (`|`) operator to separate additional `rusage` strings. The job-level `rusage` section takes precedence.

---

### Note:

Compound resource requirements do not support use of the `||` operator within the component `rusage` simple resource requirements. Multiple

rusage strings cannot be used with multi-phase rusage resource requirements.

When both job-level and application-level rusage sections are defined using simple resource requirement strings, the rusage section defined for the job overrides the rusage section defined in the application profile. The rusage definitions are merged, with the job-level rusage taking precedence. Any queue-level requirements are then merged with that result.

For example:

Application-level RES_REQ:	RES_REQ=rusage[mem=200:lic=1] ...	For the job submission:	bsub -R' rusage[mem=100]' ...
		the resulting requirement for the job is	rusage[mem=100:lic=1]
		where mem=100 specified by the job overrides mem=200 specified by the application profile. However, lic=1 from application profile is kept, since job does not specify it.	
Application-level RES_REQ with decay and duration defined:	RES_REQ=rusage[mem=200:duration=20:decay=1] ...	For a job submission with no decay or duration:	bsub -R' rusage[mem=100]' ...
		the resulting requirement for the job is:	rusage[mem=100:duration=20:decay=1]
		Application-level duration and decay are merged with the job-level specification, and mem=100 for the job overrides mem=200 specified by the application profile. However, duration=20 and decay=1 from application profile are kept, since job does not specify them.	
Application-level RES_REQ with multi-phase job- level rusage:	RES_REQ=rusage[mem=(200 150):duration=(10 10):decay=(1), swap=100] ...	For a multi-phase job submission:	bsub -app <i>app_name</i> -R' rusage[mem=(600 350):duration=(20 10):decay=(0 1)]' ...
		the resulting requirement for the job is:	rusage[mem=(600 350):duration=(20 10):decay=(0 1), swap=100]
		The job-level values for mem, duration and decay override the application-level values. However, swap=100 from the application profile is kept, since the job does not specify swap.	
Application-level RES_REQ with multi-phase application-level rusage:	RES_REQ=rusage[mem=(200 150):duration=(10 10):decay=(1)] ...	For a job submission:	bsub -app <i>app_name</i> -R' rusage[mem=200:duration=15:decay=0]' ...
		the resulting requirement for the job is:	rusage[mem=200:duration=15:decay=0]
		Job-level values override the application-level multi-phase rusage string.	

---

**Note:**

The merged application-level and job-level `rusage` consumable resource requirements must satisfy any limits set by the parameter `RESRSV_LIMIT` in `lsb.queues`, or the job will be rejected.

---

## order section

For simple resource requirements the `order` section defined at the job-level overrides any application-level `order` section. An application-level `order` section overrides queue-level specification. The `order` section defined at the application level is ignored if any resource requirements are specified at the job level. If the no resource requirements include an `order` section, the default `order r15s: pg` is used.

## span section

For simple resource requirements the `span` section defined at the job-level overrides an application-level `span` section, which overrides a queue-level `span` section.

---

**Note:**

Define `span[hosts=-1]` in the application profile or in `bsub -R` resource requirement string to disable the `span` section setting in the queue.

---

## same section

For simple resource requirements all `same` sections defined at the job-level, application-level, and queue-level are combined before the job is dispatched.

## cu section

For simple resource requirements the job-level `cu` section overwrites the application-level, and the application-level `cu` section overwrites the queue-level.

## Default

```
select [ type==local ] order [ r15s: pg ]
```

If this parameter is defined and a host model or Boolean resource is specified, the default type is `any`.

# RESIZABLE\_JOBS

## Syntax

```
RESIZABLE_JOBS = [Y|N|auto]
```

## Description

`N|n`: The resizable job feature is disabled in the application profile. Under this setting, all jobs attached to this application profile are not resizable. All `brsize` and `bsub -ar` commands will be rejected with a proper error message.

`Y|y`: Resize is enabled in the application profile and all jobs belonging to the application are resizable by default. Under this setting, users can run `brsize` commands to cancel pending resource allocation

requests for the job or release resources from an existing job allocation, or use `bsub` to submit an autoresizable job.

`auto`: All jobs belonging to the application will be autoresizable.

Resizable jobs must be submitted with an application profile that defines `RESIZABLE_JOBS` as either `auto` or `Y`. If application defines `RESIZABLE_JOBS=auto`, but administrator changes it to `N` and reconfigures LSF, jobs without job-level auto resizable attribute become not autoresizable. For running jobs that are in the middle of notification stage, LSF lets current notification complete and stops scheduling. Changing `RESIZABLE_JOBS` configuration does not affect jobs with job-level autoresizable attribute. (This behavior is same as exclusive job, `bsub -x` and `EXCLUSIVE` parameter in queue level.)

Auto-resizable jobs cannot be submitted with compute unit resource requirements. In the event a `bswitch` call or queue reconfiguration results in an auto-resizable job running in a queue with compute unit resource requirements, the job will no longer be auto-resizable.

Resizable jobs cannot have compound resource requirements.

## Default

If the parameter is undefined, the default value is `N`.

# RESIZE\_NOTIFY\_CMD

## Syntax

`RESIZE_NOTIFY_CMD = notification_command`

## Description

Defines an executable command to be invoked on the first execution host of a job when a resize event occurs. The maximum length of notification command is 4 KB.

## Default

Not defined. No resize notification command is invoked.

# RESUME\_CONTROL

## Syntax

`RESUME_CONTROL=signal | command`

---

### Remember:

Unlike the `JOB_CONTROLS` parameter in `lsb.queues`, the `RESUME_CONTROL` parameter does not require square brackets (`[ ]`) around the action.

---

- *signal* is a UNIX signal name. The specified signal is sent to the job. The same set of signals is not supported on all UNIX systems. To display a list of the symbolic names of the signals (without the `SIG` prefix) supported on your system, use the `kill -l` command.
- *command* specifies a `/bin/sh` command line to be invoked. Do not quote the command line inside an action definition. Do not specify a signal followed by an action that triggers the same signal. For example, do not specify `RESUME_CONTROL=br resume`. This causes a deadlock between the signal and the action.

## Description

Changes the behavior of the RESUME action in LSF.

- The contents of the configuration line for the action are run with `/bin/sh -c` so you can use shell features in the command.
- The standard input, output, and error of the command are redirected to the NULL device, so you cannot tell directly whether the command runs correctly. The default null device on UNIX is `/dev/null`.
- The command is run as the user of the job.
- All environment variables set for the job are also set for the command action. The following additional environment variables are set:
  - `LSB_JOBPGIDS` — a list of current process group IDs of the job
  - `LSB_JOBPIIDS` — a list of current process IDs of the job

The command path can contain up to 4094 characters for UNIX and Linux, or up to 255 characters for Windows, including the directory, file name, and expanded values for `%J` (*job\_ID*) and `%I` (*index\_ID*).

## Default

- On UNIX, by default, RESUME sends SIGCONT.
- On Windows, actions equivalent to the UNIX signals have been implemented to do the default job control actions. Job control messages replace the SIGINT and SIGTERM signals, but only customized applications are able to process them.

## RTASK\_GONE\_ACTION

### Syntax

```
RTASK_GONE_ACTION="[KILLJOB_TASKDONE | KILLJOB_TASKEEXIT]
[IGNORE_TASKCRASH]"
```

### Description

Defines the actions LSF should take if it detects that a remote task of a parallel or distributed job is gone.

This parameter only applies to the `blaunch` distributed application framework.

#### **IGNORE\_TASKCRASH**

A remote task crashes. LSF does nothing. The job continues to launch the next task.

#### **KILLJOB\_TASKDONE**

A remote task exits with zero value. LSF terminates all tasks in the job.

#### **KILLJOB\_TASKEEXIT**

A remote task exits with non-zero value. LSF terminates all tasks in the job.

## Environment variable

When defined in an application profile, the `LSB_DJOB_RTASK_GONE_ACTION` variable is set when running `bsub -app` for the specified application.

You can also use the environment variable `LSB_DJOB_RTASK_GONE_ACTION` to override the value set in the application profile.

## Example

```
RTASK_GONE_ACTION="IGNORE_TASKCRASH KILLJOB_TASKEXIT"
```

## Default

Not defined. LSF does nothing.

# RUNLIMIT

## Syntax

```
RUNLIMIT=[hour:]minute[host_name | host_model]
```

## Description

The default run limit. The name of a host or host model specifies the runtime normalization host to use.

By default, jobs that are in the RUN state for longer than the specified run limit are killed by LSF. You can optionally provide your own termination job action to override this default.

Jobs submitted with a job-level run limit (`bsub -W`) that is less than the run limit are killed when their job-level run limit is reached. Jobs submitted with a run limit greater than the maximum run limit are rejected. Application-level limits override any default limit specified in the queue.

---

### Note:

If you want to provide an estimated run time for scheduling purposes without killing jobs that exceed the estimate, define the `RUNTIME` parameter in the application profile, or submit the job with `-We` instead of a run limit.

---

The run limit is 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 run limit you specify is the normalized run time. This is done so that the job does approximately the same amount of processing, even if it is sent to host with a faster or slower CPU. Whenever a normalized run time is given, the actual time on the execution host is the specified time multiplied by the CPU factor of the normalization host then divided by the CPU factor of the execution host.

If `ABS_RUNLIMIT=Y` is defined in `lsb.params` or in the application profile, the runtime limit is not normalized by the host CPU factor. Absolute wall-clock run time is used for all jobs submitted to an application profile with a run limit configured.

Optionally, you can supply a host name or a host model name defined in LSF. You must insert `/'` between the run limit and the host name or model name. (See `lsinfo(1)` to get host model information.)

If no host or host model is given, LSF uses the default runtime normalization host defined at the queue level (`DEFAULT_HOST_SPEC` in `lsb.queues`) if it has been configured; otherwise, LSF uses the default CPU time normalization host defined at the cluster level (`DEFAULT_HOST_SPEC` in `lsb.params`) if it has been configured; otherwise, the host with the largest CPU factor (the fastest host in the cluster).

For MultiCluster jobs, if no other CPU time normalization host is defined and information about the submission host is not available, LSF uses the host with the largest CPU factor (the fastest host in the cluster).

Jobs submitted to a chunk job queue are not chunked if `RUNLIMIT` is greater than 30 minutes.

## Default

Unlimited

# RUNTIME

## Syntax

**RUNTIME**=[*hour*:]*minute*[/*host\_name* | *host\_model*]

## Description

The RUNTIME parameter specifies an estimated run time for jobs associated with an application. LSF uses the RUNTIME value for scheduling purposes only, and does not kill jobs that exceed this value unless the jobs also exceed a defined RUNLIMIT. The format of runtime estimate is same as the RUNLIMIT parameter.

The job-level runtime estimate specified by `bsub - We` overrides the RUNTIME setting in an application profile.

The following LSF features use the RUNTIME value to schedule jobs:

- Job chunking
- Advanced reservation
- SLA
- Slot reservation
- Backfill

## Default

Not defined

# STACKLIMIT

## Syntax

**STACKLIMIT**=*integer*

## Description

The per-process (soft) stack segment size limit for all of the processes belonging to a job from this queue (see `getrlimit(2)`). Application-level limits override any default limit specified in the queue, but must be less than the hard limit of the submission queue.

By default, the limit is specified in KB. Use `LSF_UNIT_FOR_LIMITS` in `lsf.conf` to specify a larger unit for the the limit (MB, GB, TB, PB, or EB).

## Default

Unlimited

# SUCCESS\_EXIT\_VALUES

## Syntax

**SUCCESS\_EXIT\_VALUES**=[*exit\_code* ...]

## Description

Specifies exit values used by LSF to determine if job was done successfully. Use spaces to separate multiple exit codes. Job-level success exit values specified with the `LSB_SUCCESS_EXIT_VALUES` environment variable override the configuration in application profile.

Use `SUCCESS_EXIT_VALUES` for applications that successfully exit with non-zero values so that LSF does not interpret non-zero exit codes as job failure.

`exit_code` should be the value between 0 and 255. Use spaces to separate exit code values.

## Default

Not defined, Jobs do not specify a success exit value.

# SUSPEND\_CONTROL

## Syntax

**SUSPEND\_CONTROL=***signal* | *command* | **CHKPNT**

---

### Remember:

Unlike the `JOB_CONTROLS` parameter in `lsb.queues`, the `SUSPEND_CONTROL` parameter does not require square brackets (`[ ]`) around the action.

---

- *signal* is a UNIX signal name (for example, `SIGTSTP`). The specified signal is sent to the job. The same set of signals is not supported on all UNIX systems. To display a list of the symbolic names of the signals (without the `SIG` prefix) supported on your system, use the `kill -l` command.
- *command* specifies a `/bin/sh` command line to be invoked.
  - Do not quote the command line inside an action definition.
  - Do not specify a signal followed by an action that triggers the same signal. For example, do not specify `SUSPEND_CONTROL=bst op`. This causes a deadlock between the signal and the action.
- `CHKPNT` is a special action, which causes the system to checkpoint the job. The job is checkpointed and then stopped by sending the `SIGSTOP` signal to the job automatically.

## Description

Changes the behavior of the `SUSPEND` action in LSF.

- The contents of the configuration line for the action are run with `/bin/sh -c` so you can use shell features in the command.
- The standard input, output, and error of the command are redirected to the `NULL` device, so you cannot tell directly whether the command runs correctly. The default null device on UNIX is `/dev/null`.
- The command is run as the user of the job.
- All environment variables set for the job are also set for the command action. The following additional environment variables are set:
  - `LSB_JOBPGIDS` — a list of current process group IDs of the job
  - `LSB_JOBPIIDS` — a list of current process IDs of the job

- `LSB_SUSP_REASONS` — an integer representing a bitmap of suspending reasons as defined in `lsbatch.h`. The suspending reason can allow the command to take different actions based on the reason for suspending the job.
- `LSB_SUSP_SUBREASONS` — an integer representing the load index that caused the job to be suspended

When the suspending reason `SUSP_LOAD_REASON` (suspended by load) is set in `LSB_SUSP_REASONS`, `LSB_SUSP_SUBREASONS` is set to one of the load index values defined in `lsf.h`.

Use `LSB_SUSP_REASONS` and `LSB_SUSP_SUBREASONS` together in your custom job control to determine the exact load threshold that caused a job to be suspended.

- If an additional action is necessary for the `SUSPEND` command, that action should also send the appropriate signal to the application. Otherwise, a job can continue to run even after being suspended by LSF. For example, `SUSPEND_CONTROL=bkill $LSB_JOBPI DS; command`

The command path can contain up to 4094 characters for UNIX and Linux, or up to 255 characters for Windows, including the directory, file name, and expanded values for `%J` (*job\_ID*) and `%I` (*index\_ID*).

## Default

- On UNIX, by default, `SUSPEND` sends `SIGTSTP` for parallel or interactive jobs and `SIGSTOP` for other jobs.
- On Windows, actions equivalent to the UNIX signals have been implemented to do the default job control actions. Job control messages replace the `SIGINT` and `SIGTERM` signals, but only customized applications are able to process them.

# SWAPLIMIT

## Syntax

**SWAPLIMIT**=*integer*

## Description

Limits the amount of total virtual memory limit for the job.

This limit applies to the whole job, no matter how many processes the job may contain. Application-level limits override any default limit specified in the queue.

The action taken when a job exceeds its `SWAPLIMIT` or `PROCESLIMIT` is to send `SIGQUIT`, `SIGINT`, `SIGTERM`, and `SIGKILL` in sequence. For `CPULIMIT`, `SIGXCPU` is sent before `SIGINT`, `SIGTERM`, and `SIGKILL`.

By default, the limit is specified in KB. Use `LSF_UNIT_FOR_LIMITS` in `lsf.conf` to specify a larger unit for the the limit (MB, GB, TB, PB, or EB).

## Default

Unlimited

# TERMINATE\_CONTROL

## Syntax

**TERMINATE\_CONTROL**=*signal* | *command* | **CHKPNT**

**Remember:**

Unlike the `JOB_CONTROLS` parameter in `lsb.queues`, the `TERMINATE_CONTROL` parameter does not require square brackets (`[ ]`) around the action.

- *signal* is a UNIX signal name (for example, `SIGTERM`). The specified signal is sent to the job. The same set of signals is not supported on all UNIX systems. To display a list of the symbolic names of the signals (without the `SIG` prefix) supported on your system, use the `kill -l` command.
- *command* specifies a `/bin/sh` command line to be invoked.
  - Do not quote the command line inside an action definition.
  - Do not specify a signal followed by an action that triggers the same signal. For example, do not specify `TERMINATE_CONTROL=bkill`. This causes a deadlock between the signal and the action.
- `CHKPNT` is a special action, which causes the system to checkpoint the job. The job is checkpointed and killed automatically.

## Description

Changes the behavior of the `TERMINATE` action in LSF.

- The contents of the configuration line for the action are run with `/bin/sh -c` so you can use shell features in the command.
- The standard input, output, and error of the command are redirected to the `NULL` device, so you cannot tell directly whether the command runs correctly. The default null device on UNIX is `/dev/null`.
- The command is run as the user of the job.
- All environment variables set for the job are also set for the command action. The following additional environment variables are set:
  - `LSB_JOBPGIDS` — a list of current process group IDs of the job
  - `LSB_JOBPIDS` — a list of current process IDs of the job

The command path can contain up to 4094 characters for UNIX and Linux, or up to 255 characters for Windows, including the directory, file name, and expanded values for `%J` (*job\_ID*) and `%I` (*index\_ID*).

## Default

- On UNIX, by default, `TERMINATE` sends `SIGINT`, `SIGTERM` and `SIGKILL` in that order.
- On Windows, actions equivalent to the UNIX signals have been implemented to do the default job control actions. Job control messages replace the `SIGINT` and `SIGTERM` signals, but only customized applications are able to process them. Termination is implemented by the `TerminateProcess()` system call.

# THREADLIMIT

## Syntax

`THREADLIMIT=integer`

## Description

Limits the number of concurrent threads that can be part of a job. Exceeding the limit causes the job to terminate. The system sends the following signals in sequence to all processes belongs to the job: `SIGINT`, `SIGTERM`, and `SIGKILL`.

By default, jobs submitted to the queue without a job-level thread limit are killed when the thread limit is reached. Application-level limits override any default limit specified in the queue.

The limit must be a positive integer.

## Default

Unlimited

# USE\_PAM\_CREDS

## Syntax

**USE\_PAM\_CREDS=y | n**

## Description

If `USE_PAM_CREDS=y`, applies PAM limits to an application when its job is dispatched to a Linux host using PAM. PAM limits are system resource limits defined in `l i m i t s. conf`.

When `USE_PAM_CREDS` is enabled, PAM limits override others.

If the execution host does not have PAM configured and this parameter is enabled, the job fails.

For parallel jobs, only takes effect on the first execution host.

Overrides `MEMLIMIT_TYPE=Process`.

Overridden (for CPU limit only) by `LSB_JOB_CPULIMIT=y`.

Overridden (for memory limits only) by `LSB_JOB_MEMLIMIT=y`.

## Default

n

## lsb.events

The LSF batch event log file `lsb.events` is used to display LSF batch event history and for `mbatchd` failure recovery.

Whenever a host, job, or queue changes status, a record is appended to the event log file. The file is located in `LSB_SHAREDIR/cluster_name/logdir`, where `LSB_SHAREDIR` must be defined in `lsf.conf` (5) and `cluster_name` is the name of the LSF cluster, as returned by `lsid`. See `mbatchd`(8) for the description of `LSB_SHAREDIR`.

The `bhist` command searches the most current `lsb.events` file for its output.

## lsb.events structure

The event log file is an ASCII file with one record per line. For the `lsb.events` file, the first line has the format `# history_seek_position`, which indicates the file position of the first history event after log switch. For the `lsb.events.#file`, the first line has the format `# timestamp_most_recent_event`, which gives the timestamp of the most recent event in the file.

## Limiting the size of lsb.events

Use `MAX_JOB_NUM` in `lsb.params` to set the maximum number of finished jobs whose events are to be stored in the `lsb.events` log file.

Once the limit is reached, `mbatchd` starts a new event log file. The old event log file is saved as `lsb.events.n`, with subsequent sequence number suffixes incremented by 1 each time a new log file is started. Event logging continues in the new `lsb.events` file.

## Records and fields

The fields of a record are separated by blanks. The first string of an event record indicates its type. The following types of events are recorded:

- JOB\_NEW
- JOB\_FORWARD
- JOB\_ACCEPT
- JOB\_START
- JOB\_START\_ACCEPT
- JOB\_STATUS
- JOB\_SWITCH
- JOB\_MOVE
- QUEUE\_CTRL
- HOST\_CTRL
- MBD\_START
- MBD\_DIE
- UNFULFILL
- LOAD\_INDEX
- JOB\_SIGACT
- MIG
- JOB\_MODIFY2
- JOB\_SIGNAL

- JOB\_EXECUTE
- JOB\_REQUEUE
- JOB\_CLEAN
- JOB\_EXCEPTION
- JOB\_EXT\_MSG
- JOB\_ATTA\_DATA
- JOB\_CHUNK
- SBD\_UNREPORTED\_STATUS
- PRE\_EXEC\_START
- JOB\_FORCE
- GRP\_ADD
- GRP\_MOD
- LOG\_SWITCH
- JOB\_RESIZE\_NOTIFY\_START
- JOB\_RESIZE\_NOTIFY\_ACCEPT
- JOB\_RESIZE\_NOTIFY\_DONE
- JOB\_RESIZE\_RELEASE
- JOB\_RESIZE\_CANCEL

## JOB\_NEW

A new job has been submitted. The fields in order of occurrence are:

**Version number (%s)**

The version number

**Event time (%d)**

The time of the event

**jobId (%d)**

Job ID

**userId (%d)**

UNIX user ID of the submitter

**options (%d)**

Bit flags for job processing

**numProcessors (%d)**

Number of processors requested for execution

**submitTime (%d)**

Job submission time

**beginTime (%d)**

Start time – the job should be started on or after this time

**termTime (%d)**

Termination deadline – the job should be terminated by this time (%d)

**sigValue (%d)**

Signal value

**chkpntPeriod (%d)**

Checkpointing period

**restartPid (%d)**

Restart process ID

**userName (%s)**

User name

**rLimits**

Soft CPU time limit (%d), see `getrlimit(2)`

**rLimits**

Soft file size limit (%d), see `getrlimit(2)`

**rLimits**

Soft data segment size limit (%d), see `getrlimit(2)`

**rLimits**

Soft stack segment size limit (%d), see `getrlimit(2)`

**rLimits**

Soft core file size limit (%d), see `getrlimit(2)`

**rLimits**

Soft memory size limit (%d), see `getrlimit(2)`

**rLimits**

Reserved (%d)

**rLimits**

Reserved (%d)

**rLimits**

Reserved (%d)

**rLimits**

Soft run time limit (%d), see `getrlimit(2)`

**rLimits**

Reserved (%d)

**hostSpec (%s)**

Model or host name for normalizing CPU time and run time

**hostFactor (%f)**

	CPU factor of the above host
<b>umask (%d)</b>	
	File creation mask for this job
<b>queue (%s)</b>	
	Name of job queue to which the job was submitted
<b>resReq (%s)</b>	
	Resource requirements
<b>fromHost (%s)</b>	
	Submission host name
<b>cwd (%s)</b>	
	Current working directory (up to 4094 characters for UNIX or 255 characters for Windows)
<b>chkpntDir (%s)</b>	
	Checkpoint directory
<b>inFile (%s)</b>	
	Input file name (up to 4094 characters for UNIX or 255 characters for Windows)
<b>outFile (%s)</b>	
	Output file name (up to 4094 characters for UNIX or 255 characters for Windows)
<b>errFile (%s)</b>	
	Error output file name (up to 4094 characters for UNIX or 255 characters for Windows)
<b>subHomeDir (%s)</b>	
	Submitter's home directory
<b>jobFile (%s)</b>	
	Job file name
<b>numAskedHosts (%d)</b>	
	Number of candidate host names
<b>askedHosts (%s)</b>	
	List of names of candidate hosts for job dispatching
<b>dependCond (%s)</b>	
	Job dependency condition
<b>preExecCmd (%s)</b>	
	Job pre-execution command
<b>jobName (%s)</b>	

	Job name (up to 4094 characters)
<b>command (%s)</b>	Job command (up to 4094 characters for UNIX or 255 characters for Windows)
<b>nx (%d)</b>	Number of files to transfer (%d)
<b>xf (%s)</b>	List of file transfer specifications
<b>mailUser (%s)</b>	Mail user name
<b>projectName (%s)</b>	Project name
<b>niosPort (%d)</b>	Callback port if batch interactive job
<b>maxNumProcessors (%d)</b>	Maximum number of processors
<b>schedHostType (%s)</b>	Execution host type
<b>loginShell (%s)</b>	Login shell
<b>timeEvent (%d)</b>	Time Event, for job dependency condition; specifies when time event ended
<b>userGroup (%s)</b>	User group
<b>exceptList (%s)</b>	Exception handlers for the job
<b>options2 (%d)</b>	Bit flags for job processing
<b>idx (%d)</b>	Job array index
<b>inFileSpool (%s)</b>	Spool input file (up to 4094 characters for UNIX or 255 characters for Windows)
<b>commandSpool (%s)</b>	Spool command file (up to 4094 characters for UNIX or 255 characters for Windows)

**jobSpoolDir (%s)**

Job spool directory (up to 4094 characters for UNIX or 255 characters for Windows)

**userPriority (%d)**

User priority

**rsvld %s**

Advance reservation ID; for example, "user2#0"

**jobGroup (%s)**

The job group under which the job runs

**sla (%s)**

SLA service class name under which the job runs

**rLimits**

Thread number limit

**extsched (%s)**

External scheduling options

**warningAction (%s)**

Job warning action

**warningTimePeriod (%d)**

Job warning time period in seconds

**SLArunLimit (%d)**

Absolute run time limit of the job for SLA service classes

**licenseProject (%s)**

Platform License Scheduler project name

**options3 (%d)**

Bit flags for job processing

**app (%s)**

Application profile name

**postExecCmd (%s)**

Post-execution command to run on the execution host after the job finishes

**runtimeEstimation (%d)**

Estimated run time for the job

**requeueEValues (%s)**

Job exit values for automatic job requeue

**resizeNotifyCmd (%s)**

Resize notification command to run on the first execution host to inform job of a resize event.

**jobDescription (%s)**

Job description (up to 4094 characters).

**submitEXT**

Submission extension field, reserved for internal use.

**Num (%d)**

Number of elements (key-value pairs) in the structure.

**key (%s)**

Reserved for internal use.

**value (%s)**

Reserved for internal use.

## JOB\_FORWARD

A job has been forwarded to a remote cluster (Platform MultiCluster only).

If `LSF_HPC_EXTENSIONS="SHORT_EVENTFILE"` is specified in `lsf.conf`, older daemons and commands (pre-LSF Version 6.0) cannot recognize the `lsb.events` file format.

The fields in order of occurrence are:

**Version number (%s)**

The version number

**Event time (%d)**

The time of the event

**jobId (%d)**

Job ID

**numReserHosts (%d)**

Number of reserved hosts in the remote cluster

If `LSF_HPC_EXTENSIONS="SHORT_EVENTFILE"` is specified in `lsf.conf`, the value of this field is the number of `.hosts` listed in the `reserHosts` field.

**cluster (%s)**

Remote cluster name

**reserHosts (%s)**

List of names of the reserved hosts in the remote cluster

If `LSF_HPC_EXTENSIONS="SHORT_EVENTFILE"` is specified in `lsf.conf`, the value of this field is logged in a shortened format.

**idx (%d)**

Job array index

## JOB\_ACCEPT

A job from a remote cluster has been accepted by this cluster. The fields in order of occurrence are:

**Version number (%s)**

The version number

**Event time (%d)**

The time of the event

**jobId (%d)**

Job ID at the accepting cluster

**remoteJid (%d)**

Job ID at the submission cluster

**cluster (%s)**

Job submission cluster name

**idx (%d)**

Job array index

## JOB\_START

A job has been dispatched.

If `LSF_HPC_EXTENSIONS="SHORT_EVENTFILE"` is specified in `lsf.conf`, older daemons and commands (pre-LSF Version 6.0) cannot recognize the `lsb.events` file format.

The fields in order of occurrence are:

**Version number (%s)**

The version number

**Event time (%d)**

The time of the event

**jobId (%d)**

Job ID

**jStatus (%d)**

Job status, (4, indicating the RUN status of the job)

**jobPid (%d)**

Job process ID

**jobPGid (%d)**

Job process group ID

**hostFactor (%f)**

CPU factor of the first execution host

**numExHosts (%d)**

Number of processors used for execution

If LSF\_HPC\_EXTENSIONS="SHORT\_EVENTFILE" is specified in `lsf.conf`, the value of this field is the number of `.hosts` listed in the `execHosts` field.

**execHosts (%s)**

List of execution host names

If LSF\_HPC\_EXTENSIONS="SHORT\_EVENTFILE" is specified in `lsf.conf`, the value of this field is logged in a shortened format.

**queuePreCmd (%s)**

Pre-execution command

**queuePostCmd (%s)**

Post-execution command

**jFlags (%d)**

Job processing flags

**userGroup (%s)**

User group name

**idx (%d)**

Job array index

**additionalInfo (%s)**

Placement information of HPC jobs

**jFlags2 (%d)**

## JOB\_START\_ACCEPT

A job has started on the execution host(s). The fields in order of occurrence are:

**Version number (%s)**

The version number

**Event time (%d)**

The time of the event

**jobId (%d)**

Job ID

**jobPid (%d)**

Job process ID

**jobPGid (%d)**

Job process group ID

**idx (%d)**

Job array index

## JOB\_STATUS

The status of a job changed after dispatch. The fields in order of occurrence are:

**Version number (%s)**

The version number

**Event time (%d)**

The time of the event

**jobId (%d)**

Job ID

**jStatus (%d)**

New status, see <1 sf /1 sbat ch. h>

For JOB\_STAT\_EXIT (32) and JOB\_STAT\_DONE (64), host-based resource usage information is appended to the JOB\_STATUS record in the fields numHostRusage and hostRusage.

**reason (%d)**

Pending or suspended reason code, see <1 sf /1 sbat ch. h>

**subreasons (%d)**

Pending or suspended subreason code, see <1 sf /1 sbat ch. h>

**cpuTime (%f)**

CPU time consumed so far

**endTime (%d)**

Job completion time

**ru (%d)**

Resource usage flag

**lsfRusage (%s)**

Resource usage statistics, see <1 sf /1 sf. h>

**exitStatus (%d)**

Exit status of the job, see <1 sf /1 sbat ch. h>

**idx (%d)**

Job array index

**exitInfo (%d)**

Job termination reason, see `<lsf/l sbatch.h>`

**duration4PreemptBackfill**

How long a backfilled job can run; used for preemption backfill jobs

**numHostRusage(%d)**

For a job status of JOB\_STAT\_EXIT (32) or JOB\_STAT\_DONE (64), this field contains the number of host-based resource usage entries (hostRusage) that follow. 0 unless HPC\_EXTENSIONS="HOST\_RUSAGE" is set in lsf.conf.

**hostRusage**

For a job status of JOB\_STAT\_EXIT (32) or JOB\_STAT\_DONE (64), these fields contain host-based resource usage information for the job for parallel jobs when HPC\_EXTENSIONS="HOST\_RUSAGE" is set in lsf.conf.

**hostname (%s)**

Name of the host.

**mem(%d)**

Total resident memory usage of all processes in the job running on this host.

**swap(%d)**

Total virtual memory usage of all processes in the job running on this host.

**utime(%d)**

User time used on this host.

**stime(%d)**

System time used on this host.

**hHostExtendInfo(%d)**

Number of following key-value pairs containing extended host information (PGIDs and PIDs). Set to 0 in lsb.events, lsb.acct, and lsb.stream files.

## JOB\_SWITCH

A job switched from one queue to another (bswitch). The fields in order of occurrence are:

**Version number (%s)**

The version number

**Event time (%d)**

The time of the event

**userId (%d)**

UNIX user ID of the user invoking the command

**jobId (%d)**

Job ID

<b>queue (%s)</b>	Target queue name
<b>idx (%d)</b>	Job array index
<b>userName (%s)</b>	Name of the job submitter

## JOB\_MOVE

A job moved toward the top or bottom of its queue (bbot or bt op). The fields in order of occurrence are:

<b>Version number (%s)</b>	The version number
<b>Event time (%d)</b>	The time of the event
<b>userId (%d)</b>	UNIX user ID of the user invoking the command
<b>jobId (%d)</b>	Job ID
<b>position (%d)</b>	Position number
<b>base (%d)</b>	Operation code, (TO_TOP or TO_BOTTOM), see <1 sf /1 sbat ch. h>
<b>idx (%d)</b>	Job array index
<b>userName (%s)</b>	Name of the job submitter

## QUEUE\_CTRL

A job queue has been altered. The fields in order of occurrence are:

<b>Version number (%s)</b>	The version number
<b>Event time (%d)</b>	The time of the event
<b>opCode (%d)</b>	Operation code), see <1 sf /1 sbat ch. h>

**queue (%s)**

Queue name

**userId (%d)**

UNIX user ID of the user invoking the command

**userName (%s)**

Name of the user

**ctrlComments (%s)**

Administrator comment text from the -C option of `badmi n` queue control commands `qclose`, `qopen`, `qact`, and `qi nact`

## HOST\_CTRL

A batch server host changed status. The fields in order of occurrence are:

**Version number (%s)**

The version number

**Event time (%d)**

The time of the event

**opCode (%d)**

Operation code, see `<lsf/l sbatch. h>`

**host (%s)**

Host name

**userId (%d)**

UNIX user ID of the user invoking the command

**userName (%s)**

Name of the user

**ctrlComments (%s)**

Administrator comment text from the -C option of `badmi n` host control commands `hclose` and `hopen`

## MBD\_START

The `mbatchd` has started. The fields in order of occurrence are:

**Version number (%s)**

The version number

**Event time (%d)**

The time of the event

**master (%s)**

Master host name

**cluster (%s)**

cluster name

**numHosts (%d)**

Number of hosts in the cluster

**numQueues (%d)**

Number of queues in the cluster

## MBD\_DIE

The `mbat chd` died. The fields in order of occurrence are:

**Version number (%s)**

The version number

**Event time (%d)**

The time of the event

**master (%s)**

Master host name

**numRemoveJobs (%d)**

Number of finished jobs that have been removed from the system and logged in the current event file

**exitCode (%d)**

Exit code from `mbat chd`

**ctrlComments (%s)**

Administrator comment text from the `-C` option of `badmi n mbdrestart`

## UNFULFILL

Actions that were not taken because the `mbat chd` was unable to contact the `sbat chd` on the job execution host. The fields in order of occurrence are:

**Version number (%s)**

The version number

**Event time (%d)**

The time of the event

**jobId (%d)**

Job ID

**notSwitched (%d)**

Not switched: the `mbatchd` has switched the job to a new queue, but the `sbatchd` has not been informed of the switch

**sig (%d)**

Signal: this signal has not been sent to the job

**sig1 (%d)**

Checkpoint signal: the job has not been sent this signal to checkpoint itself

**sig1Flags (%d)**

Checkpoint flags, see `<lsf/lscratch.h>`

**chkPeriod (%d)**

New checkpoint period for job

**notModified (%s)**

If set to true, then parameters for the job cannot be modified.

**idx (%d)**

Job array index

## LOAD\_INDEX

`mbatchd` restarted with these load index names (see `lsf.cluster(5)`). The fields in order of occurrence are:

**Version number (%s)**

The version number

**Event time (%d)**

The time of the event

**nIdx (%d)**

Number of index names

**name (%s)**

List of index names

## JOB\_SIGACT

An action on a job has been taken. The fields in order of occurrence are:

**Version number (%s)**

The version number

**Event time (%d)**

The time of the event

**jobId (%d)**

Job ID

<b>period (%d)</b>	Action period
<b>pid (%d)</b>	Process ID of the child sbat chd that initiated the action
<b>jstatus (%d)</b>	Job status
<b>reasons (%d)</b>	Job pending reasons
<b>flags (%d)</b>	Action flags, see <1 sf /1 sbat ch. h>
<b>actStatus (%d)</b>	Action status: 1: Action started 2: One action preempted other actions 3: Action succeeded 4: Action Failed
<b>signalSymbol (%s)</b>	Action name, accompanied by actFlags
<b>idx (%d)</b>	Job array index

## MIG

A job has been migrated (bmi g). The fields in order of occurrence are:

<b>Version number (%s)</b>	The version number
<b>Event time (%d)</b>	The time of the event
<b>jobId (%d)</b>	Job ID
<b>numAskedHosts (%d)</b>	Number of candidate hosts for migration
<b>askedHosts (%s)</b>	List of names of candidate hosts
<b>userId (%d)</b>	

UNIX user ID of the user invoking the command

**idx (%d)**

Job array index

**userName (%s)**

Name of the job submitter

## JOB\_MODIFY2

This is created when the `mbat chd` modifies a previously submitted job with `bmod`.

**Version number (%s)**

The version number

**Event time (%d)**

The time of the event

**jobIdStr (%s)**

Job ID

**options (%d)**

Bit flags for job modification options processing

**options2 (%d)**

Bit flags for job modification options processing

**delOptions (%d)**

Delete options for the options field

**userId (%d)**

UNIX user ID of the submitter

**userName (%s)**

User name

**submitTime (%d)**

Job submission time

**umask (%d)**

File creation mask for this job

**numProcessors (%d)**

Number of processors requested for execution. The value 2147483646 means the number of processors is undefined.

**beginTime (%d)**

Start time – the job should be started on or after this time

**termTime (%d)**

Termination deadline – the job should be terminated by this time

**sigValue (%d)**

Signal value

**restartPid (%d)**

Restart process ID for the original job

**jobName (%s)**

Job name (up to 4094 characters)

**queue (%s)**

Name of job queue to which the job was submitted

**numAskedHosts (%d)**

Number of candidate host names

**askedHosts (%s)**

List of names of candidate hosts for job dispatching; blank if the last field value is 0. If there is more than one host name, then each additional host name will be returned in its own field

**resReq (%s)**

Resource requirements

**rLimits**

Soft CPU time limit (%d), see `getrlimit(2)`

**rLimits**

Soft file size limit (%d), see `getrlimit(2)`

**rLimits**

Soft data segment size limit (%d), see `getrlimit(2)`

**rLimits**

Soft stack segment size limit (%d), see `getrlimit(2)`

**rLimits**

Soft core file size limit (%d), see `getrlimit(2)`

**rLimits**

Soft memory size limit (%d), see `getrlimit(2)`

**rLimits**

Reserved (%d)

**rLimits**

Reserved (%d)

**rLimits**

Reserved (%d)

**rLimits**

Soft run time limit (%d), see `getrlimit(2)`

**rLimits**

Reserved (%d)

**hostSpec (%s)**

Model or host name for normalizing CPU time and run time

**dependCond (%s)**

Job dependency condition

**timeEvent (%d)**

Time Event, for job dependency condition; specifies when time event ended

**subHomeDir (%s)**

Submitter's home directory

**inFile (%s)**

Input file name (up to 4094 characters for UNIX or 255 characters for Windows)

**outFile (%s)**

Output file name (up to 4094 characters for UNIX or 255 characters for Windows)

**errFile (%s)**

Error output file name (up to 4094 characters for UNIX or 255 characters for Windows)

**command (%s)**

Job command (up to 4094 characters for UNIX or 255 characters for Windows)

**chkpntPeriod (%d)**

Checkpointing period

**chkpntDir (%s)**

Checkpoint directory

**nxf (%d)**

Number of files to transfer

**xf (%s)**

List of file transfer specifications

**jobFile (%s)**

Job file name

**fromHost (%s)**

Submission host name

<b>cwd (%s)</b>	Current working directory (up to 4094 characters for UNIX or 255 characters for Windows)
<b>preExecCmd (%s)</b>	Job pre-execution command
<b>mailUser (%s)</b>	Mail user name
<b>projectName (%s)</b>	Project name
<b>niosPort (%d)</b>	Callback port if batch interactive job
<b>maxNumProcessors (%d)</b>	Maximum number of processors. The value 2147483646 means the maximum number of processors is undefined.
<b>loginShell (%s)</b>	Login shell
<b>schedHostType (%s)</b>	Execution host type
<b>userGroup (%s)</b>	User group
<b>exceptList (%s)</b>	Exception handlers for the job
<b>delOptions2 (%d)</b>	Delete options for the options2 field
<b>inFileSpool (%s)</b>	Spool input file (up to 4094 characters for UNIX or 255 characters for Windows)
<b>commandSpool (%s)</b>	Spool command file (up to 4094 characters for UNIX or 255 characters for Windows)
<b>userPriority (%d)</b>	User priority
<b>rsvld %s</b>	Advance reservation ID; for example, "user2#0"
<b>extsched (%s)</b>	External scheduling options

**warningTimePeriod (%d)**

Job warning time period in seconds

**warningAction (%s)**

Job warning action

**jobGroup (%s)**

The job group to which the job is attached

**sla (%s)**

SLA service class name that the job is to be attached to

**licenseProject (%s)**

Platform License Scheduler project name

**options3 (%d)**

Bit flags for job processing

**delOption3 (%d)**

Delete options for the `options3` field

**app (%s)**

Application profile name

**apsString (%s)**

Absolute priority scheduling (APS) value set by administrator

**postExecCmd (%s)**

Post-execution command to run on the execution host after the job finishes

**runtimeEstimation (%d)**

Estimated run time for the job

**requeueEValues (%s)**

Job exit values for automatic job requeue

**resizeNotifyCmd (%s)**

Resize notification command to run on the first execution host to inform job of a resize event.

**jobdescription (%s)**

Job description (up to 4094 characters).

## JOB\_SIGNAL

This is created when a job is signaled with `bkill` or deleted with `bdel`. The fields are in the order they appended:

**Version number (%s)**

	The version number
<b>Event time (%d)</b>	The time of the event
<b>jobId (%d)</b>	Job ID
<b>userId (%d)</b>	UNIX user ID of the user invoking the command
<b>runCount (%d)</b>	Number of runs
<b>signalSymbol (%s)</b>	Signal name
<b>idx (%d)</b>	Job array index
<b>userName (%s)</b>	Name of the job submitter

## JOB\_EXECUTE

This is created when a job is actually running on an execution host. The fields in order of occurrence are:

<b>Version number (%s)</b>	The version number
<b>Event time (%d)</b>	The time of the event
<b>jobId (%d)</b>	Job ID
<b>execUid (%d)</b>	Mapped UNIX user ID on execution host
<b>jobPGid (%d)</b>	Job process group ID
<b>execCwd (%s)</b>	Current working directory job used on execution host (up to 4094 characters for UNIX or 255 characters for Windows)
<b>execHome (%s)</b>	Home directory job used on execution host
<b>execUsername (%s)</b>	

	Mapped user name on execution host
<b>jobPid (%d)</b>	Job process ID
<b>idx (%d)</b>	Job array index
<b>additionalInfo (%s)</b>	Placement information of HPC jobs
<b>SLAscaledRunLimit (%d)</b>	Run time limit for the job scaled by the execution host
<b>execRusage</b>	An internal field used by LSF.
<b>Position</b>	An internal field used by LSF.
<b>duration4PreemptBackfill</b>	How long a backfilled job can run; used for preemption backfill jobs

## JOB\_QUEUE

This is created when a job ended and requeued by mbat chd. The fields in order of occurrence are:

<b>Version number (%s)</b>	The version number
<b>Event time (%d)</b>	The time of the event
<b>jobId (%d)</b>	Job ID
<b>idx (%d)</b>	Job array index

## JOB\_CLEAN

This is created when a job is removed from the mbat chd memory. The fields in order of occurrence are:

<b>Version number (%s)</b>	The version number
<b>Event time (%d)</b>	The time of the event
<b>jobId (%d)</b>	

Job ID

**idx (%d)**

Job array index

## JOB\_EXCEPTION

This is created when an exception condition is detected for a job. The fields in order of occurrence are:

**Version number (%s)**

The version number

**Event time (%d)**

The time of the event

**jobId (%d)**

Job ID

**exceptMask (%d)**

Exception Id

0x01: missed

0x02: overrun

0x04: underrun

0x08: abend

0x10: cantrun

0x20: hostfail

0x40: startfail

0x100:runtime\_est\_exceeded

**actMask (%d)**

Action Id

0x01: kill

0x02: alarm

0x04: rerun

0x08: setexcept

**timeEvent (%d)**

Time Event, for missed exception specifies when time event ended.

**exceptInfo (%d)**

Except Info, pending reason for missed or cant run exception, the exit code of the job for the abend exception, otherwise 0.

**idx (%d)**

Job array index

## JOB\_EXT\_MSG

An external message has been sent to a job. The fields in order of occurrence are:

**Version number (%s)**

The version number

**Event time (%d)**

The time of the event

**jobId (%d)**

Job ID

**idx (%d)**

Job array index

**msgIdx (%d)**

Index in the list

**userId (%d)**

Unique user ID of the user invoking the command

**dataSize (%ld)**

Size of the data if it has any, otherwise 0

**postTime (%ld)**

Message sending time

**dataStatus (%d)**

Status of the attached data

**desc (%s)**

Text description of the message

**userName (%s)**

Name of the author of the message

## JOB\_ATT\_DATA

An update on the data status of a message for a job has been sent. The fields in order of occurrence are:

**Version number (%s)**

The version number

**Event time (%d)**

The time of the event

**jobId (%d)**

	Job ID
<b>idx (%d)</b>	Job array index
	Job array index
<b>msgIdx (%d)</b>	Index in the list
	Index in the list
<b>dataSize (%ld)</b>	Size of the data if it has any, otherwise 0
	Size of the data if it has any, otherwise 0
<b>dataStatus (%d)</b>	Status of the attached data
	Status of the attached data
<b>fileName (%s)</b>	File name of the attached data
	File name of the attached data

## JOB\_CHUNK

This is created when a job is inserted into a chunk.

If `LSF_HPC_EXTENSIONS="SHORT_EVENTFILE"` is specified in `lsf.conf`, older daemons and commands (pre-LSF Version 6.0) cannot recognize the `lsb.events` file format.

The fields in order of occurrence are:

<b>Version number (%s)</b>	The version number
	The version number
<b>Event time (%d)</b>	The time of the event
	The time of the event
<b>membSize (%ld)</b>	Size of array <code>membJobId</code>
	Size of array <code>membJobId</code>
<b>membJobId (%ld)</b>	Job IDs of jobs in the chunk
	Job IDs of jobs in the chunk
<b>numExHosts (%ld)</b>	Number of execution hosts
	Number of execution hosts
	If <code>LSF_HPC_EXTENSIONS="SHORT_EVENTFILE"</code> is specified in <code>lsf.conf</code> , the value of this field is the number of <code>.hosts</code> listed in the <code>execHosts</code> field.
<b>execHosts (%s)</b>	Execution host name array
	Execution host name array
	If <code>LSF_HPC_EXTENSIONS="SHORT_EVENTFILE"</code> is specified in <code>lsf.conf</code> , the value of this field is logged in a shortened format.
	If <code>LSF_HPC_EXTENSIONS="SHORT_EVENTFILE"</code> is specified in <code>lsf.conf</code> , the value of this field is logged in a shortened format.

## SBD\_UNREPORTED\_STATUS

This is created when an unreported status change occurs. The fields in order of occurrence are:

### Version number (%s)

The version number

### Event time (%d)

The time of the event

### jobId (%d)

Job ID

### actPid (%d)

Acting processing ID

### jobPid (%d)

Job process ID

### jobPGid (%d)

Job process group ID

### newStatus (%d)

New status of the job

### reason (%d)

Pending or suspending reason code, see <l sf /l sbat ch. h>

### suspreason (%d)

Pending or suspending subreason code, see <l sf /l sbat ch. h>

### lsfRusage

The following fields contain resource usage information for the job (see `getrusage(2)`). If the value of some field is unavailable (due to job exit or the difference among the operating systems), -1 will be logged. Times are measured in seconds, and sizes are measured in KB.

#### ru\_utime (%f)

User time used

#### ru\_stime (%f)

System time used

#### ru\_maxrss (%f)

Maximum shared text size

#### ru\_ixrss (%f)

Integral of the shared text size over time (in KB seconds)

#### ru\_ismrss (%f)

	Integral of the shared memory size over time (valid only on Ultrix)
<b>ru_idrss (%f)</b>	Integral of the unshared data size over time
<b>ru_isrss (%f)</b>	Integral of the unshared stack size over time
<b>ru_minflt (%f)</b>	Number of page reclaims
<b>ru_majflt (%f)</b>	Number of page faults
<b>ru_nswap (%f)</b>	Number of times the process was swapped out
<b>ru_inblock (%f)</b>	Number of block input operations
<b>ru_oublock (%f)</b>	Number of block output operations
<b>ru_ioch (%f)</b>	Number of characters read and written (valid only on HP-UX)
<b>ru_msgsnd (%f)</b>	Number of System V IPC messages sent
<b>ru_msgrcv (%f)</b>	Number of messages received
<b>ru_nsignals (%f)</b>	Number of signals received
<b>ru_nvcsw (%f)</b>	Number of voluntary context switches
<b>ru_nivcsw (%f)</b>	Number of involuntary context switches
<b>ru_exutime (%f)</b>	Exact user time used (valid only on ConvexOS)
<b>exitStatus (%d)</b>	Exit status of the job, see <l sf/l sbat ch. h>
<b>execCwd (%s)</b>	

Current working directory job used on execution host (up to 4094 characters for UNIX or 255 characters for Windows)

**execHome (%s)**

Home directory job used on execution host

**execUsername (%s)**

Mapped user name on execution host

**msgId (%d)**

ID of the message

**actStatus (%d)**

Action status

1: Action started

2: One action preempted other actions

3: Action succeeded

4: Action Failed

**sigValue (%d)**

Signal value

**seq (%d)**

Sequence status of the job

**idx (%d)**

Job array index

**jRusage**

The following fields contain resource usage information for the job. If the value of some field is unavailable (due to job exit or the difference among the operating systems), -1 will be logged. Times are measured in seconds, and sizes are measured in KB.

**mem (%d)**

Total resident memory usage in KB of all currently running processes in a given process group

**swap (%d)**

Totally virtual memory usage in KB of all currently running processes in given process groups

**utime (%d)**

Cumulative total user time in seconds

**stime (%d)**

Cumulative total system time in seconds

<b>npids (%d)</b>	Number of currently active process in given process groups. This entry has four sub-fields:
<b>pid (%d)</b>	Process ID of the child sbat chd that initiated the action
<b>ppid (%d)</b>	Parent process ID
<b>pgid (%d)</b>	Process group ID
<b>jobId (%d)</b>	Process Job ID
<b>npgids (%d)</b>	Number of currently active process groups
<b>exitInfo (%d)</b>	Job termination reason, see <1 sf /1 sbat ch. h>

## PRE\_EXEC\_START

A pre-execution command has been started.

The fields in order of occurrence are:

<b>Version number (%s)</b>	The version number
<b>Event time (%d)</b>	The time of the event
<b>jobId (%d)</b>	Job ID
<b>jStatus (%d)</b>	Job status, (4, indicating the RUN status of the job)
<b>jobPid (%d)</b>	Job process ID
<b>jobPGid (%d)</b>	Job process group ID
<b>hostFactor (%f)</b>	CPU factor of the first execution host
<b>numExHosts (%d)</b>	

Number of processors used for execution

**execHosts (%s)**

List of execution host names

**queuePreCmd (%s)**

Pre-execution command

**queuePostCmd (%s)**

Post-execution command

**jFlags (%d)**

Job processing flags

**userGroup (%s)**

User group name

**idx (%d)**

Job array index

**additionalInfo (%s)**

Placement information of HPC jobs

## JOB\_FORCE

A job has been forced to run with brun.

**Version number (%s)**

The version number

**Event time (%d)**

The time of the event

**jobId (%d)**

Job ID

**userId (%d)**

UNIX user ID of the user invoking the command

**idx (%d)**

Job array index

**options (%d)**

Bit flags for job processing

**numExecHosts (%d)**

Number of execution hosts

If `LSF_HPC_EXTENSIONS="SHORT_EVENTFILE"` is specified in `lsf.conf`, the value of this field is the number of `.hosts` listed in the `execHosts` field.

**execHosts (%s)**

Execution host name array

If LSF\_HPC\_EXTENSIONS="SHORT\_EVENTFILE" is specified in lsf.conf, the value of this field is logged in a shortened format.

**userName (%s)**

Name of the user

**queue (%s)**

Name of queue if a remote brun job ran; otherwise, this field is empty

## GRP\_ADD

This is created when a job group is added. The fields in order of occurrence are:

**Version number (%s)**

The version number

**Event time (%d)**

The time of the event

**userId (%d)**

UNIX user ID of the job group owner

**submitTime (%d)**

Job submission time

**userName (%s)**

User name of the job group owner

**depCond (%s)**

Job dependency condition

**timeEvent (%d)**

Time Event, for job dependency condition; specifies when time event ended

**groupSpec (%s)**

Job group name

**delOptions (%d)**

Delete options for the options field

**delOptions2 (%d)**

Delete options for the options2 field

**sla (%s)**

SLA service class name that the job group is to be attached to

**maxJLimit (%d)**

Job group limit set by bgadd -L

**groupType (%d)**

Job group creation method:

- 0x01 - job group was created explicitly
- 0x02 - job group was created implicitly

## GRP\_MOD

This is created when a job group is modified. The fields in order of occurrence are:

**Version number (%s)**

The version number

**Event time (%d)**

The time of the event

**userId (%d)**

UNIX user ID of the job group owner

**submitTime (%d)**

Job submission time

**userName (%s)**

User name of the job group owner

**depCond (%s)**

Job dependency condition

**timeEvent (%d)**

Time Event, for job dependency condition; specifies when time event ended

**groupSpec (%s)**

Job group name

**delOptions (%d)**

Delete options for the options field

**delOptions2 (%d)**

Delete options for the options2 field

**sla (%s)**

SLA service class name that the job group is to be attached to

**maxJLimit (%d)**

Job group limit set by bgmod -L

## LOG\_SWITCH

This is created when switching the event file lsb.events. The fields in order of occurrence are:

**Version number (%s)**

The version number

**Event time (%d)**

The time of the event

**jobId (%d)**

Job ID

## JOB\_RESIZE\_NOTIFY\_START

LSF logs this event when a resize (shrink or grow) request has been sent to the first execution host. The fields in order of occurrence are:

**Version number (%s)**

The version number.

**Event time (%d)**

The time of the event.

**jobId (%d)**

The job ID.

**idx (%d)**

Job array index.

**notifyId (%d)**

Identifier or handle for notification.

**numResizeHosts (%d)**

Number of processors used for execution. If `LSF_HPC_EXTENSIONS="SHORT_EVENTFILE"` is specified in `lsf.conf`, the value of this field is the number of hosts listed in short format.

**resizeHosts (%s)**

List of execution host names. If `LSF_HPC_EXTENSIONS="SHORT_EVENTFILE"` is specified in `lsf.conf`, the value of this field is logged in a shortened format.

## JOB\_RESIZE\_NOTIFY\_ACCEPT

LSF logs this event when a resize request has been accepted from the first execution host of a job. The fields in order of occurrence are:

**Version number (%s)**

The version number.

**Event time (%d)**

The time of the event.

**jobId (%d)**

	The job ID.
<b>idx (%d)</b>	Job array index.
<b>notifyId (%d)</b>	Identifier or handle for notification.
<b>resizeNotifyCmdPid (%d)</b>	Resize notification executable process ID. If no resize notification executable is defined, this field will be set to 0.
<b>resizeNotifyCmdPGid (%d)</b>	Resize notification executable process group ID. If no resize notification executable is defined, this field will be set to 0.
<b>status (%d)</b>	Status field used to indicate possible errors. 0 Success, 1 failure.

## JOB\_RESIZE\_NOTIFY\_DONE

LSF logs this event when the resize notification command completes. The fields in order of occurrence are:

<b>Version number (%s)</b>	The version number.
<b>Event time (%d)</b>	The time of the event.
<b>jobId (%d)</b>	The job ID.
<b>idx (%d)</b>	Job array index.
<b>notifyId (%d)</b>	Identifier or handle for notification.
<b>status (%d)</b>	Resize notification exit value. (0, success, 1, failure, 2 failure but cancel request.)

## JOB\_RESIZE\_RELEASE

LSF logs this event when receiving resource release request from client. The fields in order of occurrence are:

<b>Version number (%s)</b>	The version number.
<b>Event time (%d)</b>	

	The time of the event.
<b>jobId (%d)</b>	The job ID.
<b>idx (%d)</b>	Job array index.
<b>reqid (%d)</b>	Request Identifier or handle.
<b>options (%d)</b>	Release options.
<b>userId (%d)</b>	UNIX user ID of the user invoking the command.
<b>userName (%s)</b>	User name of the submitter.
<b>resizeNotifyCmd (%s)</b>	Resize notification command to run on the first execution host to inform job of a resize event.
<b>numResizeHosts (%d)</b>	Number of processors used for execution during resize. If LSF_HPC_EXTENSIONS="SHORT_EVENTFILE" is specified in lsf.conf, the value of this field is the number of hosts listed in short format.
<b>resizeHosts (%s)</b>	List of execution host names during resize. If LSF_HPC_EXTENSIONS="SHORT_EVENTFILE" is specified in lsf.conf, the value of this field is logged in a shortened format.

## JOB\_RESIZE\_CANCEL

LSF logs this event when receiving cancel request from client. The fields in order of occurrence are:

<b>Version number (%s)</b>	The version number.
<b>Event time (%d)</b>	The time of the event.
<b>jobId (%d)</b>	The job ID.
<b>idx (%d)</b>	Job array index.

lsb.events

**userId (%d)**

UNIX user ID of the user invoking the command.

**userName (%s)**

User name of the submitter.

# lsb.hosts

The `lsb.hosts` file contains host-related configuration information for the server hosts in the cluster. It is also used to define host groups, host partitions, and compute units.

This file is optional. All sections are optional.

By default, this file is installed in `LSB_CONFDIR/cluster_name/configdir`.

## Changing lsb.hosts configuration

After making any changes to `lsb.hosts`, run `badminton reconfig` to reconfigure `mbatchd`.

## Host section

### Description

Optional. Defines the hosts, host types, and host models used as server hosts, and contains per-host configuration information. If this section is not configured, LSF uses all hosts in the cluster (the hosts listed in `lsf.cluster.cluster_name`) as server hosts.

Each host, host model or host type can be configured to:

- Limit the maximum number of jobs run in total
- Limit the maximum number of jobs run by each user
- Run jobs only under specific load conditions
- Run jobs only under specific time windows

The entries in a line for a host override the entries in a line for its model or type.

When you modify the cluster by adding or removing hosts, no changes are made to `lsb.hosts`. This does not affect the default configuration, but if hosts, host models, or host types are specified in this file, you should check this file whenever you make changes to the cluster and update it manually if necessary.

### Host section structure

The first line consists of keywords identifying the load indices that you wish to configure on a per-host basis. The keyword `HOST_NAME` must be used; the others are optional. Load indices not listed on the keyword line do not affect scheduling decisions.

Each subsequent line describes the configuration information for one host, host model or host type. Each line must contain one entry for each keyword. Use empty parentheses `()` or a dash `-` to specify the default value for an entry.

### HOST\_NAME

*Required.* Specify the name, model, or type of a host, or the keyword `default`.

#### host name

The name of a host defined in `lsf.cluster.cluster_name`.

#### host model

A host model defined in `lsf.shared`.

lsb.hosts

## host type

A host type defined in `lsf.shared`.

## default

The reserved host name `default` indicates all hosts in the cluster not otherwise referenced in the section (by name or by listing its model or type).

## CHKPNT

### Description

If `C`, checkpoint copy is enabled. With checkpoint copy, all opened files are automatically copied to the checkpoint directory by the operating system when a process is checkpointed.

### Example

```
HOST_NAME  CHKPNT hostA      C
```

### Compatibility

Checkpoint copy is only supported on Cray systems.

### Default

No checkpoint copy

## DISPATCH\_WINDOW

### Description

The time windows in which jobs from this host, host model, or host type are dispatched. Once dispatched, jobs are no longer affected by the dispatch window.

### Default

Not defined (always open)

## EXIT\_RATE

### Description

Specifies a threshold for exited jobs. Specify a number of jobs. If the number of jobs that exit over a period of time specified by `JOB_EXIT_RATE_DURATION` in `lsf.params` (5 minutes by default) exceeds the number of jobs you specify as the threshold in this parameter, LSF invokes `LSF_SERVERDIR/eadmin` to trigger a host exception.

`EXIT_RATE` for a specific host overrides a default `GLOBAL_EXIT_RATE` specified in `lsf.params`.

## Example

The following Host section defines a job exit rate of 20 jobs for all hosts, and an exit rate of 10 jobs on hostA.

```
Begin Host
HOST_NAME    MXJ      EXIT_RATE  # Keywords
Default      !          20
hostA        !          10
End Host
```

## Default

Not defined

## JL/U

## Description

Per-user job slot limit for the host. Maximum number of job slots that each user can use on this host.

## Example

```
HOST_NAME    JL/U
hostA        2
```

## Default

Unlimited

## MIG

## Syntax

**MIG**=*minutes*

## Description

Enables automatic job migration and specifies the migration threshold for checkpointable or rerunnable jobs, in minutes.

LSF automatically migrates jobs that have been in the SSUSP state for more than the specified number of minutes. Specify a value of 0 to migrate jobs immediately upon suspension. The migration threshold applies to all jobs running on the host.

Job-level command line migration threshold overrides threshold configuration in application profile and queue. Application profile configuration overrides queue level configuration. When a host migration threshold is specified, and is lower than the value for the job, the queue, or the application, the host value is used.

Does not affect MultiCluster jobs that are forwarded to a remote cluster.

## Default

Not defined. LSF does not migrate checkpointable or rerunnable jobs automatically.

## MXJ

### Description

The number of job slots on the host.

With MultiCluster resource leasing model, this is the number of job slots on the host that are available to the local cluster.

Use “!” to make the number of job slots equal to the number of CPUs on a host.

For the reserved host name default, “!” makes the number of job slots equal to the number of CPUs on all hosts in the cluster not otherwise referenced in the section.

By default, the number of running and suspended jobs on a host cannot exceed the number of job slots. If preemptive scheduling is used, the suspended jobs are not counted as using a job slot.

On multiprocessor hosts, to fully use the CPU resource, make the number of job slots equal to or greater than the number of processors.

### Default

Unlimited

### *load\_index*

### Syntax

```
load_index loadSched[/loadStop]
```

Specify *io*, *it*, *ls*, *mem*, *pg*, *r15s*, *r1m*, *r15m*, *swp*, *tmp*, *ut*, or a non-shared custom external load index as a column. Specify multiple columns to configure thresholds for multiple load indices.

### Description

Scheduling and suspending thresholds for dynamic load indices supported by LIM, including external load indices.

Each load index column must contain either the default entry or two numbers separated by a slash ‘/’, with no white space. The first number is the scheduling threshold for the load index; the second number is the suspending threshold.

Queue-level scheduling and suspending thresholds are defined in `lsb.queues`. If both files specify thresholds for an index, those that apply are the most restrictive ones.

### Example

```
HOST_NAME    mem    swp
hostA        100/10 200/30
```

This example translates into a `loadSched` condition of

```
mem>=100 && swp>=200
```

and a `loadStop` condition of

```
mem < 10 || swp < 30
```

## Default

Not defined

## Example of a Host section

```

Begin Host
HOST_NAME  MXJ  JL/U r1m          pg          DISPATCH_WINDOW
hostA      1    -   0.6/1.6   10/20   (5:19:00-1:8:30 20:00-8:30)
SUNSOL    1    -   0.5/2.5   -          23:00-8:00
default   2    1   0.6/1.6   20/40          ()
End Host

```

SUNSOL is a host type defined in `lsf.shared`. This example Host section configures one host and one host type explicitly and configures default values for all other load-sharing hosts.

HostA runs one batch job at a time. A job will only be started on hostA if the `r1m` index is below 0.6 and the `pg` index is below 10; the running job is stopped if the `r1m` index goes above 1.6 or the `pg` index goes above 20. HostA only accepts batch jobs from 19:00 on Friday evening until 8:30 Monday morning and overnight from 20:00 to 8:30 on all other days.

For hosts of type SUNSOL, the `pg` index does not have host-specific thresholds and such hosts are only available overnight from 23:00 to 8:00.

The entry with host name default applies to each of the other hosts in the cluster. Each host can run up to two jobs at the same time, with at most one job from each user. These hosts are available to run jobs at all times. Jobs may be started if the `r1m` index is below 0.6 and the `pg` index is below 20, and a job from the lowest priority queue is suspended if `r1m` goes above 1.6 or `pg` goes above 40.

## HostGroup section

### Description

Optional. Defines host groups.

The name of the host group can then be used in other host group, host partition, and queue definitions, as well as on the command line. Specifying the name of a host group has exactly the same effect as listing the names of all the hosts in the group.

### Structure

Host groups are specified in the same format as user groups in `lsb.users`.

The first line consists of two mandatory keywords, `GROUP_NAME` and `GROUP_MEMBER`, as well as optional keywords, `CONDENSE` and `GROUP_ADMIN`. Subsequent lines name a group and list its membership.

The sum of all host groups, compute groups, and host partitions cannot be more than 1024.

## GROUP\_NAME

### Description

An alphanumeric string representing the name of the host group.

You cannot use the reserved name `all`, and group names must not conflict with host names.

## CONDENSE

### Description

Optional. Defines condensed host groups.

Condensed host groups are displayed in a condensed output format for the `bhosts` and `bjobs` commands.

If you configure a host to belong to more than one condensed host group, `bjobs` can display any of the host groups as execution host name.

### Valid values

Y or N.

### Default

N (the specified host group is not condensed)

## GROUP\_MEMBER

### Description

A space-delimited list of host names or previously defined host group names, enclosed in one pair of parentheses.

You cannot use more than one pair of parentheses to define the list.

The names of hosts and host groups can appear on multiple lines because hosts can belong to multiple groups. The reserved name `all` specifies all hosts in the cluster. An exclamation mark (!) indicates an externally-defined host group, which the `egroup` executable retrieves.

### Pattern definition

You can use string literals and special characters when defining host group members. Each entry cannot contain any spaces, as the list itself is space delimited.

When a leased-in host joins the cluster, the host name is in the form of `host@cluster`. For these hosts, only the host part of the host name is subject to pattern definitions.

You can use the following special characters to specify host group members:

- Use a tilde (~) to exclude specified hosts or host groups from the list.
- Use an asterisk (\*) as a wildcard character to represent any number of characters.
- Use square brackets with a hyphen (`[integer1 - integer2]`) to define a range of non-negative integers at the end of a host name. The first integer must be less than the second integer.
- Use square brackets with commas (`[integer1, integer2...]`) to define individual non-negative integers at the end of a host name.
- Use square brackets with commas and hyphens (for example, `[integer1 - integer2, integer3, integer4 - integer5]`) to define different ranges of non-negative integers at the end of a host name.

### Restrictions

- You cannot use more than one set of square brackets in a single host group definition.

- The following example is *not* correct:  

```
... (hostA[ 1- 10]B[ 1- 20] hostC[ 101- 120])
```
- The following example is correct:  

```
... (hostA[ 1- 20] hostC[ 101- 120])
```
- You cannot define subgroups that contain wildcards and special characters.

## GROUP\_ADMIN

### Description

Host group administrators have the ability to open or close the member hosts for the group they are administering.

the GROUP\_ADMIN field is a space-delimited list of user names or previously defined user group names, enclosed in one pair of parentheses.

You cannot use more than one pair of parentheses to define the list.

The names of users and user groups can appear on multiple lines because users can belong to and administer multiple groups.

Host group administrator rights are inherited. For example, if the user admin2 is an administrator for host group hg1 and host group hg2 is a member of hg1, admin2 is also an administrator for host group hg2.

When host group administrators (who are not also cluster administrators) open or close a host, they must specify a comment with the -C option.

### Valid values

Any existing user or user group can be specified. A user group that specifies an external list is also allowed; however, in this location, you use the user group name that has been defined with (!) rather than (!) itself.

### Restrictions

- You cannot specify any wildcards or special characters (for example: \*, !, \$, #, &, ~).
- You cannot specify an external group (egroup).
- You cannot use the keyword ALL and you cannot administer any group that has ALL as its members.
- User names and user group names cannot have spaces.

## Example HostGroup sections

### Example 1

```
Begin HostGroup
GROUP_NAME  GROUP_MEMBER  GROUP_ADMIN
groupA      (hostA hostD) (user1 user10)
groupB      (hostF groupA hostK) ()
groupC      (!) ()
End HostGroup
```

This example defines three host groups:

- groupA includes hostA and hostD and can be administered by user1 and user10.

- groupB includes host F and host K, along with all hosts in groupA. It has no administrators (only the cluster administrator can control the member hosts).
- The group membership of groupC is defined externally and retrieved by the egroup executable.

## Example 2

```

Begin HostGroup
GROUP_NAME  GROUP_MEMBER  GROUP_ADMIN
groupA      (all) ()
groupB      (groupA ~hostA ~hostB) (user11 user14)
groupC      (hostX hostY hostZ) ()
groupD      (groupC ~hostX) usergroupB
groupE      (all ~groupC ~hostB) ()
groupF      (hostF groupC hostK) ()
End HostGroup

```

This example defines the following host groups:

- groupA contains all hosts in the cluster and is administered by the cluster administrator.
- groupB contains all the hosts in the cluster except for host A and host B and is administered by user11 and user14.
- groupC contains only host X, host Y, and host Z and is administered by the cluster administrator.
- groupD contains the hosts in groupC except for host X. Note that host X must be a member of host group groupC to be excluded from groupD. usergroupB is the administrator for groupD.
- groupE contains all hosts in the cluster excluding the hosts in groupC and host B and is administered by the cluster administrator.
- groupF contains host F, host K, and the 3 hosts in groupC and is administered by the cluster administrator.

## Example 3

```

Begin HostGroup
GROUP_NAME  CONDENSE  GROUP_MEMBER  GROUP_ADMIN
groupA      N          (all) ()
groupB      N          (hostA, hostB) (usergroupC user1)
groupC      Y          (all) ()
End HostGroup

```

This example defines the following host groups:

- groupA shows uncondensed output and contains all hosts in the cluster and is administered by the cluster administrator.
- groupB shows uncondensed output, and contains host A and host B. It is administered by all members of usergroupC and user1.
- groupC shows condensed output and contains all hosts in the cluster and is administered by the cluster administrator.

## Example 4

```

Begin HostGroup
GROUP_NAME CONDENSE GROUP_MEMBER GROUP_ADMIN
groupA      Y (host*) (user7)
groupB      N (*A) ()
groupC      N (hostB* ~hostB[1-50]) ()
groupD      Y (hostC[1-50] hostC[101-150]) (usergroupJ)
groupE      N (hostC[51-100] hostC[151-200]) ()
groupF      Y (hostD[1,3] hostD[5-10]) ()
groupG      N (hostD[11-50] ~hostD[15,20,25] hostD2) ()
End HostGroup

```

This example defines the following host groups:

- groupA shows condensed output, and contains all hosts starting with the string host. It is administered by user7.
- groupB shows uncondensed output, and contains all hosts ending with the string A, such as host A and is administered by the cluster administrator.
- groupC shows uncondensed output, and contains all hosts starting with the string host B except for the hosts from host B1 to host B50 and is administered by the cluster administrator.
- groupD shows condensed output, and contains all hosts from host C1 to host C50 and all hosts from host C101 to host C150 and is administered by the the members of usergroupJ.
- groupE shows uncondensed output, and contains all hosts from host C51 to host C100 and all hosts from host C151 to host C200 and is administered by the cluster administrator.
- groupF shows condensed output, and contains host D1, host D3, and all hosts from host D5 to host D10 and is administered by the cluster administrator.
- groupG shows uncondensed output, and contains all hosts from host D11 to host D50 except for host D15, host D20, and host D25. groupG also includes host D2. It is administered by the cluster administrator.

## HostPartition section

### Description

Optional. Used with host partition user-based fairshare scheduling. Defines a host partition, which defines a user-based fairshare policy at the host level.

Configure multiple sections to define multiple partitions.

The members of a host partition form a host group with the same name as the host partition.

---

#### Restriction:

You cannot use host partitions and host preference simultaneously.

---

### Limitations on queue configuration

- If you configure a host partition, you cannot configure fairshare at the queue level.
- If a queue uses a host that belongs to a host partition, it should not use any hosts that don't belong to that partition. All the hosts in the queue should belong to the same partition. Otherwise, you might notice unpredictable scheduling behavior:

- Jobs in the queue sometimes may be dispatched to the host partition even though hosts not belonging to any host partition have a lighter load.
- If some hosts belong to one host partition and some hosts belong to another, only the priorities of one host partition are used when dispatching a parallel job to hosts from more than one host partition.

## Shared resources and host partitions

- If a resource is shared among hosts included in host partitions and hosts that are not included in any host partition, jobs in queues that use the host partitions will always get the shared resource first, regardless of queue priority.
- If a resource is shared among host partitions, jobs in queues that use the host partitions listed first in the `HostPartiti on` section of `lsb. hosts` will always have priority to get the shared resource first. To allocate shared resources among host partitions, LSF considers host partitions in the order they are listed in `lsb. hosts`.

## Structure

Each host partition always consists of 3 lines, defining the name of the partition, the hosts included in the partition, and the user share assignments.

## HPART\_NAME

### Syntax

**HPART\_NAME**=*partition\_name*

### Description

Specifies the name of the partition. The name must be 59 characters or less.

## HOSTS

### Syntax

**HOSTS**=[[~]*host\_name* / [~]*host\_group* / all]...

### Description

Specifies the hosts in the partition, in a space-separated list.

A host cannot belong to multiple partitions.

A host group cannot be empty.

Hosts that are not included in any host partition are controlled by the FCFS scheduling policy instead of the fairshare scheduling policy.

Optionally, use the reserved host name `all` to configure a single partition that applies to all hosts in a cluster.

Optionally, use the not operator (`~`) to exclude hosts or host groups from the list of hosts in the host partition.

## Examples

`HOSTS=all ~hostK ~hostM`

The partition includes all the hosts in the cluster, except for hostK and hostM.

```
HOSTS=groupA ~hostL
```

The partition includes all the hosts in host group groupA except for hostL.

## USER\_SHARES

### Syntax

```
USER_SHARES=[user, number_shares]...
```

### Description

Specifies user share assignments

- Specify at least one user share assignment.
- Enclose each user share assignment in square brackets, as shown.
- Separate a list of multiple share assignments with a space between the square brackets.
- *user*—Specify users who are also configured to use the host partition. You can assign the shares:
  - To a single user (specify *user\_name*). To specify a Windows user account, include the domain name in uppercase letters (*DOMAIN\_NAME\user\_name*).
  - To users in a group, individually (specify *group\_name@*) or collectively (specify *group\_name*). To specify a Windows user group, include the domain name in uppercase letters (*DOMAIN\_NAME\group\_name*).
  - To users not included in any other share assignment, individually (specify the keyword `default`) or collectively (specify the keyword `others`).

By default, when resources are assigned collectively to a group, the group members compete for the resources according to FCFS scheduling. You can use hierarchical fairshare to further divide the shares among the group members.

When resources are assigned to members of a group individually, the share assignment is recursive. Members of the group and of all subgroups always compete for the resources according to FCFS scheduling, regardless of hierarchical fairshare policies.

- *number\_shares*
  - Specify a positive integer representing the number of shares of the cluster resources assigned to the user.
  - The number of shares assigned to each user is only meaningful when you compare it to the shares assigned to other users or to the total number of shares. The total number of shares is just the sum of all the shares assigned in each share assignment.

### Example of a HostPartition section

```
Begin HostPartition
HPART_NAME = Partition1 HOSTS = hostA hostB USER_SHARES = [groupA@, 3] [groupB, 7]
[default, 1]
End HostPartition
```

## ComputeUnit section

### Description

Optional. Defines compute units.

Once defined, the compute unit can be used in other compute unit and queue definitions, as well as in the command line. Specifying the name of a compute unit has the same effect as listing the names of all the hosts in the compute unit.

Compute units are similar to host groups, with the added feature of granularity allowing the construction of structures that mimic the network architecture. Job scheduling using compute unit resource requirements effectively spreads jobs over the cluster based on the configured compute units.

To enforce consistency, compute unit configuration has the following requirements:

- Hosts and host groups appear in the finest granularity compute unit type, and nowhere else.
- Hosts appear in only one compute unit of the finest granularity.
- All compute units of the same type have the same type of compute units (or hosts) as members.

## Structure

Compute units are specified in the same format as host groups in `lsb.hosts`.

The first line consists of three mandatory keywords, `NAME`, `MEMBER`, and `TYPE`, as well as optional keywords `CONDENSE` and `ADMIN`. Subsequent lines name a compute unit and list its membership.

The sum of all host groups, compute groups, and host partitions cannot be more than 1024.

## NAME

### Description

An alphanumeric string representing the name of the compute unit.

You cannot use the reserved names `all`, `all remote`, `others`, and `default`. Compute unit names must not conflict with host names, host partitions, or host group names.

## CONDENSE

### Description

Optional. Defines condensed compute units.

Condensed compute units are displayed in a condensed output format for the `bhosts` and `bjobs` commands. The condensed compute unit format includes the slot usage for each compute unit.

### Valid values

Y or N.

### Default

N (the specified host group is not condensed)

## MEMBER

### Description

A space-delimited list of host names or previously defined compute unit names, enclosed in one pair of parentheses.

You cannot use more than one pair of parentheses to define the list.

The names of hosts and host groups can appear only once, and only in a compute unit type of the finest granularity.

An exclamation mark (!) indicates an externally-defined host group, which the egroup executable retrieves.

## Pattern definition

You can use string literals and special characters when defining compute unit members. Each entry cannot contain any spaces, as the list itself is space delimited.

You can use the following special characters to specify host and host group compute unit members:

- Use a tilde (~) to exclude specified hosts or host groups from the list.
- Use an asterisk (\*) as a wildcard character to represent any number of characters.
- Use square brackets with a hyphen (*[integer1 - integer2]*) to define a range of non-negative integers at the end of a host name. The first integer must be less than the second integer.
- Use square brackets with commas (*[integer1, integer2...]*) to define individual non-negative integers at the end of a host name.
- Use square brackets with commas and hyphens (for example, *[integer1 - integer2, integer3, integer4 - integer5]*) to define different ranges of non-negative integers at the end of a host name.

## Restrictions

- You cannot use more than one set of square brackets in a single compute unit definition.

- The following example is *not* correct:

```
... (encl A[ 1- 10] B[ 1- 20] encl C[ 101- 120])
```

- The following example is correct:

```
... (encl A[ 1- 20] encl C[ 101- 120])
```

- Compute unit names cannot be used in compute units of the finest granularity.
- You cannot include host or host group names except in compute units of the finest granularity.
- You must not skip levels of granularity. For example:

If lsb.params contains COMPUTE\_UNIT\_TYPES=encl osure rack cabi net then a compute unit of type cabi net can contain compute units of type rack, but not of type encl osure.

- The keywords al l, al l remote, al l @cl uster, other and defaul t cannot be used when defining compute units.

## TYPE

### Description

The type of the compute unit, as defined in the COMPUTE\_UNIT\_TYPES parameter of lsb.params.

## ADMIN

### Description

Compute unit administrators have the ability to open or close the member hosts for the compute unit they are administering.

the ADMIN field is a space-delimited list of user names or previously defined user group names, enclosed in one pair of parentheses.

You cannot use more than one pair of parentheses to define the list.

The names of users and user groups can appear on multiple lines because users can belong to and administer multiple compute units.

Compute unit administrator rights are inherited. For example, if the user admin2 is an administrator for compute unit cu1 and compute unit cu2 is a member of cu1, admin2 is also an administrator for compute unit cu2.

When compute unit administrators (who are not also cluster administrators) open or close a host, they must specify a comment with the -C option.

## Valid values

Any existing user or user group can be specified. A user group that specifies an external list is also allowed; however, in this location, you use the user group name that has been defined with (!) rather than (!) itself.

## Restrictions

- You cannot specify any wildcards or special characters (for example: \*, !, \$, #, &, ~).
- You cannot specify an external group (egroup).
- You cannot use the keyword ALL and you cannot administer any group that has ALL as its members.
- User names and user group names cannot have spaces.

## Example ComputeUnit sections

### Example 1

```
(For the l sb. params entry
COMPUTE_UNIT_TYPES=encl osure rack cabi net
)
Begin ComputeUnit
NAME    MEMBER          TYPE
encl 1  (host 1 host 2)  encl osure
encl 2  (host 3 host 4)  encl osure
encl 3  (host 5 host 6)  encl osure
encl 4  (host 7 host 8)  encl osure
rack1   (encl 1 encl 2)   rack
rack2   (encl 3 encl 4)   rack
cbnt 1  (rack1 rack2)    cabi net
End ComputeUnit
```

This example defines seven compute units:

- encl 1, encl 2, encl 3 and encl 4 are the finest granularity, and each contain two hosts.
- rack1 is of coarser granularity and contains two levels. At the enclosure level rack1 contains encl 1 and encl 2. At the lowest level rack1 contains host 1, host 2, host 3, and host 4.
- rack2 has the same structure as rack1, and contains encl 3 and encl 4.
- cbnt 1 contains two racks (rack1 and rack2), four enclosures (encl 1, encl 2, encl 3, and encl 4) and all eight hosts. Compute unit cbnt 1 is the coarsest granularity in this example.

## Example 2

(For the `lsb.params` entry `COMPUTE_UNIT_TYPES=encl osure rack cabi net`)

Begin ComputeUnit

NAME	CONDENSE	MEMBER	TYPE	ADMIN
encl 1	Y	(hg123 ~hostA ~hostB)	encl osure	(user11 user14)
encl 2	Y	(hg456)	encl osure	()
encl 3	N	(hostA hostB)	encl osure	usergroupB
encl 4	N	(hgroupX ~hostB)	encl osure	()
encl 5	Y	(hostC* ~hostC[ 101- 150])	encl osure	usergroupJ
encl 6	N	(hostC[ 101- 150])	encl osure	()
rack1	Y	(encl 1 encl 2 encl 3)	rack	()
rack2	N	(encl 4 encl 5)	rack	usergroupJ
rack3	N	(encl 6)	rack	()
cbnt1	Y	(rack1 rack2)	cabi net	()
cbnt2	N	(rack3)	cabi net	user14

End ComputeUnit

This example defines 11 compute units:

- All six enclosures (finest granularity) contain only hosts and host groups. All three racks contain only enclosures. Both cabinets (coarsest granularity) contain only racks.
- `encl 1` contains all the hosts in host group `hg123` except for `hostA` and `hostB` and is administered by `user11` and `user14`. Note that `hostA` and `hostB` must be members of host group `hg123` to be excluded from `encl 1`. `encl 1` shows condensed output.
- `encl 2` contains host group `hg456` and is administered by the cluster administrator. `encl 2` shows condensed output.
- `encl 3` contains `hostA` and `hostB`. `usergroupB` is the administrator for `encl 3`. `encl 3` shows uncondensed output.
- `encl 4` contains host group `hgroupX` except for `hostB`. Since each host can appear in only one enclosure and `hostB` is already in `encl 3`, it cannot be in `encl 4`. `encl 4` is administered by the cluster administrator. `encl 4` shows uncondensed output.
- `encl 5` contains all hosts starting with the string `hostC` except for hosts `hostC101` to `hostC150`, and is administered by `usergroupJ`. `encl 5` shows condensed output.
- `rack1` contains `encl 1`, `encl 2`, and `encl 3`. `rack1` shows condensed output.
- `rack2` contains `encl 4`, and `encl 5`. `rack2` shows uncondensed output.
- `rack3` contains `encl 6`. `rack3` shows uncondensed output.
- `cbnt1` contains `rack1` and `rack2`. `cbnt 1` shows condensed output.
- `cbnt 2` contains `rack3`. Even though `rack3` only contains `encl6`, `cbnt 3` cannot contain `encl 6` directly because that would mean skipping the level associated with compute unit type `rack`. `cbnt 2` shows uncondensed output.

## Automatic time-based configuration

Variable configuration is used to automatically change LSF configuration based on time windows. You define automatic configuration changes in `lsb.hosts` by using `if-else` constructs and time expressions. After you change the files, reconfigure the cluster with the `badmi n reconfi g` command.

The expressions are evaluated by LSF every 10 minutes based on `mbat chd start time`. When an expression evaluates true, LSF dynamically changes the configuration based on the associated configuration

statements. Reconfiguration is done in real time without restarting `mbat chd`, providing continuous system availability.

## Example

In the following example, the `#if`, `#else`, `#endif` are not interpreted as comments by LSF but as if-else constructs.

```
Begin Host
HOST_NAME    r15s    r1m    pg
host1        3/5     3/5     12/20
#if time(5:16:30-1:8:30 20:00-8:30)
host2        3/5     3/5     12/20
#else
Ohost2       2/3     2/3     10/12
#endif
host3        3/5     3/5     12/20
End Host
```

# lsb.modules

The `lsb.modules` file contains configuration information for LSF scheduler and resource broker modules. The file contains only one section, named `PluginModule`.

This file is optional. If no scheduler or resource broker modules are configured, LSF uses the default scheduler plugin modules named `schmod_default` and `schmod_fcfs`.

The `lsb.modules` file is stored in the directory `LSB_CONFDIR/cluster_name/confdir`, where `LSB_CONFDIR` is defined in `lsf.conf`.

## Changing lsb.modules configuration

After making any changes to `lsb.modules`, run `badminton reconfig` to reconfigure `mbatchd`.

## PluginModule section

### Description

Defines the plugin modules for the LSF scheduler and LSF resource broker. If this section is not configured, LSF uses the default scheduler plugin modules named `schmod_default` and `schmod_fcfs`, which enable the LSF default scheduling features.

### Example PluginModule section

The following `PluginModule` section enables all scheduling policies provided by LSF:

```
Begin PluginModule
SCH_PLUGIN          RB_PLUGIN          SCH_DISABLE_PHASES
schmod_default      ()                  ()
schmod_fairshare    ()                  ()
schmod_fcfs         ()                  ()
schmod_limit        ()                  ()
schmod_parallel     ()                  ()
schmod_reserve      ()                  ()
schmod_preemption  ()                  ()
schmod_advrsv       ()                  ()
schmod_mc           ()                  ()
schmod_jobweight    ()                  ()
schmod_cpuset       ()                  ()
schmod_pset         ()                  ()
schmod_ps           ()                  ()
schmod_aps          ()                  ()
End PluginModule
```

### PluginModule section structure

The first line consists of the following keywords:

- `SCH_PLUGIN`

- RB\_PLUGIN
- SCH\_DISABLE\_PHASES

They identify the scheduler plugins, resource broker plugins, and the scheduler phase to be disabled for the plugins that you wish to configure.

Each subsequent line describes the configuration information for one scheduler plugin module, resource broker plugin module, and scheduler phase, if any, to be disabled for the plugin. Each line must contain one entry for each keyword. Use empty parentheses ( ) or a dash (-) to specify the default value for an entry.

## SCH\_PLUGIN

### Description

Required. The SCH\_PLUGIN column specifies the shared module name for the LSF scheduler plugin. Each plugin requires a corresponding license. Scheduler plugins are called in the order they are listed in the PluginModule section.

By default, all shared modules for scheduler plugins are located in LSF\_LIBDIR. On UNIX, you can also specify a full path to the name of the scheduler plugin.

The following modules are supplied with LSF:

#### schmod\_default

Enables the default LSF scheduler features.

Licensed by: LSF\_Manager

#### schmod\_fcfs

Enables the first-come, first-served (FCFS) scheduler features. `schmod_fcfs` can appear anywhere in the SCH\_PLUGIN list. By default, if `schmod_fcfs` is not configured in `lsb.modules`, it is loaded automatically along with `schmod_default`.

Source code (`sch.mod.fcfs.c`) for the `schmod_fcfs` scheduler plugin module is installed in the directory

`LSF_TOP/8.0/misc/examples/external_plugin/`

Use the LSF scheduler plugin SDK to modify the FCFS scheduler module code to suit the job scheduling requirements of your site.

See *Platform LSF Programmer's Guide* for more detailed information about writing, building, and configuring your own custom scheduler plugins.

#### schmod\_fairshare

Enables the LSF fairshare scheduling features.

#### schmod\_limit

Enables the LSF resource allocation limit features.

Licensed by: LSF\_Manager

#### schmod\_parallel

Enables scheduling of parallel jobs submitted with `bsub -n`.

## schmod\_reserve

Enables the LSF resource reservation features.

To enable processor reservation, backfill, and memory reservation for parallel jobs, you must configure both `schmod_parallel` and `schmod_reserve` in `lsb.modules`. If only `schmod_reserve` is configured, backfill and memory reservation are enabled only for sequential jobs, and processor reservation is not enabled.

## schmod\_preemption

Enables the LSF preemption scheduler features.

## schmod\_advrsv

Handles jobs that use advance reservations (`brsvadd`, `brsvs`, `brsvdel`, `bsub -U`)

## schmod\_cpuset

Handles jobs that use SGI cpusets (`bsub -ext [sched] "CPuset[cpuset_options]"`)

The `schmod_cpuset` plugin name must be configured after the standard LSF plugin names in the `PluginModule` list.

## schmod\_mc

Enables MultiCluster job forwarding

Licensed by: LSF\_MultiCluster

## schmod\_ps

Enables resource ownership functionality of EGO-enabled SLA scheduling policies

## schmod\_pset

Enables scheduling policies required for jobs that use HP-UX processor sets (pset) allocations (`bsub -ext [sched] "PSET[topology]"`)

The `schmod_pset` plugin name must be configured after the standard LSF plugin names in the `PluginModule` list.

## schmod\_aps

Enables absolute priority scheduling (APS) policies configured by `APS_PRIORITY` in `lsb.queues`.

The `schmod_aps` plugin name must be configured after the `schmod_fairshare` plugin name in the `PluginModule` list, so that the APS value can override the fairshare job ordering decision.

## schmod\_jobweight

An optional scheduler plugin module to enable Cross-Queue Job Weight scheduling policies. The `schmod_jobweight` plugin must be listed before `schmod_cpuset` and after all other scheduler plugin modules.

You should not use job weight scheduling together with fairshare scheduling or job preemption. To avoid scheduling conflicts, you should comment out `schmod_fairshare` and `schmod_preemption` in `lsb.modules`.

## Scheduler plugin SDK

Use the LSF scheduler plugin SDK to write customized scheduler modules that give you more flexibility and control over job scheduling. Enable your custom scheduling policies by configuring your modules under `SCH_PLUGIN` in the `PluginModules` section of `lsb.modules`.

The directory

```
LSF_TOP/8.0/misc/examples/external_plugin/
```

contains sample plugin code. See *Platform LSF Programmer's Guide* for more detailed information about writing, building, and configuring your own custom scheduler plugins.

## RB\_PLUGIN

### Description

`RB_PLUGIN` specifies the shared module name for resource broker plugins. Resource broker plugins collect and update job resource accounting information, and provide it to the scheduler.

Normally, for each scheduler plugin module, there is a corresponding resource broker plugin module to support it. However, the resource broker also supports multiple plugin modules for one scheduler plugin module.

For example, a fairshare policy may need more than one resource broker plugin module to support it if the policy has multiple configurations.

A scheduler plugin can have one, multiple, or none RB plugins corresponding to it.

### Example

NAME	RB_PLUGIN
<code>schmod_default</code>	<code>()</code>
<code>schmod_fairshare</code>	<code>(rb_fairshare)</code>

### Default

Undefined

## SCH\_DISABLE\_PHASES

### Description

`SCH_DISABLE_PHASES` specifies which scheduler phases, if any, to be disabled for the plugin. LSF scheduling has four phases:

1. **Preprocessing** — the scheduler checks the readiness of the job for scheduling and prepares a list of ready resource seekers. It also checks the start time of a job, and evaluates any job dependencies.
2. **Match/limit** — the scheduler evaluates the job resource requirements and prepares candidate hosts for jobs by matching jobs with resources. It also applies resource allocation limits. Jobs with all required resources matched go on to order/allocation phase. Not all jobs are mapped to all potential available resources. Jobs without any matching resources will not go through the Order/Allocation Phase but can go through the Post-processing phase, where preemption may be applied to get resources the job needs to run.
3. **Order/allocation** — the scheduler sorts jobs with matched resources and allocates resources for each job, assigning job slot, memory, and other resources to the job. It also checks if the allocation satisfies all constraints defined in configuration, such as queue slot limit, deadline for the job, etc.

1. In the order phase, the scheduler applies policies such as FCFS, Fairshare and Host-partition and consider job priorities within user groups and share groups. By default, job priority within a pool of jobs from the same user is based on how long the job has been pending.
2. For resource intensive jobs (jobs requiring a lot of CPUs or a large amount of memory), resource reservation is performed so that these jobs are not starved.
3. When all the currently available resources are allocated, jobs go on to post-processing.
4. Post-processing — the scheduler prepares jobs from the order/allocation phase for dispatch and applies preemption or backfill policies to obtain resources for the jobs that have completed pre-processing or match/limit phases, but did not have resources available to enter the next scheduling phase.

Each scheduler plugin module invokes one or more scheduler phase. The processing for a give phase can be disabled or skipped if:

The plugin module does not need to do any processing for that phase or the processing has already been done by a previous plugin module in the list.

The scheduler will not invoke phases marked by SCH\_DISABLE\_PHASES when scheduling jobs.

None of the plugins provided by LSF should require phases to be disabled, but your own custom plugin modules using the scheduler SDK may need to disable one or more scheduler phases.

## Example

In the following configuration, the `schmod_custom` plugin module disables the order allocation (3) and post-processing (4) phases:

NAME	SCH_DISABLE_PHASES
<code>schmod_default</code>	( )
<code>schmod_custom</code>	( 3, 4)

## Default

Undefined

# lsb.params

The `lsb.params` file defines general parameters used by the LSF system. This file contains only one section, named Parameters. `mbatchd` uses `lsb.params` for initialization. The file is optional. If not present, the LSF-defined defaults are assumed.

Some of the parameters that can be defined in `lsb.params` control timing within the system. The default settings provide good throughput for long-running batch jobs while adding a minimum of processing overhead in the batch daemons.

This file is installed by default in `LSB_CONFDIR/cluster_name/configdir`.

## Changing lsb.params configuration

After making any changes to `lsb.params`, run `badminton reconfig` to reconfigure `mbatchd`.

## Parameters section

This section and all the keywords in this section are optional. If keywords are not present, the default values are assumed.

## Parameters set at installation

The following parameter values are set at installation for the purpose of testing a new cluster:

Begin Parameters

```

DEFAULT_QUEUE = normal    #default job queue name
MBD_SLEEP_TIME = 20       #Time used for calculating parameter values (60 secs is
default)
SBD_SLEEP_TIME = 15       #sbatchd scheduling interval (30 secs is default)
JOB_ACCEPT_INTERVAL = 1   #interval for any host to accept a job
                           #(default is 1 (one-fold of MBD_SLEEP_TIME))

```

End Parameters

With this configuration, jobs submitted to the LSF system will be started on server hosts quickly. If this configuration is not suitable for your production use, you should either remove the parameters to take the default values, or adjust them as needed.

For example, to avoid having jobs start when host load is high, increase `JOB_ACCEPT_INTERVAL` so that the job scheduling interval is longer to give hosts more time to adjust load indices after accepting jobs.

In production use, you should define `DEFAULT_QUEUE` to the `normal` queue, `MBD_SLEEP_TIME` to 60 seconds (the default), and `SBD_SLEEP_TIME` to 30 seconds (the default).

## ABS\_RUNLIMIT

### Syntax

**ABS\_RUNLIMIT=y | Y**

## Description

If set, absolute (wall-clock) run time is used instead of normalized run time for all jobs submitted with the following values:

- Run time limit specified by the `-W` option of `bsub`
- `RUNLIMIT` queue-level parameter in `l sb. queues`
- `RUNLIMIT` application-level parameter in `l sb. appl i cat i ons`
- `RUNTIME` parameter in `l sb. appl i cat i ons`

The run time estimates and limits are not normalized by the host CPU factor.

## Default

N (run limit and run time estimate are normalized)

## ACCT\_ARCHIVE\_AGE

### Syntax

**ACCT\_ARCHIVE\_AGE**=*days*

## Description

Enables automatic archiving of LSF accounting log files, and specifies the archive interval. LSF archives the current log file if the length of time from its creation date exceeds the specified number of days.

## See also

- `ACCT_ARCHIVE_SIZE` also enables automatic archiving
- `ACCT_ARCHIVE_TIME` also enables automatic archiving
- `MAX_ACCT_ARCHIVE_FILE` enables automatic deletion of the archives

## Default

-1 (Not defined; no limit to the age of `l sb. acct`)

## ACCT\_ARCHIVE\_SIZE

### Syntax

**ACCT\_ARCHIVE\_SIZE**=*kilobytes*

## Description

Enables automatic archiving of LSF accounting log files, and specifies the archive threshold. LSF archives the current log file if its size exceeds the specified number of kilobytes.

## See also

- `ACCT_ARCHIVE_SIZE` also enables automatic archiving
- `ACCT_ARCHIVE_TIME` also enables automatic archiving
- `MAX_ACCT_ARCHIVE_FILE` enables automatic deletion of the archives

## Default

-1 (Not defined; no limit to the size of `l sb. acct`)

## ACCT\_ARCHIVE\_TIME

### Syntax

**ACCT\_ARCHIVE\_TIME**=*hh:mm*

### Description

Enables automatic archiving of LSF accounting log file `lsb.acct`, and specifies the time of day to archive the current log file.

### See also

- `ACCT_ARCHIVE_SIZE` also enables automatic archiving
- `ACCT_ARCHIVE_TIME` also enables automatic archiving
- `MAX_ACCT_ARCHIVE_FILE` enables automatic deletion of the archives

### Default

Not defined (no time set for archiving `lsb.acct`)

## ADVRSV\_USER\_LIMIT

### Syntax

**ADVRSV\_USER\_LIMIT**=*integer*

### Description

Sets the number of advanced reservations each user or user group can have in the system.

### Valid values

1-10000

### Default

100

## CHUNK\_JOB\_DURATION

### Syntax

**CHUNK\_JOB\_DURATION**=*minutes*

### Description

Specifies a CPU limit, run limit, or estimated run time for jobs submitted to a chunk job queue to be chunked.

When `CHUNK_JOB_DURATION` is set, the CPU limit or run limit set at the queue level (`CPULIMIT` or `RUNLIMIT`), application level (`CPULIMIT` or `RUNLIMIT`), or job level (`-c` or `-W` `bsub` options), or the run time estimate set at the application level (`RUNTIME`) must be less than or equal to `CHUNK_JOB_DURATION` for jobs to be chunked.

If `CHUNK_JOB_DURATION` is set, jobs are *not* chunked if:

- No CPU limit, run time limit, or run time estimate is specified at any level, or
- A CPU limit, run time limit, or run time estimate is greater than the value of `CHUNK_JOB_DURATION`.

The value of `CHUNK_JOB_DURATION` is displayed by `bparams -l`.

## Examples

- `CHUNK_JOB_DURATION` is not defined:
  - Jobs with no CPU limit, run limit, or run time estimate are chunked
  - Jobs with a CPU limit, run limit, or run time estimate less than or equal to 30 are chunked
  - Jobs with a CPU limit, run limit, or run time estimate greater than 30 are *not* chunked
- `CHUNK_JOB_DURATION=90`:
  - Jobs with no CPU limit, run limit, or run time estimate are *not* chunked
  - Jobs with a CPU limit, run limit, or run time estimate less than or equal to 90 are chunked
  - Jobs with a CPU limit, run limit, or run time estimate greater than 90 are *not* chunked

## Default

-1 (Not defined.)

## CLEAN\_PERIOD

### Syntax

`CLEAN_PERIOD=seconds`

### Description

For non-repetitive jobs, the amount of time that job records for jobs that have finished or have been killed are kept in `mbatchd` core memory after they have finished.

Users can still see all jobs after they have finished using the `bj obs` command.

For jobs that finished more than `CLEAN_PERIOD` seconds ago, use the `bhist` command.

## Default

3600 (1 hour)

## COMMITTED\_RUN\_TIME\_FACTOR

### Syntax

`COMMITTED_RUN_TIME_FACTOR=number`

### Description

Used only with fairshare scheduling. Committed run time weighting factor.

In the calculation of a user's dynamic priority, this factor determines the relative importance of the committed run time in the calculation. If the `-W` option of `bsub` is not specified at job submission and a `RUNLIMIT` has not been set for the queue, the committed run time is not considered.

This parameter can also be set for an individual queue in `lsb_queues`. If defined, the queue value takes precedence.

lsb.params

## Valid values

Any positive number between 0.0 and 1.0

## Default

0.0

# COMPUTE\_UNIT\_TYPES

## Syntax

**COMPUTE\_UNIT\_TYPES**=*type1 type2...*

## Description

Used to define valid compute unit types for topological resource requirement allocation.

The order in which compute unit types appear specifies the containment relationship between types. Finer grained compute unit types appear first, followed by the coarser grained type that contains them, and so on.

At most one compute unit type in the list can be followed by an exclamation mark designating it as the default compute unit type. If no exclamation mark appears, the first compute unit type in the list is taken as the default type.

## Valid values

Any space-separated list of alphanumeric strings.

## Default

Not defined

## Example

**COMPUTE\_UNIT\_TYPES**=cell enclosure! rack

Specifies three compute unit types, with the default type enclosure. Compute units of type rack contain type enclosure, and of type enclosure contain type cell.

# CONDENSE\_PENDING\_REASONS

## Syntax

**CONDENSE\_PENDING\_REASONS**=ALL | PARTIAL |N

## Description

Set to **ALL**, condenses all host-based pending reasons into one generic pending reason. This is equivalent to setting **CONDENSE\_PENDING\_REASONS=Y**.

Set to **PARTIAL**, condenses all host-based pending reasons except shared resource pending reasons into one generic pending reason.

If enabled, you can request a full pending reason list by running the following command:

```
badmIn diagnose jobId
```

---

**Tip:**

You must be LSF administrator or a queue administrator to run this command.

---

## Examples

- **CONDENSE\_PENDING\_REASONS=ALL** If a job has no other pending reason, `bj obs -p` or `bj obs -l` displays the following:  
Individual host based reasons
- **CONDENSE\_PENDING\_REASONS=N** The pending reasons are not suppressed. Host-based pending reasons are displayed.

## Default

N

## CPU\_TIME\_FACTOR

### Syntax

**CPU\_TIME\_FACTOR**=*number*

### Description

Used only with fairshare scheduling. CPU time weighting factor.

In the calculation of a user's dynamic share priority, this factor determines the relative importance of the cumulative CPU time used by a user's jobs.

This parameter can also be set for an individual queue in `lsb.queues`. If defined, the queue value takes precedence.

## Default

0.7

## DEFAULT\_APPLICATION

### Syntax

**DEFAULT\_APPLICATION**=*application\_profile\_name*

### Description

The name of the default application profile. The application profile must already be defined in `lsb.applications`.

When you submit a job to LSF without explicitly specifying an application profile, LSF associates the job with the specified application profile.

## Default

Not defined. When a user submits a job without explicitly specifying an application profile, and no default application profile is defined by this parameter, LSF does not associate the job with any application profile.

## DEFAULT\_HOST\_SPEC

### Syntax

**DEFAULT\_HOST\_SPEC**=*host\_name* | *host\_model*

### Description

The default CPU time normalization host for the cluster.

The CPU factor of the specified host or host model will be used to normalize the CPU time limit of all jobs in the cluster, unless the CPU time normalization host is specified at the queue or job level.

### Default

Not defined

## DEFAULT\_JOBGROUP

### Syntax

**DEFAULT\_JOBGROUP**=*job\_group\_name*

### Description

The name of the default job group.

When you submit a job to LSF without explicitly specifying a job group, LSF associates the job with the specified job group. The `LSB_DEFAULT_JOBGROUP` environment variable overrides the setting of `DEFAULT_JOBGROUP`. The `bsub -g job_group_name` option overrides both `LSB_DEFAULT_JOBGROUP` and `DEFAULT_JOBGROUP`.

Default job group specification supports macro substitution for project name (%p) and user name (%u). When you specify `bsub -P project_name`, the value of %p is the specified project name. If you do not specify a project name at job submission, %p is the project name defined by setting the environment variable `LSB_DEFAULTPROJECT`, or the project name specified by `DEFAULT_PROJECT` in `lsb.params`. the default project name is `default`.

For example, a default job group name specified by `DEFAULT_JOBGROUP=/canada/%p/%u` is expanded to the value for the LSF project name and the user name of the job submission user (for example, `/canada/projects/user1`).

Job group names must follow this format:

- Job group names must start with a slash character (/). For example, `DEFAULT_JOBGROUP=/A/B/C` is correct, but `DEFAULT_JOBGROUP=A/B/C` is not correct.
- Job group names cannot end with a slash character (/). For example, `DEFAULT_JOBGROUP=/A/` is not correct.
- Job group names cannot contain more than one slash character (/) in a row. For example, job group names like `DEFAULT_JOBGROUP=/A//B` or `DEFAULT_JOBGROUP=A///B` are not correct.
- Job group names cannot contain spaces. For example, `DEFAULT_JOBGROUP=/A/B C/D` is not correct.
- Project names and user names used for macro substitution with %p and %u cannot start or end with slash character (/).
- Project names and user names used for macro substitution with %p and %u cannot contain spaces or more than one slash character (/) in a row.

- Project names or user names containing slash character (/) will create separate job groups. For example, if the project name is `canada/projects`, `DEFAULT_JOBGROUP=%p` results in a job group hierarchy `/canada/projects`.

## Example

```
DEFAULT_JOBGROUP=/canada/projects
```

## Default

Not defined. When a user submits a job without explicitly specifying job group name, and the `LSB_DEFAULT_JOBGROUP` environment variable is not defined, LSF does not associate the job with any job group.

## DEFAULT\_PROJECT

### Syntax

```
DEFAULT_PROJECT=project_name
```

### Description

The name of the default project. Specify any string.

Project names can be up to 59 characters long.

When you submit a job without specifying any project name, and the environment variable `LSB_DEFAULTPROJECT` is not set, LSF automatically assigns the job to this project.

### Default

```
default
```

## DEFAULT\_QUEUE

### Syntax

```
DEFAULT_QUEUE=queue_name ...
```

### Description

Space-separated list of candidate default queues (candidates must already be defined in `lsb.queues`).

When you submit a job to LSF without explicitly specifying a queue, and the environment variable `LSB_DEFAULTQUEUE` is not set, LSF puts the job in the first queue in this list that satisfies the job's specifications subject to other restrictions, such as requested hosts, queue status, etc.

### Default

This parameter is set at installation to `DEFAULT_QUEUE=normal`.

When a user submits a job to LSF without explicitly specifying a queue, and there are no candidate default queues defined (by this parameter or by the user's environment variable `LSB_DEFAULTQUEUE`), LSF automatically creates a new queue named `default`, using the default configuration, and submits the job to that queue.

## DEFAULT\_USER\_GROUP

### Syntax

**DEFAULT\_USER\_GROUP**=*default\_user\_group*

### Description

When **DEFAULT\_USER\_GROUP** is defined, all submitted jobs must be associated with a user group. Jobs without a user group specified will be associated with *default\_user\_group*, where *default\_user\_group* is a group configured in `lsb.users` and contains `al1` as a direct member. **DEFAULT\_USER\_GROUP** can only contain one user group.

If the default user group does not have shares assigned in a fairshare queue, jobs can still run from the default user group and are charged to the highest priority account the user can access in the queue. A job submitted to a user group without shares in a specified fairshare queue is transferred to the default user group where the job can run. A job modified or moved using `bmod` or `bswitch` may similarly be transferred to the default user group.

---

#### Note:

The default user group should be configured in most queues and have shares in most fairshare queues to ensure jobs run smoothly.

---

Jobs linked to a user group, either through the *default\_user\_group* or a user group specified at submission using `bsub -G`, allow the user group administrator to issue job control operations. User group administrator rights are configured in the `UserGroup` section `lsb.users`, under `GROUP_ADMIN`.

When **DEFAULT\_USER\_GROUP** is not defined, jobs do not require a user group association.

After adding or changing **DEFAULT\_USER\_GROUP** in `lsb.params`, use `badmadmin reconfig` to reconfigure your cluster

### Default

Not defined. When a user submits a job without explicitly specifying user group name, LSF does not associate the job with any user group.

### See also

`STRICT_UG_CONTROL`, `ENFORCE_ONE_UG_LIMIT`

## DEFAULT\_SLA\_VELOCITY

### Syntax

**DEFAULT\_SLA\_VELOCITY**=*num\_slots*

### Description

For EGO-enabled SLA scheduling, the number of slots that the SLA should request for parallel jobs running in the SLA.

By default, an EGO-enabled SLA requests slots from EGO based on the number of jobs the SLA needs to run. If the jobs themselves require more than one slot, they will remain pending. To avoid this for parallel jobs, set **DEFAULT\_SLA\_VELOCITY** to the total number of slots that are expected to be used by parallel jobs.

## Default

1

## DETECT\_IDLE\_JOB\_AFTER

### Syntax

**DETECT\_IDLE\_JOB\_AFTER**=*time\_minutes*

### Description

The minimum job run time before mbat chd reports that the job is idle.

## Default

20 (mbat chd checks if the job is idle after 20 minutes of run time)

## DISABLE\_UACCT\_MAP

### Syntax

**DISABLE\_UACCT\_MAP**=y | Y

### Description

Specify y or Y to disable user-level account mapping.

## Default

N

## EADMIN\_TRIGGER\_DURATION

### Syntax

**EADMIN\_TRIGGER\_DURATION**=*minutes*

### Description

Defines how often LSF\_SERVERDIR/eadmi n is invoked once a job exception is detected. Used in conjunction with job exception handling parameters JOB\_IDLE, JOB\_OVERRUN, and JOB\_UNDERRUN in lsb. queues.

---

**Tip:**

Tune EADMIN\_TRIGGER\_DURATION carefully. Shorter values may raise false alarms, longer values may not trigger exceptions frequently enough.

---

### Example

```
EADMIN_TRIGGER_DURATION=5
```

## Default

1 minute

## ENABLE\_DEFAULT\_EGO\_SLA

### Syntax

**ENABLE\_DEFAULT\_EGO\_SLA**=*service\_class\_name* | *consumer\_name*

### Description

The name of the default service class or EGO consumer name for EGO-enabled SLA scheduling. If the specified SLA does not exist in `lsb.serviceclasses`, LSF creates one with the specified consumer name, velocity of 1, priority of 1, and a time window that is always open.

If the name of the default SLA is not configured in `lsb.serviceclasses`, it must be the name of a valid EGO consumer.

**ENABLE\_DEFAULT\_EGO\_SLA** is required to turn on EGO-enabled SLA scheduling. All LSF resource management is delegated to Platform EGO, and all LSF hosts are under EGO control. When all jobs running in the default SLA finish, all allocated hosts are released to EGO after the default idle timeout of 120 seconds (configurable by `MAX_HOST_IDLE_TIME` in `lsb.serviceclasses`).

When you submit a job to LSF without explicitly using the `-sla` option to specify a service class name, LSF puts the job in the default service class specified by *service\_class\_name*.

### Default

Not defined. When a user submits a job to LSF without explicitly specifying a service class, and there is no default service class defined by this parameter, LSF does not attach the job to any service class.

## ENABLE\_EVENT\_STREAM

### Syntax

**ENABLE\_EVENT\_STREAM**=Y | N

### Description

Used only with event streaming for system performance analysis tools.

### Default

N (event streaming is not enabled)

## ENABLE\_EXIT\_RATE\_PER\_SLOT

### Syntax

**ENABLE\_EXIT\_RATE\_PER\_SLOT**=Y | N

### Description

Scales the actual exit rate thresholds on a host according to the number of slots on the host. For example, if **EXIT\_RATE=2** in `lsb.hosts` or **GLOBAL\_EXIT\_RATE=2** in `lsb.params`, and the host has 2 job slots, the job exit rate threshold will be 4.

### Default

N

## ENABLE\_HIST\_RUN\_TIME

### Syntax

**ENABLE\_HIST\_RUN\_TIME=y | Y | n | N**

### Description

Used only with fairshare scheduling. If set, enables the use of historical run time in the calculation of fairshare scheduling priority.

This parameter can also be set for an individual queue in lsb. queues. If defined, the queue value takes precedence.

### Default

N

## ENABLE\_HOST\_INTERSECTION

### Syntax

**ENABLE\_HOST\_INTERSECTION=Y | N**

### Description

When enabled, allows job submission to any host that belongs to the intersection created when considering the queue the job was submitted to, any advance reservation hosts, or any hosts specified by bsub -m at the time of submission.

When disabled job submission with hosts specified can be accepted only if specified hosts are a subset of hosts defined in the queue.

The following commands are affected by ENABLE\_HOST\_INTERSECTION:

- bsub
- bmod
- bmi g
- brest art
- bswi tch

If no hosts exist in the intersection, the job is rejected.

### Default

N

## ENABLE\_USER\_RESUME

### Syntax

**ENABLE\_USER\_RESUME=Y | N**

### Description

Defines job resume permissions.

When this parameter is defined:

lsb.params

- If the value is Y, users can resume their own jobs that have been suspended by the administrator.
- If the value is N, jobs that are suspended by the administrator can only be resumed by the administrator or root; users do not have permission to resume a job suspended by another user or the administrator. Administrators can resume jobs suspended by users or administrators.

## Default

N (users cannot resume jobs suspended by administrator)

# ENFORCE\_ONE\_UG\_LIMITS

## Syntax

**ENFORCE\_ONE\_UG\_LIMITS=Y | N**

Upon job submission with the -G option and when user groups have overlapping members, defines whether only the specified user group's limits (or those of any parent group) are enforced or whether the most restrictive user group limits of any overlapping user/user group are enforced.

- If the value is Y, only the limits defined for the user group that you specify with -G during job submission apply to the job, even if there are overlapping members of groups.

If you have nested user groups, the limits of a user's group parent also apply.

View existing limits by running `blimit s`.

- If the value is N and the user group has members that overlap with other user groups, the strictest possible limits (that you can view by running `blimit s`) defined for any of the member user groups are enforced for the job.

If the user group specified at submission is no longer valid when the job runs and

**ENFORCE\_ONE\_UG\_LIMIT=Y**, only the user limit is applied to the job. This can occur if the user group is deleted or the user is removed from the user group.

## Default

N

# ENFORCE\_UG\_TREE

## Syntax

**ENFORCE\_UG\_TREE=Y | N**

## Description

When **ENFORCE\_UG\_TREE=Y** is defined, user groups must form a tree-like structure, with each user group having at most one parent. User group definitions in the UserGroup section of `lsb.users` will be checked in configuration order, and any user group appearing in `GROUP_MEMBER` more than once will be ignored after the first occurrence.

After adding or changing `ENFORCE_UG_TREE` in `lsb.params`, use `badadmin reconfig` to reconfigure your cluster

## Default

N (Not defined.)

## See also

DEFAULT\_USER\_GROUP, ENFORCE\_ONE\_UG\_LIMIT, STRICT\_UG\_CONTROL

## EVENT\_STREAM\_FILE

### Syntax

**EVENT\_STREAM\_FILE**=*file\_path*

### Description

Determines the path to the event data stream file used by system performance analysis tools.

### Default

LSF\_TOP/work/*cluster\_name*/logdir/stream/l sb. stream

## EVENT\_UPDATE\_INTERVAL

### Syntax

**EVENT\_UPDATE\_INTERVAL**=*seconds*

### Description

Used with duplicate logging of event and accounting log files. LSB\_LOCALDIR in `lsf.conf` must also be specified. Specifies how often to back up the data and synchronize the directories (LSB\_SHARED\_DIR and LSB\_LOCALDIR).

If you do not define this parameter, the directories are synchronized when data is logged to the files, or when `mbatchd` is started on the first LSF master host. If you define this parameter, `mbatchd` synchronizes the directories only at the specified time intervals.

Use this parameter if NFS traffic is too high and you want to reduce network traffic.

### Valid values

1 to 2147483647

### Recommended values

Between 10 and 30 seconds, or longer depending on the amount of network traffic.

---

#### Note:

Avoid setting the value to exactly 30 seconds, because this will trigger the default behavior and cause `mbatchd` to synchronize the data every time an event is logged.

---

### Default

-1 (Not defined.)

## See also

LSB\_LOCALDIR in `lsf.conf`

## EXIT\_RATE\_TYPE

### Syntax

**EXIT\_RATE\_TYPE=[JOBEXIT | JOBEXIT\_NONLSF] [JOBINIT] [HPCINIT]**

### Description

When host exception handling is configured (EXIT\_RATE in lsb.hosts or GLOBAL\_EXIT\_RATE in lsb.params), specifies the type of job exit to be handled.

#### **JOBEXIT**

Job exited after it was dispatched and started running.

#### **JOBEXIT\_NONLSF**

Job exited with exit reasons related to LSF and not related to a host problem (for example, user action or LSF policy). These jobs are not counted in the exit rate calculation for the host.

#### **JOBINIT**

Job exited during initialization because of an execution environment problem. The job did not actually start running.

#### **HPCINIT**

HPC job exited during initialization because of an execution environment problem. The job did not actually start running.

### Default

JOBEXIT\_NONLSF

## EXTEND\_JOB\_EXCEPTION\_NOTIFY

### Syntax

**EXTEND\_JOB\_EXCEPTION\_NOTIFY=Y | y | N | n**

### Description

Sends extended information about a job exception in a notification email sent when a job exception occurs. Extended information includes:

- JOB\_ID
- RUN\_TIME
- IDLE\_FACTOR (Only applicable if the job has been idle.)
- USER
- QUEUE
- EXEC\_HOST
- JOB\_NAME

You can also set format options of the email in the eadmi n script, located in the *LSF\_SERVERDIR* directory. Valid values are *fixed* or *full*.

## Default

N (Notification for job exception is standard and includes only job ID and either run time or idle factor.)

## FAIRSHARE\_ADJUSTMENT\_FACTOR

### Syntax

**FAIRSHARE\_ADJUSTMENT\_FACTOR**=*number*

### Description

Used only with fairshare scheduling. Fairshare adjustment plugin weighting factor.

In the calculation of a user's dynamic share priority, this factor determines the relative importance of the user-defined adjustment made in the fairshare plugin (libfairshareadjust.\*).

A positive float number both enables the fairshare plugin and acts as a weighting factor.

This parameter can also be set for an individual queue in lsb. queues. If defined, the queue value takes precedence.

## Default

0 (user-defined adjustment made in the fairshare plugin not used)

## GLOBAL\_EXIT\_RATE

### Syntax

**GLOBAL\_EXIT\_RATE**=*number*

### Description

Specifies a cluster-wide threshold for exited jobs. Specify a number of jobs. If EXIT\_RATE is not specified for the host in lsb. hosts, GLOBAL\_EXIT\_RATE defines a default exit rate for all hosts in the cluster. Host-level EXIT\_RATE overrides the GLOBAL\_EXIT\_RATE value.

If the number of jobs that exit over the period of time specified by JOB\_EXIT\_RATE\_DURATION (5 minutes by default) exceeds the number of jobs that you specify as the threshold in this parameter, LSF invokes LSF\_SERVERDIREADMIN to trigger a host exception.

### Example

**GLOBAL\_EXIT\_RATE=10** defines a job exit rate of 10 jobs for all hosts.

## Default

2147483647 (Unlimited threshold.)

## HIST\_HOURS

### Syntax

**HIST\_HOURS**=*hours*

## Description

Used only with fairshare scheduling. Determines a rate of decay for cumulative CPU time, run time, and historical run time.

To calculate dynamic user priority, LSF scales the actual CPU time and the run time using a decay factor, so that 1 hour of recently-used time is equivalent to 0.1 hours after the specified number of hours has elapsed.

To calculate dynamic user priority with decayed run time and historical run time, LSF scales the accumulated run time of finished jobs and run time of running jobs using the same decay factor, so that 1 hour of recently-used time is equivalent to 0.1 hours after the specified number of hours has elapsed.

When **HIST\_HOURS=0**, CPU time and run time accumulated by running jobs is not decayed.

This parameter can also be set for an individual queue in `l sb. queues`. If defined, the queue value takes precedence.

## Default

5

## JOB\_ACCEPT\_INTERVAL

### Syntax

**JOB\_ACCEPT\_INTERVAL**=*integer*

## Description

The number you specify is multiplied by the value of `l sb. params MBD_SLEEP_TIME` (60 seconds by default). The result of the calculation is the number of seconds to wait after dispatching a job to a host, before dispatching a second job to the same host.

If 0 (zero), a host may accept more than one job. By default, there is no limit to the total number of jobs that can run on a host, so if this parameter is set to 0, a very large number of jobs might be dispatched to a host all at once. This can overload your system to the point that it will be unable to create any more processes. It is not recommended to set this parameter to 0.

**JOB\_ACCEPT\_INTERVAL** set at the queue level (`l sb. queues`) overrides **JOB\_ACCEPT\_INTERVAL** set at the cluster level (`l sb. params`).

---

### Note:

The parameter **JOB\_ACCEPT\_INTERVAL** only applies when there are running jobs on a host. A host running a short job which finishes before **JOB\_ACCEPT\_INTERVAL** has elapsed is free to accept a new job without waiting.

---

## Default

1

## JOB\_ATTA\_DIR

### Syntax

**JOB\_ATTA\_DIR**=*directory*

## Description

The shared directory in which `mbatchd` saves the attached data of messages posted with the `bpost` command.

Use `JOB_ATTACHEDIR` if you use `bpost` and `bread` to transfer large data files between jobs and want to avoid using space in `LSB_SHAREDDIR`. By default, the `bread` command reads attachment data from the `JOB_ATTACHEDIR` directory.

`JOB_ATTACHEDIR` should be shared by all hosts in the cluster, so that any potential LSF master host can reach it. Like `LSB_SHAREDDIR`, the directory should be owned and writable by the primary LSF administrator. The directory must have at least 1 MB of free space.

The attached data will be stored under the directory in the format:

```
JOB_ATTACHEDIR/timestamp.jobid.msgs/msg$msgidnex
```

On UNIX, specify an absolute path. For example:

```
JOB_ATTACHEDIR=/opt/share/l sf_work
```

On Windows, specify a UNC path or a path with a drive letter. For example:

```
JOB_ATTACHEDIR=\\HostA\temp\l sf_work
```

or

```
JOB_ATTACHEDIR=D: \temp\l sf_work
```

After adding `JOB_ATTACHEDIR` to `lsb.params`, use `badminton reconfig` to reconfigure your cluster.

## Valid values

`JOB_ATTACHEDIR` can be any valid UNIX or Windows path up to a maximum length of 256 characters.

## Default

Not defined

If `JOB_ATTACHEDIR` is not specified, job message attachments are saved in `LSB_SHAREDINFO/`.

## JOB\_DEP\_LAST\_SUB

### Description

Used only with job dependency scheduling.

If set to 1, whenever dependency conditions use a job name that belongs to multiple jobs, LSF evaluates only the most recently submitted job.

Otherwise, all the jobs with the specified name must satisfy the dependency condition.

### Default

0

## JOB\_EXIT\_RATE\_DURATION

### Description

Defines how long LSF waits before checking the job exit rate for a host. Used in conjunction with `EXIT_RATE` in `lsb.hosts` for LSF host exception handling.

If the job exit rate is exceeded for the period specified by `JOB_EXIT_RATE_DURATION`, LSF invokes `LSF_SERVERDIR/eadmin` to trigger a host exception.

## Tuning

### Tip:

Tune `JOB_EXIT_RATE_DURATION` carefully. Shorter values may raise false alarms, longer values may not trigger exceptions frequently enough.

## Example

```
JOB_EXIT_RATE_DURATION=10
```

## Default

5 minutes

## JOB\_GROUP\_CLEAN

### Syntax

```
JOB_GROUP_CLEAN=Y | N
```

### Description

If **JOB\_GROUP\_CLEAN = Y**, implicitly created job groups that are empty and have no limits assigned to them are automatically deleted.

Job groups can only be deleted automatically if they have no limits specified (directly or in descendent job groups), have no explicitly created children job groups, and haven't been attached to an SLA.

### Default

N (Implicitly created job groups are not automatically deleted unless they are deleted manually with `bgdel`.)

## JOB\_INCLUDE\_POSTPROC

### Syntax

```
JOB_INCLUDE_POSTPROC=Y | N
```

### Description

Specifies whether LSF includes the post-execution processing of the job as part of the job. When set to Y:

- Prevents a new job from starting on a host until post-execution processing is finished on that host
- Includes the CPU and run times of post-execution processing with the job CPU and run times
- `sbatchd` sends both job finish status (DONE or EXIT) and post-execution processing status (POST\_DONE or POST\_ERR) to `mbatchd` at the same time

In MultiCluster job forwarding model, the `JOB_INCLUDE_POSTPROC` value in the receiving cluster applies to the job.

MultiCluster job lease model, the `JOB_INCLUDE_POSTPROC` value applies to jobs running on remote leased hosts as if they were running on local hosts.

The variable `LSB_JOB_INCLUDE_POSTPROC` in the user environment overrides the value of `JOB_INCLUDE_POSTPROC` in an application profile in `lsb.appl icat ions`.

`JOB_INCLUDE_POSTPROC` in an application profile in `lsb.appl icat ions` overrides the value of `JOB_INCLUDE_POSTPROC` in `lsb.params`.

For SGI cpusets, if `JOB_INCLUDE_POSTPROC=Y`, LSF does not release the cpuset until post-execution processing has finished, even though post-execution processes are not attached to the cpuset.

## Default

N (Post-execution processing is not included as part of the job, and a new job can start on the execution host before post-execution processing finishes.)

## JOB\_POSITION\_CONTROL\_BY\_ADMIN

### Syntax

`JOB_POSITION_CONTROL_BY_ADMIN=Y | N`

### Description

Allows LSF administrators to control whether users can use `bt op` and `bbot` to move jobs to the top and bottom of queues. When `JOB_POSITION_CONTROL_BY_ADMIN=Y`, only the LSF administrator (including any queue administrators) can use `bbot` and `bt op` to move jobs within a queue.

### Default

N

### See also

`bbot`, `bt op`

## JOB\_POSTPROC\_TIMEOUT

### Syntax

`JOB_POSTPROC_TIMEOUT=minutes`

### Description

Specifies a timeout in minutes for job post-execution processing. The specified timeout must be greater than zero.

If post-execution processing takes longer than the timeout, `sbat chd` reports that post-execution has failed (`POST_ERR` status), and kills the entire process group of the job's post-execution processes on UNIX and Linux. On Windows, only the parent process of the post-execution command is killed when the timeout expires. The child processes of the post-execution command are not killed.

If `JOB_INCLUDE_POSTPROC=Y`, and `sbat chd` kills the post-execution processes because the timeout has been reached, the CPU time of the post-execution processing is set to 0, and the job's CPU time does not include the CPU time of post-execution processing.

`JOB_POSTPROC_TIMEOUT` defined in an application profile in `lsb.appl icat ions` overrides the value in `lsb.params`. `JOB_POSTPROC_TIMEOUT` cannot be defined in user environment.

In MultiCluster job forwarding model, the `JOB_POSTPROC_TIMEOUT` value in the receiving cluster applies to the job.

MultiCluster job lease model, the JOB\_POSTPROC\_TIMEOUT value applies to jobs running on remote leased hosts as if they were running on local hosts.

## Default

2147483647 (Unlimited; post-execution processing does not time out.)

## JOB\_PRIORITY\_OVER\_TIME

### Syntax

**JOB\_PRIORITY\_OVER\_TIME**=*increment*/*interval*

### Description

JOB\_PRIORITY\_OVER\_TIME enables automatic job priority escalation when MAX\_USER\_PRIORITY is also defined.

### Valid values

*increment*

Specifies the value used to increase job priority every *interval* minutes. Valid values are positive integers.

*interval*

Specifies the frequency, in minutes, to *increment* job priority. Valid values are positive integers.

### Default

-1 (Not defined.)

### Example

JOB\_PRIORITY\_OVER\_TIME=3/20

Specifies that every 20 minute *interval* *increment* to job priority of pending jobs by 3.

### See also

MAX\_USER\_PRIORITY

## JOB\_RUNLIMIT\_RATIO

### Syntax

**JOB\_RUNLIMIT\_RATIO**=*integer* | 0

### Description

Specifies a ratio between a job run limit and the runtime estimate specified by bsub -We or bmod -We, -We+, -Wep. The ratio does not apply to the RUNTIME parameter in lsb. applications.

This ratio can be set to 0 and no restrictions are applied to the runtime estimate.

JOB\_RUNLIMIT\_RATIO prevents abuse of the runtime estimate. The value of this parameter is the ratio of run limit divided by the runtime estimate.

By default, the ratio value is 0. Only administrators can set or change this ratio. If the ratio changes, it only applies to newly submitted jobs. The changed value does not retroactively reapply to already submitted jobs.

If the ratio value is greater than 0:

- If the users specify a runtime estimate only (bsub -We), the job-level run limit will automatically be set to  $runtime\_ratio * runtime\_estimate$ . Jobs running longer than this run limit are killed by LSF. If the job-level run limit is greater than the hard run limit in the queue, the job is rejected.
- If the users specify a runtime estimate (-We) and job run limit (-W) at job submission, and the run limit is greater than  $runtime\_ratio * runtime\_estimate$ , the job is rejected.
- If the users modify the run limit to be greater than  $runtime\_ratio$ , they must increase the runtime estimate first (bmod -We). Then they can increase the default run limit.
- LSF remembers the run limit is set with bsub -W or convert from  $runtime\_ratio * runtime\_estimate$ . When users modify the run limit with bmod -Wn, the run limit is automatically be set to  $runtime\_ratio * runtime\_estimate$ . If the run limit is set from  $runtime\_ratio$ , LSF rejects the run limit modification.
- If users modify the runtime estimate with bmod -We and the run limit is set by the user, the run limit is  $\text{MIN}(new\_estimate * new\_ratio, run\_limit)$ . If the run limit is set by  $runtime\_ratio$ , the run limit is set to  $new\_estimate * new\_ratio$ .
- If users modify the runtime estimate by using bmod -Wen and the run limit is set by the user, it is not changed. If the run limit is set by  $runtime\_ratio$ , it is set to unlimited.

In MultiCluster job forwarding model, JOB\_RUNLIMIT\_RATIO values in both the sending and receiving clusters apply to the job. The run limit in the receiving cluster cannot be greater than the value of  $runtime * JOB\_RUNLIMIT\_RATIO$  in the receiving cluster. Some examples:

- Run limit (for example with bsub -We) is 10, **JOB\_RUNLIMIT\_RATIO=5** in the sending cluster, **JOB\_RUNLIMIT\_RATIO=0** in the receiving cluster—run limit=50, and the job will run
- Run limit (for example with bsub -We) is 10, **JOB\_RUNLIMIT\_RATIO=5** in the sending cluster, **JOB\_RUNLIMIT\_RATIO=3** in the receiving cluster—run limit=50, and the job will pend
- Run limit (for example with bsub -We) is 10, **JOB\_RUNLIMIT\_RATIO=5** in the sending cluster, **JOB\_RUNLIMIT\_RATIO=6** in the receiving cluster—run limit=50, and the job will run
- Run limit (for example with bsub -We) is 10, **JOB\_RUNLIMIT\_RATIO=0** in the sending cluster, **JOB\_RUNLIMIT\_RATIO=5** in the receiving cluster—run limit=50, and the job will run

MultiCluster job lease model, the JOB\_RUNLIMIT\_RATIO value applies to jobs running on remote leased hosts as if they were running on local hosts.

## Default

0

## JOB\_SCHEDULING\_INTERVAL

### Syntax

**JOB\_SCHEDULING\_INTERVAL=seconds | milliseconds ms**

### Description

Time interval at which mbat chd sends jobs for scheduling to the scheduling daemon mbschd along with any collected load information. Specify in seconds, or include the keyword ms to specify in milliseconds.

If set to 0, there is no interval between job scheduling sessions.

The smaller the value of this parameter, the quicker jobs are scheduled. However, when the master batch daemon spends more time doing job scheduling, it has less time to respond to user commands. To have a balance between speed of job scheduling and response to the LSF commands, start with a setting of 0 or 1, and increase if users see the message "Batch system not responding...".

## Valid Value

Number of seconds or milliseconds greater than or equal to zero (0).

## Default

5 seconds

## JOB\_SPOOL\_DIR

## Syntax

**JOB\_SPOOL\_DIR=dir**

## Description

Specifies the directory for buffering batch standard output and standard error for a job.

When JOB\_SPOOL\_DIR is defined, the standard output and standard error for the job is buffered in the specified directory.

Files are copied from the submission host to a temporary file in the directory specified by the JOB\_SPOOL\_DIR on the execution host. LSF removes these files when the job completes.

If JOB\_SPOOL\_DIR is not accessible or does not exist, files are spooled to the default job output directory \$HOME/.lsbatch.

For `bsub -is` and `bsub -Zs`, JOB\_SPOOL\_DIR must be readable and writable by the job submission user, and it must be shared by the master host and the submission host. If the specified directory is not accessible or does not exist, and JOB\_SPOOL\_DIR is specified, `bsub -is` cannot write to the default directory `LSB_SHAREDIR/cluster_name/l sf_i ndr`, and `bsub -Zs` cannot write to the default directory `LSB_SHAREDIR/cluster_name/l sf_cmddr`, and the job will fail.

As LSF runs jobs, it creates temporary directories and files under JOB\_SPOOL\_DIR. By default, LSF removes these directories and files after the job is finished. See `bsub` for information about job submission options that specify the disposition of these files.

On UNIX, specify an absolute path. For example:

```
JOB_SPOOL_DIR=/home/share/l sf_spool
```

On Windows, specify a UNC path or a path with a drive letter. For example:

```
JOB_SPOOL_DIR=\\HostA\share\spool di r
```

or

```
JOB_SPOOL_DIR=D: \share\spool di r
```

In a mixed UNIX/Windows cluster, specify one path for the UNIX platform and one for the Windows platform. Separate the two paths by a pipe character (|):

```
JOB_SPOOL_DIR=/usr/share/l sf_spool | \\HostA\share\spool di r
```

## Valid value

JOB\_SPOOL\_DIR can be any valid path.

The entire path including JOB\_SPOOL\_DIR can up to 4094 characters on UNIX and Linux or up to 255 characters for Windows. This maximum path length includes:

- All directory and file paths attached to the JOB\_SPOOL\_DIR path
- Temporary directories and files that the LSF system creates as jobs run.

The path you specify for JOB\_SPOOL\_DIR should be as short as possible to avoid exceeding this limit.

## Default

Not defined

Batch job output (standard output and standard error) is sent to the `.lsbatch` directory on the execution host:

- On UNIX: `$HOME/.lsbatch`
- On Windows: `%windir%\lsbatchmpuser_id\lsbatch`

If `%HOME%` is specified in the user environment, uses that directory instead of `%windir%` for spooled output.

## JOB\_TERMINATE\_INTERVAL

### Syntax

**JOB\_TERMINATE\_INTERVAL**=*seconds*

### Description

UNIX only.

Specifies the time interval in seconds between sending SIGINT, SIGTERM, and SIGKILL when terminating a job. When a job is terminated, the job is sent SIGINT, SIGTERM, and SIGKILL in sequence with a sleep time of JOB\_TERMINATE\_INTERVAL between sending the signals. This allows the job to clean up if necessary.

### Default

10 (seconds)

## LOCAL\_MAX\_PREEEXEC\_RETRY

### Syntax

**LOCAL\_MAX\_PREEEXEC\_RETRY**=*integer*

### Description

The maximum number of times to attempt the pre-execution command of a job on the local cluster.

### Valid values

`0 < LOCAL_MAX_PREEEXEC_RETRY < 2147483647`

### Default

2147483647 (Unlimited number of pre-execution retry times.)

## LSB\_SYNC\_HOST\_STAT\_LIM

### Syntax

**LSB\_SYNC\_HOST\_STAT\_LIM=y | Y**

### Description

Improves the speed with which `mbat chd` obtains host status, and therefore the speed with which LSF reschedules rerunnable jobs: the sooner LSF knows that a host has become unavailable, the sooner LSF reschedules any rerunnable jobs executing on that host. Useful for a large cluster.

When you define this parameter, `mbat chd` periodically obtains the host status from the master LIM, and then verifies the status by polling each `sbat chd` at an interval defined by the parameters `MBD_SLEEP_TIME` and `LSB_MAX_PROBE_SBD`.

### Default

N. `mbat chd` obtains and reports host status, without contacting the master LIM, by polling each `sbat chd` at an interval defined by the parameters `MBD_SLEEP_TIME` and `LSB_MAX_PROBE_SBD`.

### See also

`MBD_SLEEP_TIME`

`LSB_MAX_PROBE_SBD` in `lsf.conf`

## MAX\_ACCT\_ARCHIVE\_FILE

### Syntax

**MAX\_ACCT\_ARCHIVE\_FILE=*integer***

### Description

Enables automatic deletion of archived LSF accounting log files and specifies the archive limit.

### Compatibility

`ACCT_ARCHIVE_SIZE` or `ACCT_ARCHIVE_AGE` should also be defined.

### Example

```
MAX_ACCT_ARCHIVE_FILE=10
```

LSF maintains the current `lsb.acct` and up to 10 archives. Every time the old `lsb.acct.9` becomes `lsb.acct.10`, the old `lsb.acct.10` gets deleted.

### See also

- `ACCT_ARCHIVE_AGE` also enables automatic archiving
- `ACCT_ARCHIVE_SIZE` also enables automatic archiving
- `ACCT_ARCHIVE_TIME` also enables automatic archiving

### Default

-1 (Not defined. No deletion of `lsb.acct.n` files).

## MAX\_CONCURRENT\_JOB\_QUERY

### Syntax

**MAX\_CONCURRENT\_JOB\_QUERY=integer**

### Description

Defines how many concurrent job queries `mbat chd` can handle.

If a job information query is sent after the limit has been reached, an error message ("*Batch system concurrent query limit exceeded*") is displayed.

- If `mbat chd` is not using multithreading, the value of `MAX_CONCURRENT_JOB_QUERY` is always the maximum number of job queries in the cluster.
- If `mbat chd` is using multithreading (defined by the parameter `LSB_QUERY_PORT` in `lsf.conf`), the number of job queries in the cluster can temporarily become higher than the number specified by `MAX_CONCURRENT_JOB_QUERY`.

This increase in the total number of job queries is possible because the value of `MAX_CONCURRENT_JOB_QUERY` actually sets the maximum number of queries that can be handled by each child `mbat chd` that is forked by `mbat chd`. When the new child `mbat chd` starts, it handles new queries, but the old child `mbat chd` continues to run until all the old queries are finished. It is possible that the total number of job queries can be as high as `MAX_CONCURRENT_JOB_QUERY` multiplied by the number of child daemons forked by `mbat chd`.

### Valid values

1-100

### Default

2147483647 (Unlimited concurrent job queries.)

### See also

`LSB_QUERY_PORT` in `lsf.conf`

## MAX\_EVENT\_STREAM\_FILE\_NUMBER

### Syntax

**MAX\_EVENT\_STREAM\_FILE\_NUMBER=integer**

### Description

Determines the maximum number of different `lsb.stream.ut c` files that `mbat chd` uses. If the number of `lsb.stream.ut c` files reaches this number, `mbat chd` logs and error message to the `mbd.log` file and stops writing events to the `lsb.stream` file.

### Default

10

## MAX\_EVENT\_STREAM\_SIZE

### Syntax

**MAX\_EVENT\_STREAM\_SIZE**=*integer*

### Description

Determines the maximum size in MB of the `lsb.stream` file used by system performance analysis tools.

When the `MAX_EVENT_STREAM_SIZE` size is reached, LSF logs a special event `EVENT_END_OF_STREAM`, closes the stream and moves it to `lsb.stream.0` and a new stream is opened.

All applications that read the file once the event `EVENT_END_OF_STREAM` is logged should close the file and reopen it.

### Recommended value

2000 MB

### Default

1024 MB

## MAX\_INFO\_DIRS

### Syntax

**MAX\_INFO\_DIRS**=*num\_subdirs*

### Description

The number of subdirectories under the `LSB_SHAREDIR/cluster_name/logdir/info` directory.

When `MAX_INFO_DIRS` is enabled, `mbatchd` creates the specified number of subdirectories in the `info` directory. These subdirectories are given an integer as its name, starting with 0 for the first subdirectory. `mbatchd` writes the job files of all new submitted jobs into these subdirectories using the following formula to choose the subdirectory in which to store the job file:

```
subdirectory = jobID % MAX_INFO_DIRS
```

This formula ensures an even distribution of job files across the subdirectories.

---

#### Important:

If you are using local duplicate event logging, you must run `badadmin mbdrestart` after changing `MAX_INFO_DIRS` for the changes to take effect.

---

### Valid values

0-1024

### Default

0 (no subdirectories under the `info` directory; `mbatchd` writes all jobfiles to the `info` directory)

## Example

**MAX\_INFO\_DIRS=10**

mbat.chd creates ten subdirectories from `LSB_SHAREDIR/cluster_name/lsbinfo/0` to `LSB_SHAREDIR/cluster_name/lsbinfo/9`.

## MAX\_JOB\_ARRAY\_SIZE

### Syntax

**MAX\_JOB\_ARRAY\_SIZE=*integer***

### Description

Specifies the maximum number of jobs in a job array that can be created by a user for a single job submission. The maximum number of jobs in a job array cannot exceed this value.

A large job array allows a user to submit a large number of jobs to the system with a single job submission.

### Valid values

Specify a positive integer between 1 and 2147483646

### Default

1000

## MAX\_JOB\_ATTACH\_SIZE

### Syntax

**MAX\_JOB\_ATTACH\_SIZE=*integer* | 0**

Specify any number less than 20000.

### Description

Maximum attached data size, in KB, that can be transferred to a job.

Maximum size for data attached to a job with the `bpost` command. Useful if you use `bpost` and `bread` to transfer large data files between jobs and you want to limit the usage in the current working directory.

0 indicates that jobs cannot accept attached data files.

### Default

2147483647 (Unlimited; LSF does not set a maximum size of job attachments.)

## MAX\_JOB\_MSG\_NUM

### Syntax

**MAX\_JOB\_MSG\_NUM=*integer* | 0**

lsb.params

## Description

Maximum number of message slots for each job. Maximum number of messages that can be posted to a job with the `bpost` command.

0 indicates that jobs cannot accept external messages.

## Default

128

## MAX\_JOB\_NUM

## Syntax

**MAX\_JOB\_NUM**=*integer*

## Description

The maximum number of finished jobs whose events are to be stored in the `lsb.events` log file.

Once the limit is reached, `mbatchd` starts a new event log file. The old event log file is saved as `lsb.events.n`, with subsequent sequence number suffixes incremented by 1 each time a new log file is started. Event logging continues in the new `lsb.events` file.

## Default

1000

## MAX\_JOB\_PREEMPT

## Syntax

**MAX\_JOB\_PREEMPT**=*integer*

## Description

The maximum number of times a job can be preempted. Applies to queue-based preemption only.

## Valid values

$0 < \text{MAX\_JOB\_PREEMPT} < 2147483647$

## Default

2147483647 (Unlimited number of preemption times.)

## MAX\_JOB\_REQUEUE

## Syntax

**MAX\_JOB\_REQUEUE**=*integer*

## Description

The maximum number of times to requeue a job automatically.

## Valid values

$0 < \text{MAX\_JOB\_REQUEUE} < 2147483647$

## Default

2147483647 (Unlimited number of requeue times.)

## MAX\_JOBID

### Syntax

**MAX\_JOBID**=*integer*

### Description

The job ID limit. The job ID limit is the highest job ID that LSF will ever assign, and also the maximum number of jobs in the system.

By default, LSF assigns job IDs up to 6 digits. This means that no more than 999999 jobs can be in the system at once.

Specify any integer from 999999 to 2147483646 (for practical purposes, you can use any 10-digit integer less than this value).

You cannot lower the job ID limit, but you can raise it to 10 digits. This allows longer term job accounting and analysis, and means you can have more jobs in the system, and the job ID numbers will roll over less often.

LSF assigns job IDs in sequence. When the job ID limit is reached, the count rolls over, so the next job submitted gets job ID "1". If the original job 1 remains in the system, LSF skips that number and assigns job ID "2", or the next available job ID. If you have so many jobs in the system that the low job IDs are still in use when the maximum job ID is assigned, jobs with sequential numbers could have totally different submission times.

### Example

MAX\_JOBID=125000000

### Default

999999

## MAX\_JOBINFO\_QUERY\_PERIOD

### Syntax

**MAX\_JOBINFO\_QUERY\_PERIOD**=*integer*

### Description

Maximum time for job information query commands (for example, with `bj obs`) to wait.

When the time arrives, the query command processes exit, and all associated threads are terminated.

If the parameter is not defined, query command processes will wait for all threads to finish.

Specify a multiple of `MBD_REFRESH_TIME`.

lsb.params

## Valid values

Any positive integer greater than or equal to one (1)

## Default

2147483647 (Unlimited wait time.)

## See also

LSB\_BLOCK\_JOBINFO\_TIMEOUT in `lsf.conf`

# MAX\_PEND\_JOBS

## Syntax

**MAX\_PEND\_JOBS**=*integer*

## Description

The maximum number of pending jobs in the system.

This is the hard system-wide pending job threshold. No user or user group can exceed this limit unless the job is forwarded from a remote cluster.

If the user or user group submitting the job has reached the pending job threshold as specified by `MAX_PEND_JOBS`, LSF will reject any further job submission requests sent by that user or user group. The system will continue to send the job submission requests with the interval specified by `SUB_TRY_INTERVAL` in `lsb.params` until it has made a number of attempts equal to the `LSB_NTRIES` environment variable. If `LSB_NTRIES` is not defined and LSF rejects the job submission request, the system will continue to send the job submission requests indefinitely as the default behavior.

## Default

2147483647 (Unlimited number of pending jobs.)

## See also

`SUB_TRY_INTERVAL`

# MAX\_PREEEXEC\_RETRY

## Syntax

**MAX\_PREEEXEC\_RETRY**=*integer*

## Description

MultiCluster job forwarding model only. The maximum number of times to attempt the pre-execution command of a job from a remote cluster.

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

## Valid values

$0 < \text{MAX\_PREEEXEC\_RETRY} < 2147483647$

## Default

5

## MAX\_SBD\_CONNS

### Syntax

**MAX\_SBD\_CONNS**=*integer*

### Description

The maximum number of file descriptors `mbat chd` can have open and connected concurrently to `sbat chd`.

Controls the maximum number of connections that LSF can maintain to `sbat chds` in the system.

Do not exceed the file descriptor limit of the root process (the usual limit is 1024). Setting it equal or larger than this limit can cause `mbat chd` to constantly die because `mbat chd` allocates all file descriptors to `sbat chd` connection. This could cause `mbat chd` to run out of descriptors, which results in an `mbat chd` fatal error, such as failure to open `lsb.events`.

Use together with `LSB_MAX_JOB_DISPATCH_PER_SESSION` in `lsf.conf`.

### Example

A reasonable setting is:

**MAX\_SBD\_CONNS=768**

For a large cluster, specify a value equal to the number of hosts in your cluster plus a buffer. For example, if your cluster includes 4000 hosts: **MAX\_SBD\_CONNS=4100**

---

#### Important:

Set `LSB_MAX_JOB_DISPATCH_PER_SESSION` in `lsf.conf` equal to one-half the value of `MAX_SBD_CONNS`.

---

## Default

64

## MAX\_SBD\_FAIL

### Syntax

**MAX\_SBD\_FAIL**=*integer*

### Description

The maximum number of retries for reaching a non-responding slave batch daemon, `sbat chd`.

The interval between retries is defined by `MBD_SLEEP_TIME`. If `mbat chd` fails to reach a host and has retried `MAX_SBD_FAIL` times, the host is considered unreachable.

If you define `LSB_SYNC_HOST_STAT_LIM=Y`, `mbat chd` obtains the host status from the master LIM before it polls `sbat chd`. When the master LIM reports that a host is unavailable (LIM is down) or unreachable (`sbat chd` is down) `MAX_SBD_FAIL` number of times, `mbat chd` reports the host status as unavailable or unreachable.

When a host becomes unavailable, `mbatchd` assumes that all jobs running on that host have exited and that all rerunnable jobs (jobs submitted with the `bsub -r` option) are scheduled to be rerun on another host.

## Default

3

# MAX\_TOTAL\_TIME\_PREEMPT

## Syntax

**MAX\_TOTAL\_TIME\_PREEMPT**=*integer*

## Description

The accumulated preemption time in minutes after which a job cannot be preempted again, where *minutes* is wall-clock time, not normalized time.

The parameter of the same name in `lsb.queues` overrides this parameter. The parameter of the same name in `lsb.applications` overrides both this parameter and the parameter of the same name in `lsb.queues`.

## Valid values

Any positive integer greater than or equal to one (1)

## Default

Unlimited

# MAX\_USER\_PRIORITY

## Syntax

**MAX\_USER\_PRIORITY**=*integer*

## Description

Enables user-assigned job priority and specifies the maximum job priority a user can assign to a job.

LSF and queue administrators can assign a job priority higher than the specified value for jobs they own.

## Compatibility

User-assigned job priority changes the behavior of `bt op` and `bbot`.

## Example

```
MAX_USER_PRIORITY=100
```

Specifies that 100 is the maximum job priority that can be specified by a user.

## Default

-1 (Not defined.)

## See also

- bsub, bmod, bt op, bbot
- JOB\_PRIORITY\_OVER\_TIME

## MBD\_EGO\_CONNECT\_TIMEOUT

### Syntax

**MBD\_EGO\_CONNECT\_TIMEOUT**=*seconds*

### Description

For EGO-enabled SLA scheduling, timeout parameter for network I/O connection with EGO vemkd.

### Default

0 seconds

## MBD\_EGO\_READ\_TIMEOUT

### Syntax

**MBD\_EGO\_READ\_TIMEOUT**=*seconds*

### Description

For EGO-enabled SLA scheduling, timeout parameter for network I/O read from EGO vemkd after connection with EGO.

### Default

0 seconds

## MBD\_EGO\_TIME2LIVE

### Syntax

**MBD\_EGO\_TIME2LIVE**=*minutes*

### Description

For EGO-enabled SLA scheduling, specifies how long EGO should keep information about host allocations in case mbatchd restarts,

### Default

0 minutes

## MBD\_QUERY\_CPUS

### Syntax

**MBD\_QUERY\_CPUS**=*cpu\_list*

*cpu\_list* defines the list of master host CPUS on which the mbatchd child query processes can run. Format the list as a white-space delimited list of CPU numbers.

For example, if you specify

```
MBD_QUERY_CPUS=1 2 3
```

the `mbat chd` child query processes will run only on CPU numbers 1, 2, and 3 on the master host.

## Description

This parameter allows you to specify the master host CPUs on which `mbat chd` child query processes can run (hard CPU affinity). This improves `mbat chd` scheduling and dispatch performance by binding query processes to specific CPUs so that higher priority `mbat chd` processes can run more efficiently.

When you define this parameter, LSF runs `mbat chd` child query processes *only* on the specified CPUs. The operating system can assign other processes to run on the same CPU; however, if utilization of the bound CPU is lower than utilization of the unbound CPUs.

## Important

1. You can specify CPU affinity only for master hosts that use one of the following operating systems:
  - Linux 2.6 or higher
  - Solaris 8 or higher
2. If failover to a master host candidate occurs, LSF maintains the hard CPU affinity, provided that the master host candidate has the same CPU configuration as the original master host. If the configuration differs, LSF ignores the CPU list and reverts to default behavior.

## Related parameters

To improve scheduling and dispatch performance of all LSF daemons, you should use `MBD_QUERY_CPUS` together with `EGO_DAEMONS_CPUS` (in `ego.conf`), which controls LIM CPU allocation, and `LSF_DAEMONS_CPUS`, which binds `mbat chd` and `mbschd` daemon processes to specific CPUs so that higher priority daemon processes can run more efficiently. To get best performance, CPU allocation for all four daemons should be assigned their own CPUs. For example, on a 4 CPU SMP host, the following configuration will give the best performance:

```
EGO_DAEMONS_CPUS=0 LSF_DAEMONS_CPUS=1: 2 MBD_QUERY_CPUS=3
```

## Default

Not defined

## See also

`LSF_DAEMONS_CPUS` in `lsf.conf`

## MBD\_REFRESH\_TIME

### Syntax

```
MBD_REFRESH_TIME=seconds [min_refresh_time]
```

where *min\_refresh\_time* defines the minimum time (in seconds) that the child `mbat chd` will stay to handle queries.

## Description

Time interval, in seconds, when `mbat chd` will fork a new child `mbat chd` to service query requests to keep information sent back to clients updated. A child `mbat chd` processes query requests creating threads.

`MBD_REFRESH_TIME` applies only to UNIX platforms that support thread programming.

To enable `MBD_REFRESH_TIME` you must specify `LSB_QUERY_PORT` in `lsf.conf`. The child `mbat chd` listens to the port number specified by `LSB_QUERY_PORT` and creates threads to service requests until the job changes status, a new job is submitted, or `MBD_REFRESH_TIME` has expired.

- If `MBD_REFRESH_TIME` is  $< \textit{min\_refresh\_time}$ , the child `mbat chd` exits at `MBD_REFRESH_TIME` even if the job changes status or a new job is submitted before `MBD_REFRESH_TIME` expires.
- If `MBD_REFRESH_TIME`  $> \textit{min\_refresh\_time}$ :
  - the child `mbat chd` exits at *min\_refresh\_time* if a job changes status or a new job is submitted before the *min\_refresh\_time*
  - the child `mbat chd` exits after the *min\_refresh\_time* when a job changes status or a new job is submitted
- If `MBD_REFRESH_TIME`  $> \textit{min\_refresh\_time}$  and no job changes status or a new job is submitted, the child `mbat chd` exits at `MBD_REFRESH_TIME`

The value of this parameter must be between 0 and 300. Any values specified out of this range are ignored, and the system default value is applied.

The `bj obs` command may not display up-to-date information if two consecutive query commands are issued before a child `mbat chd` expires because child `mbat chd` job information is not updated. If you use the `bj obs` command and do not get up-to-date information, you may need to decrease the value of this parameter. Note, however, that the lower the value of this parameter, the more you negatively affect performance.

The number of concurrent requests is limited by the number of concurrent threads that a process can have. This number varies by platform:

- Sun Solaris, 2500 threads per process
- AIX, 512 threads per process
- Digital, 256 threads per process
- HP-UX, 64 threads per process

## Valid Values

5-300 seconds

## Default

5 seconds

*min\_refresh\_time* default is 10 seconds

## See also

`LSB_QUERY_PORT` in `lsf.conf`

## MBD\_SLEEP\_TIME

### Syntax

**MBD\_SLEEP\_TIME**=*seconds*

### Description

Used in conjunction with the parameters `SLOT_RESERVE`, `MAX_SBD_FAIL`, and `JOB_ACCEPT_INTERVAL`

lsb.params

Amount of time in seconds used for calculating parameter values.

## Default

If not defined, 60 seconds. MBD\_SLEEP\_TIME is set at installation to 20 seconds.

## MBD\_USE\_EGO\_MXJ

### Syntax

**MBD\_USE\_EGO\_MXJ=Y | N**

### Description

By default, when EGO-enabled SLA scheduling is configured, EGO allocates an entire host to LSF, which uses its own MXJ definition to determine how many slots are available on the host. LSF gets its host allocation from EGO, and runs as many jobs as the LSF configured MXJ for that host dictates.

MBD\_USE\_EGO\_MXJ forces LSF to use the job slot maximum configured in the EGO consumer. This allows partial sharing of hosts (for example, a large SMP computer) among different consumers or workload managers. When MBD\_USE\_EGO\_MXJ is set, LSF schedules jobs based on the number of slots allocated from EGO. For example, if host A has 4 processors, but EGO allocates 2 slots to an EGO-enabled SLA consumer. LSF can schedule a maximum of 2 jobs from that SLA on host A.

## Default

N (mbatcthd uses the LSF MXJ)

## MC\_PENDING\_REASON\_PKG\_SIZE

### Syntax

**MC\_PENDING\_REASON\_PKG\_SIZE=*kilobytes* | 0**

### Description

MultiCluster job forwarding model only. Pending reason update package size, in KB. Defines the maximum amount of pending reason data this cluster will send to submission clusters in one cycle.

Specify the keyword 0 (zero) to disable the limit and allow any amount of data in one package.

## Default

512

## MC\_PENDING\_REASON\_UPDATE\_INTERVAL

### Syntax

**MC\_PENDING\_REASON\_UPDATE\_INTERVAL=*seconds* | 0**

### Description

MultiCluster job forwarding model only. Pending reason update interval, in seconds. Defines how often this cluster will update submission clusters about the status of pending MultiCluster jobs.

Specify the keyword 0 (zero) to disable pending reason updating between clusters.

## Default

300

## MC\_PLUGIN\_SCHEDULE\_ENHANCE

### Syntax

**MC\_PLUGIN\_SCHEDULE\_ENHANCE= RESOURCE\_ONLY**

**MC\_PLUGIN\_SCHEDULE\_ENHANCE= COUNT\_PREEMPTABLE [HIGH\_QUEUE\_PRIORITY]  
[PREEMPTABLE\_QUEUE\_PRIORITY] [PENDING\_WHEN\_NOSLOTS]**

---

#### Note:

When any one of HIGH\_QUEUE\_PRIORITY, PREEMPTABLE\_QUEUE\_PRIORITY or PENDING\_WHEN\_NOSLOTS is defined, COUNT\_PREEMPTABLE is enabled automatically.

---

### Description

MultiCluster job forwarding model only. The parameter MC\_PLUGIN\_SCHEDULE\_ENHANCE enhances the scheduler for the MultiCluster job forwarding model based on the settings selected. Use in conjunction with MC\_PLUGIN\_UPDATE\_INTERVAL to set the data update interval between remote clusters. MC\_PLUGIN\_UPDATE\_INTERVAL must be a non-zero value to enable the MultiCluster enhanced scheduler.

With the parameter MC\_PLUGIN\_SCHEDULE\_ENHANCE set to a valid value, remote resources are considered as if **MC\_PLUGIN\_REMOTE\_RESOURCE=Y** regardless of the actual setting. In addition the submission cluster scheduler considers specific execution queue resources when scheduling jobs. See *Using Platform MultiCluster* for details.

---

#### Note:

The parameter MC\_PLUGIN\_SCHEDULE\_ENHANCE was introduced in LSF Version 7 Update 6. All clusters within a MultiCluster configuration must be running a version of LSF containing this parameter to enable the enhanced scheduler.

After a MultiCluster connection is established, counters take the time set in MC\_PLUGIN\_UPDATE\_INTERVAL to update. Scheduling decisions made before this first interval has passed do not accurately account for remote queue workload.

---

### Default

Not defined.

The enhanced scheduler is not used. If **MC\_PLUGIN\_REMOTE\_RESOURCE=Y** in `lsf.conf` remote resource availability is considered before jobs are forwarded to the queue with the most available slots.

### See also

MC\_PLUGIN\_UPDATE\_INTERVAL in `lsb.params`.

MC\_PLUGIN\_REMOTE\_RESOURCE in `lsf.conf`.

## MC\_PLUGIN\_UPDATE\_INTERVAL

### Syntax

**MC\_PLUGIN\_UPDATE\_INTERVAL**=*seconds* | 0

### Description

MultiCluster job forwarding model only; set for the execution cluster. The number of seconds between data updates between clusters.

A non-zero value enables collection of remote cluster queue data for use by the submission cluster enhanced scheduler.

Suggested value when enabled is MBD\_SLEEP\_TIME (default is 20 seconds).

A value of 0 disables collection of remote cluster queue data.

### Default

0

### See Also

MC\_PLUGIN\_SCHEDULE\_ENHANCE in lsf.params.

## MC\_RECLAIM\_DELAY

### Syntax

**MC\_RECLAIM\_DELAY**=*minutes*

### Description

MultiCluster resource leasing model only. The reclaim interval (how often to reconfigure shared leases) in minutes.

Shared leases are defined by Type=shared in the lsb.resources HostExport section.

### Default

10 (minutes)

## MC\_RUSAGE\_UPDATE\_INTERVAL

### Syntax

**MC\_RUSAGE\_UPDATE\_INTERVAL**=*seconds*

### Description

MultiCluster only. Enables resource use updating for MultiCluster jobs running on hosts in the cluster and specifies how often to send updated information to the submission or consumer cluster.

### Default

300

## MIN\_SWITCH\_PERIOD

### Syntax

**MIN\_SWITCH\_PERIOD**=*seconds*

### Description

The minimum period in seconds between event log switches.

Works together with `MAX_JOB_NUM` to control how frequently `mbat chd` switches the file. `mbat chd` checks if `MAX_JOB_NUM` has been reached every `MIN_SWITCH_PERIOD` seconds. If `mbat chd` finds that `MAX_JOB_NUM` has been reached, it switches the events file.

To significantly improve the performance of `mbat chd` for large clusters, set this parameter to a value equal to or greater than 600. This causes `mbat chd` to fork a child process that handles event switching, thereby reducing the load on `mbat chd`. `mbat chd` terminates the child process and appends delta events to new events after the `MIN_SWITCH_PERIOD` has elapsed.

### Default

0

No minimum period. Log switch frequency is not restricted.

### See also

`MAX_JOB_NUM`

## NEWJOB\_REFRESH

### Syntax

**NEWJOB\_REFRESH**=Y | N

### Description

Enables a child `mbat chd` to get up to date information about new jobs from the parent `mbat chd`. When set to Y, job queries with `bj obs` display new jobs submitted after the child `mbat chd` was created.

If you have enabled multithreaded `mbatchd` support, the `bj obs` command may not display up-to-date information if two consecutive query commands are issued before a child `mbat chd` expires because child `mbat chd` job information is not updated. Use **NEWJOB\_REFRESH=Y** to enable the parent `mbat chd` to push new job information to a child `mbat chd`.

When **NEWJOB\_REFRESH=Y**, as users submit new jobs, the parent `mbat chd` pushes the new job event to the child `mbat chd`. The parent `mbat chd` transfers the following kinds of new jobs to the child `mbat chd`:

- Newly submitted jobs
- Restarted jobs
- Remote lease model jobs from the submission cluster
- Remote forwarded jobs from the submission cluster

When **NEWJOB\_REFRESH=Y**, you should set `MBD_REFRESH_TIME` to a value greater than 10 seconds.

## Required parameters

LSB\_QUERY\_PORT must be enabled in lsf.conf.

## Restrictions

The parent mbat chd only pushes the new job event to a child mbat chd. The child mbat chd is not aware of status changes of existing jobs. The child mbat chd will not reflect the results of job control commands (bmod, bmi g, bswi t ch, btop, bbot, brequeue, bst op, bresume, and so on) invoked after the child mbat chd is created.

## Default

N (Not defined. New jobs are not pushed to the child mbat chd.)

## See also

MBD\_REFRESH\_TIME

## NO\_PREEMPT\_FINISH\_TIME

### Syntax

**NO\_PREEMPT\_FINISH\_TIME**=*minutes* | *percentage*

### Description

Prevents preemption of jobs that will finish within the specified number of minutes or the specified percentage of the estimated run time or run limit.

Specifies that jobs due to finish within the specified number of minutes or percentage of job duration should not be preempted, where *minutes* is wall-clock time, not normalized time. Percentage must be greater than 0 or less than 100% (between 1% and 99%).

For example, if the job run limit is 60 minutes and **NO\_PREEMPT\_FINISH\_TIME=10%**, the job cannot be preempted after it running 54 minutes or longer.

If you specify percentage for **NO\_PREEMPT\_FINISH\_TIME**, requires a run time (bsub - We or RUNTIME in lsb. appl i cat i ons), or run limit to be specified for the job (bsub - W, or RUNLIMIT in lsb. queues, or RUNLIMIT in lsb. appl i cat i ons)

### Default

-1 (Not defined.)

## NO\_PREEMPT\_INTERVAL

### Syntax

**NO\_PREEMPT\_INTERVAL**=*minutes*

### Description

Prevents preemption of jobs for the specified number of minutes of uninterrupted run time, where *minutes* is wall-clock time, not normalized time. **NO\_PREEMPT\_INTERVAL=0** allows immediate preemption of jobs as soon as they start or resume running.

The parameter of the same name in `lsb.queues` overrides this parameter. The parameter of the same name in `lsb.applications` overrides both this parameter and the parameter of the same name in `lsb.queues`.

## Default

0

## NO\_PREEMPT\_RUN\_TIME

### Syntax

**NO\_PREEMPT\_RUN\_TIME**=*minutes* | *percentage*

### Description

Prevents preemption of jobs that have been running for the specified number of minutes or the specified percentage of the estimated run time or run limit.

Specifies that jobs that have been running for the specified number of minutes or longer should not be preempted, where *minutes* is wall-clock time, not normalized time. Percentage must be greater than 0 or less than 100% (between 1% and 99%).

For example, if the job run limit is 60 minutes and **NO\_PREEMPT\_RUN\_TIME=50%**, the job cannot be preempted after it running 30 minutes or longer.

If you specify percentage for **NO\_PREEMPT\_RUN\_TIME**, requires a run time (`bsub -We` or `RUNTIME` in `lsb.applications`), or run limit to be specified for the job (`bsub -W`, or `RUNLIMIT` in `lsb.queues`, or `RUNLIMIT` in `lsb.applications`)

## Default

-1 (Not defined.)

## NQS\_QUEUES\_FLAGS

### Syntax

**NQS\_QUEUES\_FLAGS**=*integer*

### Description

For Cray NQS compatibility only. Used by LSF to get the NQS queue information.

If the NQS version on a Cray is NQS 1.1, 80.42 or NQS 71.3, this parameter does not need to be defined.

For other versions of NQS on Cray, define both **NQS\_QUEUES\_FLAGS** and **NQS\_REQUESTS\_FLAGS**.

To determine the value of this parameter, run the `NQS qstat` command. The value of `Npk_int[1]` in the output is the value you need for this parameter. Refer to the NQS chapter in *Administering Platform LSF* for more details.

## Default

2147483647 (Not defined.)

## NQS\_REQUESTS\_FLAGS

### Syntax

**NQS\_REQUESTS\_FLAGS**=*integer*

### Description

For Cray NQS compatibility only.

If the NQS version on a Cray is NQS 80.42 or NQS 71.3, this parameter does not need to be defined.

If the version is NQS 1.1 on a Cray, set this parameter to 251918848. This is the `qstat` flag that LSF uses to retrieve requests on Cray in long format.

For other versions of NQS on a Cray, run the `NQS qstat` command. The value of `Npk_int[1]` in the output is the value you need for this parameter. Refer to the NQS chapter in *Administering Platform LSF* for more details.

### Default

2147483647 (Not defined.)

## PARALLEL\_SCHED\_BY\_SLOT

### Syntax

**PARALLEL\_SCHED\_BY\_SLOT**=*y* | **Y**

### Description

If defined, LSF schedules jobs based on the number of slots assigned to the hosts instead of the number of CPUs. These slots can be defined by `host` in `lsb.hosts` or by `slot limit` in `lsb.resources`.

All slot-related messages still show the word “processors”, but actually refer to “slots” instead. Similarly, all scheduling activities also use slots instead of processors.

### Default

N (Disabled.)

### See also

- JL/U and MXJ in `lsb.hosts`
- SLOTS and SLOTS\_PER\_PROCESSOR in `lsb.resources`

## PEND\_REASON\_MAX\_JOBS

### Syntax

**PEND\_REASON\_MAX\_JOBS**=*integer*

### Description

Number of jobs for each user per queue for which pending reasons are calculated by the scheduling daemon `mbschd`. Pending reasons are calculated at a time period set by `PEND_REASON_UPDATE_INTERVAL`.

## Default

20 jobs

## PEND\_REASON\_UPDATE\_INTERVAL

### Syntax

**PEND\_REASON\_UPDATE\_INTERVAL=seconds**

### Description

Time interval that defines how often pending reasons are calculated by the scheduling daemon mbschd.

## Default

30 seconds

## PG\_SUSP\_IT

### Syntax

**PG\_SUSP\_IT=seconds**

### Description

The time interval that a host should be interactively idle (it > 0) before jobs suspended because of a threshold on the pg load index can be resumed.

This parameter is used to prevent the case in which a batch job is suspended and resumed too often as it raises the paging rate while running and lowers it while suspended. If you are not concerned with the interference with interactive jobs caused by paging, the value of this parameter may be set to 0.

## Default

180 seconds

## PREEXEC\_EXCLUDE\_HOST\_EXIT\_VALUES

### Syntax

**PREEXEC\_EXCLUDE\_HOST\_EXIT\_VALUES=all [~exit\_value] | exit\_value [exit\_value] [...]**

### Description

Specify one or more values (between 1 and 255, but not 99) that corresponds to the exit code your pre-execution scripts exits with in the case of failure. LSF excludes any hosts that attempt to run the pre-exec script and exit with the value specified in PREEXEC\_EXCLUDE\_HOST\_EXIT\_VALUES.

The exclusion list exists for this job until the mbat chd restarts.

Specify more than one value by separating them with a space. 99 is a reserved value. For example, **PREEXEC\_EXCLUDE\_HOST\_EXIT\_VALUES=1 14 19 20 21**.

Exclude values using a "~": **PREEXEC\_EXCLUDE\_HOST\_EXIT\_VALUES=all ~40**

In the case of failures that could be avoided by retrying on the same host, add the retry process to the pre-exec script.

Use in combination with `MAX_PREEEXEC_RETRY` in `lsb.params` to limit the total number of hosts that are tried. In a multicluster environment, use in combination with `LOCAL_MAX_PREEEXEC_RETRY` and `REMOTE_MAX_PREEEXEC_RETRY`.

## Default

None.

# PREEMPT\_FOR

## Syntax

```
PREEMPT_FOR=[GROUP_JLP] [GROUP_MAX] [HOST_JLU] [LEAST_RUN_TIME] [MINI_JOB]
[USER_JLP] [OPTIMAL_MINI_JOB]
```

## Description

If preemptive scheduling is enabled, this parameter is used to disregard suspended jobs when determining if a job slot limit is exceeded, to preempt jobs with the shortest running time, and to optimize preemption of parallel jobs.

If preemptive scheduling is enabled, more lower-priority parallel jobs may be preempted than necessary to start a high-priority parallel job. Both running and suspended jobs are counted when calculating the number of job slots in use, except for the following limits:

- The total job slot limit for hosts, specified at the host level
- Total job slot limit for individual users, specified at the user level—by default, suspended jobs still count against the limit for user groups

Specify one or more of the following keywords. Use spaces to separate multiple keywords.

### GROUP\_JLP

Counts only running jobs when evaluating if a user group is approaching its per-processor job slot limit (`SLOTS_PER_PROCESSOR`, `USERS`, and `PER_HOST=all` in the `lsb.resources` file). Suspended jobs are ignored when this keyword is used.

### GROUP\_MAX

Counts only running jobs when evaluating if a user group is approaching its total job slot limit (`SLOTS`, `PER_USER=all`, and `HOSTS` in the `lsb.resources` file). Suspended jobs are ignored when this keyword is used. When preemptive scheduling is enabled, suspended jobs never count against the total job slot limit for individual users.

### HOST\_JLU

Counts only running jobs when evaluating if a user or user group is approaching its per-host job slot limit (`SLOTS` and `USERS` in the `lsb.resources` file). Suspended jobs are ignored when this keyword is used.

### LEAST\_RUN\_TIME

Preempts the job that has been running for the shortest time. Run time is wall-clock time, not normalized run time.

### MINI\_JOB

Optimizes the preemption of parallel jobs by preempting only enough parallel jobs to start the high-priority parallel job.

### **OPTIMAL\_MINI\_JOB**

Optimizes preemption of parallel jobs by preempting only low-priority parallel jobs based on the least number of jobs that will be suspended to allow the high-priority parallel job to start.

User limits and user group limits can interfere with preemption optimization of OPTIMAL\_MINI\_JOB. You should not configure OPTIMAL\_MINI\_JOB if you have user or user group limits configured.

You should configure **PARALLEL\_SCHED\_BY\_SLOT=Y** when using OPTIMAL\_MINI\_JOB.

### **USER\_JLP**

Counts only running jobs when evaluating if a user is approaching their per-processor job slot limit (SLOTS\_PER\_PROCESSOR, USERS, and PER\_HOST=all in the lsb.resources file). Suspended jobs are ignored when this keyword is used. Ignores suspended jobs when calculating the user-processor job slot limit for individual users. When preemptive scheduling is enabled, suspended jobs never count against the total job slot limit for individual users.

## Default

0 (The parameter is not defined.)

Both running and suspended jobs are included in job slot limit calculations, except for job slots limits for hosts and individual users where only running jobs are ever included.

## PREEMPT\_JOBTYPE

### Syntax

**PREEMPT\_JOBTYPE=[EXCLUSIVE] [BACKFILL]**

### Description

If preemptive scheduling is enabled, this parameter enables preemption of exclusive and backfill jobs.

Specify one or both of the following keywords. Separate keywords with a space.

#### **EXCLUSIVE**

Enables preemption of and preemption by exclusive jobs.

LSB\_DISABLE\_LIMLOCK\_EXCL=Y in lsf.conf must also be defined.

#### **BACKFILL**

Enables preemption of backfill jobs. Jobs from higher priority queues can preempt jobs from backfill queues that are either backfilling reserved job slots or running as normal jobs.

lsb.params

## Default

Not defined. Exclusive and backfill jobs are only preempted if the exclusive low priority job is running on a different host than the one used by the preemptive high priority job.

## PREEMPTABLE\_RESOURCES

### Syntax

**PREEMPTABLE\_RESOURCES**=*resource\_name1* [*resource\_name2*] [*resource\_name3*] ...

### Description

Enables license preemption when preemptive scheduling is enabled (has no effect if **PREEMPTIVE** is not also specified) and specifies the licenses that will be preemption resources. Specify shared numeric resources, static or decreasing, that LSF is configured to release (**RELEASE=Y** in `lsf.shared`, which is the default).

You must also configure LSF preemption actions to make the preempted application releases its licenses. To kill preempted jobs instead of suspending them, set **TERMINATE\_WHEN=PREEMPT** in `lsb.queues`, or set **JOB\_CONTROLS** in `lsb.queues` and specify `brequeue` as the **SUSPEND** action.

### Default

Not defined (if preemptive scheduling is configured, LSF preempts on job slots only)

## PREEMPTION\_WAIT\_TIME

### Syntax

**PREEMPTION\_WAIT\_TIME**=*seconds*

### Description

You must also specify **PREEMPTABLE\_RESOURCES** in `lsb.params`.

The amount of time LSF waits, after preempting jobs, for preemption resources to become available. Specify at least 300 seconds.

If LSF does not get the resources after this time, LSF might preempt more jobs.

### Default

300 (seconds)

## PRIVILEGED\_USER\_FORCE\_BKILL

### Syntax

**PRIVILEGED\_USER\_FORCE\_BKILL**=*y* | **Y**

### Description

If **Y**, only `root` or the LSF administrator can successfully run `bkill -r`. For any other users, `-r` is ignored. If not defined, any user can run `bkill -r`.

## Default

Not defined.

## REMOTE\_MAX\_PREEEXEC\_RETRY

### Syntax

**REMOTE\_MAX\_PREEEXEC\_RETRY**=*integer*

### Description

The maximum number of times to attempt the pre-execution command of a job from the remote cluster.

### Valid values

0 < REMOTE\_MAX\_PREEEXEC\_RETRY < 2147483647

## Default

5

## RESOURCE\_RESERVE\_PER\_SLOT

### Syntax

**RESOURCE\_RESERVE\_PER\_SLOT**=*y* | Y

### Description

If Y, `mbat chd` reserves resources based on job slots instead of per-host.

By default, `mbat chd` only reserves resources for parallel jobs on a per-host basis. For example, by default, the command:

```
bsub -n 4 -R "rusage[mem=500]" -q reservation my_j ob
```

requires the job to reserve 500 MB on each host where the job runs.

Some parallel jobs need to reserve resources based on job slots, rather than by host. In this example, if per-slot reservation is enabled by `RESOURCE_RESERVE_PER_SLOT`, the job `my_j ob` must reserve 500 MB of memory for each job slot ( $4 \times 500 = 2$  GB) on the host in order to run.

If `RESOURCE_RESERVE_PER_SLOT` is set, the following command reserves the resource `my_resource` on all 4 job slots instead of only 1 on the host where the job runs:

```
bsub -n 4 -R "my_resource > 0 rusage[my_resource=1]" myj ob
```

## Default

N (Not defined; reserve resources per-host.)

## RUN\_JOB\_FACTOR

### Syntax

**RUN\_JOB\_FACTOR**=*number*

## Description

Used only with fairshare scheduling. Job slots weighting factor.

In the calculation of a user's dynamic share priority, this factor determines the relative importance of the number of job slots reserved and in use by a user.

This parameter can also be set for an individual queue in `l sb. queues`. If defined, the queue value takes precedence.

## Default

3.0

## RUN\_TIME\_DECAY

### Syntax

**RUN\_TIME\_DECAY=Y | y | N | n**

## Description

Used only with fairshare scheduling. Enables decay for run time at the same rate as the decay set by `HIST_HOURS` for cumulative CPU time and historical run time.

In the calculation of a user's dynamic share priority, this factor determines whether run time is decayed.

This parameter can also be set for an individual queue in `l sb. queues`. If defined, the queue value takes precedence.

## Restrictions

Running `badmin reconfig` or restarting `mbatchd` during a job's run time results in the decayed run time being recalculated.

When a suspended job using run time decay is resumed, the decay time is based on the elapsed time.

## Default

N

## RUN\_TIME\_FACTOR

### Syntax

**RUN\_TIME\_FACTOR=*number***

## Description

Used only with fairshare scheduling. Run time weighting factor.

In the calculation of a user's dynamic share priority, this factor determines the relative importance of the total run time of a user's running jobs.

This parameter can also be set for an individual queue in `l sb. queues`. If defined, the queue value takes precedence.

## Default

0.7

## SBD\_SLEEP\_TIME

### Syntax

**SBD\_SLEEP\_TIME**=*seconds*

### Description

The interval at which LSF checks the load conditions of each host, to decide whether jobs on the host must be suspended or resumed.

The job-level resource usage information is updated at a maximum frequency of every SBD\_SLEEP\_TIME seconds.

The update is done only if the value for the CPU time, resident memory usage, or virtual memory usage has changed by more than 10 percent from the previous update or if a new process or process group has been created.

## Default

SBD\_SLEEP\_TIME is set at installation to 15 seconds. If not defined, 30 seconds.

## SCHED\_METRIC\_ENABLE

### Syntax

**SCHED\_METRIC\_ENABLE**=Y | N

### Description

Enable scheduler performance metric collection.

Use `badmi n perfmon stop` and `badmi n perfmon start` to dynamically control performance metric collection.

The update is done only if the value for the CPU time, resident memory usage, or virtual memory usage has changed by more than 10 percent from the previous update or if a new process or process group has been created.

## Default

N

## SCHED\_METRIC\_SAMPLE\_PERIOD

### Syntax

**SCHED\_METRIC\_SAMPLE\_PERIOD**=*seconds*

### Description

Set a default performance metric sampling period in seconds.

Cannot be less than 60 seconds.

lsb.params

Use `badm n perfmon setperiod` to dynamically change performance metric sampling period.

## Default

60 seconds

## SLA\_TIMER

### Syntax

**SLA\_TIMER**=*seconds*

### Description

For EGO-enabled SLA scheduling. Controls how often each service class is evaluated and a network message is sent to EGO communicating host demand.

### Valid values

Positive integer between 2 and 21474847

### Default

0 (Not defined.)

## SSCHED\_ACCT\_DIR

### Syntax

**SSCHED\_ACCT\_DIR**=*directory*

### Description

Used by Platform Session Scheduler (`ssched`).

A universally accessible and writable directory that will store Session Scheduler task accounting files. Each Session Scheduler session (each `ssched` instance) creates one accounting file. Each file contains one accounting entry for each task. The accounting file is named `job_ID.ssched.acct`. If no directory is specified, accounting records are not written.

### Valid values

Specify any string up to 4096 characters long

### Default

Not defined. No task accounting file is created.

## SSCHED\_MAX\_RUNLIMIT

### Syntax

**SSCHED\_MAX\_RUNLIMIT**=*seconds*

### Description

Used by Platform Session Scheduler (`ssched`).

Maximum run time for a task. Users can override this value with a lower value. Specify a value greater than or equal to zero (0).

## Recommended value

For very short-running tasks, a reasonable value is twice the typical runtime. Because LSF does not release slots allocated to the session until all tasks are completed and `ssched` exits, you should avoid setting a large value for `SSCHED_MAX_RUNLIMIT`.

## Valid values

Specify a positive integer between 0 and 2147483645

## Default

600 seconds (10 minutes)

# SSCHED\_MAX\_TASKS

## Syntax

**SSCHED\_MAX\_TASKS**=*integer*

## Description

Used by Platform Session Scheduler (`ssched`).

Maximum number of tasks that can be submitted to Session Scheduler. Session Scheduler exits if this limit is reached. Specify a value greater than or equal to zero (0).

## Valid values

Specify a positive integer between 0 and 2147483645

## Default

50000 tasks

# SSCHED\_REQUEUE\_LIMIT

## Syntax

**SSCHED\_REQUEUE\_LIMIT**=*integer*

## Description

Used by Platform Session Scheduler (`ssched`).

Number of times Session Scheduler tries to requeue a task as a result of the `REQUEUE_EXIT_VALUES` (`ssched -Q`) setting. `SSCHED_REQUEUE_LIMIT=0` means never requeue. Specify a value greater than or equal to zero (0).

## Valid values

Specify a positive integer between 0 and 2147483645

lsb.params

## Default

3 requeue attempts

## SSCHED\_RETRY\_LIMIT

### Syntax

**SSCHED\_RETRY\_LIMIT**=*integer*

### Description

Used by Platform Session Scheduler (ssched).

Number of times Session Scheduler tries to retry a task that fails during dispatch or setup. SSCHED\_RETRY\_LIMIT=0 means never retry. Specify a value greater than or equal to zero (0).

### Valid values

Specify a positive integer between 0 and 2147483645

## Default

3 retry attempts

## SSCHED\_UPDATE\_SUMMARY\_BY\_TASK

### Syntax

**SSCHED\_UPDATE\_SUMMARY\_INTERVAL**=*integer*

### Description

Used by Platform Session Scheduler (ssched).

Update the Session Scheduler task summary via bpost after the specified number of tasks finish. Specify a value greater than or equal to zero (0).

If both SSCHED\_UPDATE\_SUMMARY\_INTERVAL and SSCHED\_UPDATE\_SUMMARY\_BY\_TASK are set to zero (0), bpost is not run.

### Valid values

Specify a positive integer between 0 and 2147483645

## Default

0

### See also

SSCHED\_UPDATE\_SUMMARY\_INTERVAL

## SSCHED\_UPDATE\_SUMMARY\_INTERVAL

### Syntax

**SSCHED\_UPDATE\_SUMMARY\_INTERVAL**=*seconds*

## Description

Used by Platform Session Scheduler (ssched).

Update the Session Scheduler task summary via bpost after the specified number of seconds. Specify a value greater than or equal to zero (0).

If both SSCHED\_UPDATE\_SUMMARY\_INTERVAL and SSCHED\_UPDATE\_SUMMARY\_BY\_TASK are set to zero (0), bpost is not run.

## Valid values

Specify a positive integer between 0 and 2147483645

## Default

60 seconds

## See also

SSCHED\_UPDATE\_SUMMARY\_BY\_TASK

## STRICT\_UG\_CONTROL

### Syntax

**STRICT\_UG\_CONTROL=Y | N**

### Description

When **STRICT\_UG\_CONTROL=Y** is defined:

- Jobs submitted with *-G usergroup* specified can only be controlled by the usergroup administrator of the specified user group.
- user group administrators can be defined for user groups with *al l* as a member

After adding or changing STRICT\_UG\_CONTROL in *lsb.params*, use *badadmin reconfig* to reconfigure your cluster.

### Default

N (Not defined.)

### See also

DEFAULT\_USER\_GROUP, ENFORCE\_ONE\_UG\_LIMIT, ENFORCE\_UG\_TREE

## SUB\_TRY\_INTERVAL

### Syntax

**SUB\_TRY\_INTERVAL=*integer***

### Description

The number of seconds for the requesting client to wait before resubmitting a job. This is sent by *mbatchd* to the client.

lsb.params

## Default

60 seconds

## See also

MAX\_PEND\_JOBS

# SYSTEM\_MAPPING\_ACCOUNT

## Syntax

**SYSTEM\_MAPPING\_ACCOUNT**=*user\_account*

## Description

Enables Windows workgroup account mapping, which allows LSF administrators to map all Windows workgroup users to a single Windows system account, eliminating the need to create multiple users and passwords in LSF. Users can submit and run jobs using their local user names and passwords, and LSF runs the jobs using the mapped system account name and password. With Windows workgroup account mapping, all users have the same permissions because all users map to the same system account.

To specify the user account, include the domain name in uppercase letters (*DOMAIN\_NAME \user\_name*).

Define this parameter for LSF Windows Workgroup installations only.

## Default

Not defined

# USE\_SUSP\_SLOTS

## Syntax

**USE\_SUSP\_SLOTS**=Y | N

## Description

If **USE\_SUSP\_SLOTS**=Y, allows jobs from a low priority queue to use slots held by suspended jobs in a high priority queue, which has a preemption relation with the low priority queue.

Set **USE\_SUSP\_SLOTS**=N to prevent low priority jobs from using slots held by suspended jobs in a high priority queue, which has a preemption relation with the low priority queue.

## Default

Y

# Automatic time-based configuration

Variable configuration is used to automatically change LSF configuration based on time windows. You define automatic configuration changes in `lsb.params` by using if-else constructs and time expressions. After you change the files, reconfigure the cluster with the `badminton reconfig` command.

The expressions are evaluated by LSF every 10 minutes based on `mbatchd` start time. When an expression evaluates true, LSF dynamically changes the configuration based on the associated configuration

statements. Reconfiguration is done in real time without restarting `mbat chd`, providing continuous system availability.

## Example

```
# if 18:30-19:30 is your short job express period, but
# you want all jobs going to the short queue by default
# and be subject to the thresholds of that queue

# for all other hours, normal is the default queue

#if time(18:30-19:30)
DEFAULT_QUEUE=short
#else
DEFAULT_QUEUE=normal
#endif
```

# lsb.queues

The `lsb.queues` file defines batch queues. Numerous controls are available at the queue level to allow cluster administrators to customize site policies.

This file is optional; if no queues are configured, LSF creates a queue named `default`, with all parameters set to default values.

This file is installed by default in `LSB_CONFDIR/cluster_name/configdir`.

## Changing lsb.queues configuration

After making any changes to `lsb.queues`, run `badminton reconfig` to reconfigure `mbatchd`.

Some parameters such as run window and run time limit do not take effect immediately for running jobs unless you run `mbatchd restart` or `sbatchd restart` on the job execution host.

## lsb.queues structure

Each queue definition begins with the line `Begin Queue` and ends with the line `End Queue`. The queue name must be specified; all other parameters are optional.

# ADMINISTRATORS

## Syntax

**ADMINISTRATORS**=*user\_name* | *user\_group* ...

## Description

List of queue administrators. To specify a Windows user account or user group, include the domain name in uppercase letters (`DOMAIN_NAME\user_name` or `DOMAIN_NAME\user_group`).

Queue administrators can perform operations on any user's job in the queue, as well as on the queue itself.

## Default

Not defined. You must be a cluster administrator to operate on this queue.

# APS\_PRIORITY

## Syntax

**APS\_PRIORITY**=**WEIGHT**[[*factor*, *value*] [*subfactor*, *value*]...][...] **LIMIT**[[*factor*, *value*] [*subfactor*, *value*]...][...] **GRACE\_PERIOD**[[*factor*, *value*] [*subfactor*, *value*]...][...]

## Description

Specifies calculation factors for absolute priority scheduling (APS). Pending jobs in the queue are ordered according to the calculated APS value.

If weight of a subfactor is defined, but the weight of parent factor is not defined, the parent factor weight is set as 1.

The **WEIGHT** and **LIMIT** factors are floating-point values. Specify a *value* for **GRACE\_PERIOD** in seconds (*values*), minutes (*valuem*), or hours (*valueh*).

The default unit for grace period is hours.

For example, the following sets a grace period of 10 hours for the MEM factor, 10 minutes for the JRIORITY factor, 10 seconds for the QRIORITY factor, and 10 hours (default) for the RSRC factor:

```
GRACE_PERIOD[ [MEM, 10h] [JRIORITY, 10m] [QRIORITY, 10s] [RSRC, 10] ]
```

You cannot specify zero (0) for the WEIGHT, LIMIT, and GRACE\_PERIOD of any factor or subfactor.

APS queues cannot configure cross-queue fairshare (FAIRSHARE\_QUEUES). The QUEUE\_GROUP parameter replaces FAIRSHARE\_QUEUES, which is obsolete in LSF 7.0.

Suspended (bst op) jobs and migrated jobs (bmi g) are always scheduled before pending jobs. For migrated jobs, LSF keeps the existing job priority information.

If LSB\_REQUEUE\_TO\_BOTTOM and LSB\_MIG2PEND are configured in `lsf.conf`, the migrated jobs keep their APS information. When LSB\_REQUEUE\_TO\_BOTTOM and LSB\_MIG2PEND are configured, the migrated jobs need to compete with other pending jobs based on the APS value. If you want to reset the APS value, the you should use `brequeue`, not `bmi g`.

## Default

Not defined

# BACKFILL

## Syntax

**BACKFILL=Y | N**

## Description

If Y, enables backfill scheduling for the queue.

A possible conflict exists if BACKFILL and PREEMPTION are specified together. If `PREEMPT_JOBTYPE = BACKFILL` is set in the `lsb.params` file, a backfill queue can be preemptable. Otherwise a backfill queue cannot be preemptable. If BACKFILL is enabled do not also specify `PREEMPTION = PREEMPTABLE`.

BACKFILL is required for interruptible backfill queues (`INTERRUPTIBLE_BACKFILL=seconds`).

When `MAX_SLOTS_IN_POOL`, `SLOT_RESERVE`, and BACKFILL are defined for the same queue, jobs in the queue cannot backfill using slots reserved by other jobs in the same queue.

## Default

Not defined. No backfilling.

# CHKPNT

## Syntax

**CHKPNT=chkpnt\_dir [chkpnt\_period]**

## Description

Enables automatic checkpointing for the queue. All jobs submitted to the queue are checkpointable.

The checkpoint directory is the directory where the checkpoint files are created. Specify an absolute path or a path relative to CWD, do not use environment variables.

Specify the optional checkpoint period in minutes.

Only running members of a chunk job can be checkpointed.

If checkpoint-related configuration is specified in both the queue and an application profile, the application profile setting overrides queue level configuration.

If checkpoint-related configuration is specified in the queue, application profile, and at job level:

- Application-level and job-level parameters are merged. If the same parameter is defined at both job-level and in the application profile, the job-level value overrides the application profile value.
- The merged result of job-level and application profile settings override queue-level configuration.

To enable checkpointing of MultiCluster jobs, define a checkpoint directory in both the send-jobs and receive-jobs queues (CHKPNT in `lsb.queues`), or in an application profile (CHKPNT\_DIR, CHKPNT\_PERIOD, CHKPNT\_INITPERIOD, CHKPNT\_METHOD in `lsb.applications`) of both submission cluster and execution cluster. LSF uses the directory specified in the execution cluster.

To make a MultiCluster job checkpointable, both submission and execution queues must enable checkpointing, and the application profile or queue setting on the execution cluster determines the checkpoint directory. Checkpointing is not supported if a job runs on a leased host.

The file path of the checkpoint directory can contain up to 4000 characters for UNIX and Linux, or up to 255 characters for Windows, including the directory and file name.

## Default

Not defined

# CHUNK\_JOB\_SIZE

## Syntax

**CHUNK\_JOB\_SIZE**=*integer*

## Description

Chunk jobs only. Enables job chunking and specifies the maximum number of jobs allowed to be dispatched together in a chunk. Specify a positive integer greater than 1.

The ideal candidates for job chunking are jobs that have the same host and resource requirements and typically take 1 to 2 minutes to run.

Job chunking can have the following advantages:

- Reduces communication between `sbatchd` and `mbatchd` and reduces scheduling overhead in `mbschd`.
- Increases job throughput in `mbatchd` and CPU utilization on the execution hosts.

However, throughput can deteriorate if the chunk job size is too big. Performance may decrease on queues with `CHUNK_JOB_SIZE` greater than 30. You should evaluate the chunk job size on your own systems for best performance.

With MultiCluster job forwarding model, this parameter does not affect MultiCluster jobs that are forwarded to a remote cluster.

## Compatibility

This parameter is ignored in the following kinds of queues and applications:

- Interactive (INTERACTIVE=ONLY parameter)
- CPU limit greater than 30 minutes (CPULIMIT parameter)
- Run limit greater than 30 minutes (RUNLIMIT parameter)
- Runtime estimate greater than 30 minutes (RUNTIME parameter in lsb. applications only)

If CHUNK\_JOB\_DURATION is set in lsb. params, chunk jobs are accepted regardless of the value of CPULIMIT, RUNLIMIT or RUNTIME.

## Example

The following configures a queue named chunk, which dispatches up to 4 jobs in a chunk:

```
Begin Queue
QUEUE_NAME      = chunk
PRIORITY        = 50
CHUNK_JOB_SIZE  = 4
End Queue
```

## Default

Not defined

# COMMITTED\_RUN\_TIME\_FACTOR

## Syntax

**COMMITTED\_RUN\_TIME\_FACTOR**=*number*

## Description

Used only with fairshare scheduling. Committed run time weighting factor.

In the calculation of a user's dynamic priority, this factor determines the relative importance of the committed run time in the calculation. If the -W option of bsub is not specified at job submission and a RUNLIMIT has not been set for the queue, the committed run time is not considered.

If undefined, the cluster-wide value from the lsb. params parameter of the same name is used.

## Valid values

Any positive number between 0.0 and 1.0

## Default

Not defined.

# CORELIMIT

## Syntax

**CORELIMIT**=*integer*

## Description

The per-process (hard) core file size limit (in KB) for all of the processes belonging to a job from this queue (see `getrlimit(2)`).

## Default

Unlimited

# CPULIMIT

## Syntax

**CPULIMIT**=[*default\_limit*] *maximum\_limit*

where *default\_limit* and *maximum\_limit* are:

[*hour*:]*minute*[/*host\_name* | /*host\_model*]

## Description

Maximum normalized CPU time and optionally, the default normalized CPU time allowed for all processes of a job running in this queue. The name of a host or host model specifies the CPU time normalization host to use.

Limits the total CPU time the job can use. This parameter is useful for preventing runaway jobs or jobs that use up too many resources.

When the total CPU time for the whole job has reached the limit, a SIGXCPU signal is sent to all processes belonging to the job. If the job has no signal handler for SIGXCPU, the job is killed immediately. If the SIGXCPU signal is handled, blocked, or ignored by the application, then after the grace period expires, LSF sends SIGINT, SIGTERM, and SIGKILL to the job to kill it.

If a job dynamically spawns processes, the CPU time used by these processes is accumulated over the life of the job.

Processes that exist for fewer than 30 seconds may be ignored.

By default, if a default CPU limit is specified, jobs submitted to the queue without a job-level CPU limit are killed when the default CPU limit is reached.

If you specify only one limit, it is the maximum, or hard, CPU limit. If you specify two limits, the first one is the default, or soft, CPU limit, and the second one is the maximum CPU limit. The number of minutes may be greater than 59. Therefore, three and a half hours can be specified either as 3:30 or 210.

If no host or host model is given with the CPU time, LSF uses the default CPU time normalization host defined at the queue level (DEFAULT\_HOST\_SPEC in `lsb.queues`) if it has been configured, otherwise uses the default CPU time normalization host defined at the cluster level (DEFAULT\_HOST\_SPEC in `lsb.params`) if it has been configured, otherwise uses the host with the largest CPU factor (the fastest host in the cluster).

On Windows, a job that runs under a CPU time limit may exceed that limit by up to SBD\_SLEEP\_TIME. This is because `sbat chd` periodically checks if the limit has been exceeded.

On UNIX systems, the CPU limit can be enforced by the operating system at the process level.

You can define whether the CPU limit is a per-process limit enforced by the OS or a per-job limit enforced by LSF with LSB\_JOB\_CPULIMIT in `lsf.conf`.

Jobs submitted to a chunk job queue are not chunked if CPULIMIT is greater than 30 minutes.

## Default

Unlimited

# CPU\_TIME\_FACTOR

## Syntax

**CPU\_TIME\_FACTOR**=*number*

## Description

Used only with fairshare scheduling. CPU time weighting factor.

In the calculation of a user's dynamic share priority, this factor determines the relative importance of the cumulative CPU time used by a user's jobs.

If undefined, the cluster-wide value from the `lsb.params` parameter of the same name is used.

## Default

0.7

# DATALIMIT

## Syntax

**DATALIMIT**=[*default\_limit*] *maximum\_limit*

## Description

The per-process data segment size limit (in KB) for all of the processes belonging to a job from this queue (see `getrlimit(2)`).

By default, if a default data limit is specified, jobs submitted to the queue without a job-level data limit are killed when the default data limit is reached.

If you specify only one limit, it is the maximum, or hard, data limit. If you specify two limits, the first one is the default, or soft, data limit, and the second one is the maximum data limit

## Default

Unlimited

# DEFAULT\_EXTSCHED

## Syntax

**DEFAULT\_EXTSCHED**=*external\_scheduler\_options*

## Description

Specifies default external scheduling options for the queue.

-extsched options on the `bsub` command are merged with `DEFAULT_EXTSCHED` options, and -extsched options override any conflicting queue-level options set by `DEFAULT_EXTSCHED`.

## Default

Not defined

# DEFAULT\_HOST\_SPEC

## Syntax

**DEFAULT\_HOST\_SPEC**=*host\_name / host\_model*

## Description

The default CPU time normalization host for the queue.

The CPU factor of the specified host or host model is used to normalize the CPU time limit of all jobs in the queue, unless the CPU time normalization host is specified at the job level.

## Default

Not defined. The queue uses the DEFAULT\_HOST\_SPEC defined in `lsb.params`. If DEFAULT\_HOST\_SPEC is not defined in either file, LSF uses the fastest host in the cluster.

# DESCRIPTION

## Syntax

**DESCRIPTION**=*text*

## Description

Description of the job queue displayed by `bqueues -l`.

This description should clearly describe the service features of this queue, to help users select the proper queue for each job.

The text can include any characters, including white space. The text can be extended to multiple lines by ending the preceding line with a backslash (`\`). The maximum length for the text is 512 characters.

# DISPATCH\_ORDER

## Syntax

**DISPATCH\_ORDER**=**QUEUE**

## Description

Defines an *ordered* cross-queue fairshare set. DISPATCH\_ORDER indicates that jobs are dispatched according to the order of queue priorities first, then user fairshare priority.

By default, a user has the same priority across the master and slave queues. If the same user submits several jobs to these queues, user priority is calculated by taking into account all the jobs the user has submitted across the master-slave set.

If DISPATCH\_ORDER=QUEUE is set in the master queue, jobs are dispatched according to queue priorities first, then user priority. Jobs from users with lower fairshare priorities who have pending jobs in higher priority queues are dispatched before jobs in lower priority queues. This avoids having users with higher fairshare priority getting jobs dispatched from low-priority queues.

Jobs in queues having the same priority are dispatched according to user priority.

Queues that are not part of the cross-queue fairshare can have any priority; they are not limited to fall outside of the priority range of cross-queue fairshare queues.

## Default

Not defined

# DISPATCH\_WINDOW

## Syntax

**DISPATCH\_WINDOW**=*time\_window* ...

## Description

The time windows in which jobs from this queue are dispatched. Once dispatched, jobs are no longer affected by the dispatch window.

## Default

Not defined. Dispatch window is always open.

# ENABLE\_HIST\_RUN\_TIME

## Syntax

**ENABLE\_HIST\_RUN\_TIME**=*y* | **Y** | *n* | **N**

## Description

Used only with fairshare scheduling. If set, enables the use of historical run time in the calculation of fairshare scheduling priority.

If undefined, the cluster-wide value from the `lsb.params` parameter of the same name is used.

## Default

Not defined.

# EXCLUSIVE

## Syntax

**EXCLUSIVE**=**Y** | **N** | **CU**[*cu\_type*]

## Description

If **Y**, specifies an exclusive queue.

If **CU**, **CU[]**, or **CU**[*cu\_type*], specifies an exclusive queue as well as a queue exclusive to compute units of type *cu\_type* (as defined in `lsb.params`). If no type is specified, the default compute unit type is used.

Jobs submitted to an exclusive queue with `bsub -x` are only dispatched to a host that has no other LSF jobs running. Jobs submitted to a compute unit exclusive queue with `bsub -R "cu[excl]"` only run on a compute unit that has no other jobs running.

For hosts shared under the MultiCluster resource leasing model, jobs are not dispatched to a host that has LSF jobs running, even if the jobs are from another cluster.

## Default

N

# FAIRSHARE

## Syntax

**FAIRSHARE=USER\_SHARES**[[*user, number\_shares*] ...]

- Specify at least one user share assignment.
- Enclose the list in square brackets, as shown.
- Enclose each user share assignment in square brackets, as shown.
- *user*: Specify users who are also configured to use queue. You can assign the shares to:
  - A single user (specify *user\_name*). To specify a Windows user account, include the domain name in uppercase letters (*DOMAIN\_NAME\user\_name*).
  - Users in a group, individually (specify *group\_name@*) or collectively (specify *group\_name*). To specify a Windows user group, include the domain name in uppercase letters (*DOMAIN\_NAME\group\_name*).
  - Users not included in any other share assignment, individually (specify the keyword *default*) or collectively (specify the keyword *others*)
    - By default, when resources are assigned collectively to a group, the group members compete for the resources on a first-come, first-served (FCFS) basis. You can use hierarchical fairshare to further divide the shares among the group members.
    - When resources are assigned to members of a group individually, the share assignment is recursive. Members of the group and of all subgroups always compete for the resources according to FCFS scheduling, regardless of hierarchical fairshare policies.
- *number\_shares*
  - Specify a positive integer representing the number of shares of the cluster resources assigned to the user.
  - The number of shares assigned to each user is only meaningful when you compare it to the shares assigned to other users or to the total number of shares. The total number of shares is just the sum of all the shares assigned in each share assignment.

## Description

Enables queue-level user-based fairshare and specifies share assignments. Only users with share assignments can submit jobs to the queue.

## Compatibility

Do not configure hosts in a cluster to use fairshare at both queue and host levels. However, you can configure user-based fairshare and queue-based fairshare together.

## Default

Not defined. No fairshare.

# FAIRSHARE\_ADJUSTMENT\_FACTOR

## Syntax

**FAIRSHARE\_ADJUSTMENT\_FACTOR**=*number*

## Description

Used only with fairshare scheduling. Fairshare adjustment plugin weighting factor.

In the calculation of a user's dynamic share priority, this factor determines the relative importance of the user-defined adjustment made in the fairshare plugin (libfairshareadjust.\*).

A positive float number both enables the fairshare plugin and acts as a weighting factor.

If undefined, the cluster-wide value from the `lsb.params` parameter of the same name is used.

## Default

Not defined.

# FAIRSHARE\_QUEUES

## Syntax

**FAIRSHARE\_QUEUES**=*queue\_name*[*queue\_name* ...]

## Description

Defines cross-queue fairshare. When this parameter is defined:

- The queue in which this parameter is defined becomes the “*master queue*”.
- Queues listed with this parameter are “*slave queues*” and inherit the fairshare policy of the master queue.
- A user has the same priority across the master and slave queues. If the same user submits several jobs to these queues, user priority is calculated by taking into account all the jobs the user has submitted across the master-slave set.

## Notes

- By default, the `PRIORITY` range defined for queues in cross-queue fairshare cannot be used with any other queues. For example, you have 4 queues: `queue1`, `queue2`, `queue3`, `queue4`. You configure cross-queue fairshare for `queue1`, `queue2`, `queue3` and assign priorities of 30, 40, 50 respectively.
- By default, the priority of `queue4` (which is not part of the cross-queue fairshare) cannot fall between the priority range of the cross-queue fairshare queues (30-50). It can be any number up to 29 or higher than 50. It does not matter if `queue4` is a fairshare queue or FCFS queue. If `DISPATCH_ORDER=QUEUE` is set in the master queue, the priority of `queue4` (which is not part of the cross-queue fairshare) can be any number, including a priority falling between the priority range of the cross-queue fairshare queues (30-50).
- `FAIRSHARE` must be defined in the master queue. If it is also defined in the queues listed in `FAIRSHARE_QUEUES`, it is ignored.
- Cross-queue fairshare can be defined more than once within `lsb.queues`. You can define several sets of master-slave queues. However, a queue cannot belong to more than one master-slave set. For example, you can define:

- In queue normal : FAIRSHARE\_QUEUES=short license
- In queue priority: FAIRSHARE\_QUEUES=night owners

---

### Restriction:

You cannot, however, define night, owners, or priority as slaves in the queue normal; or normal, short and license as slaves in the priority queue; or short, license, night, owners as master queues of their own.

- Cross-queue fairshare cannot be used with host partition fairshare. It is part of queue-level fairshare.
- Cross-queue fairshare cannot be used with absolute priority scheduling.

## Default

Not defined

## FILELIMIT

### Syntax

**FILELIMIT**=*integer*

### Description

The per-process (hard) file size limit (in KB) for all of the processes belonging to a job from this queue (see `getrlimit(2)`).

## Default

Unlimited

## HIST\_HOURS

### Syntax

**HIST\_HOURS**=*hours*

### Description

Used only with fairshare scheduling. Determines a rate of decay for cumulative CPU time, run time, and historical run time.

To calculate dynamic user priority, LSF scales the actual CPU time and the run time using a decay factor, so that 1 hour of recently-used time is equivalent to 0.1 hours after the specified number of hours has elapsed.

To calculate dynamic user priority with decayed run time and historical run time, LSF scales the accumulated run time of finished jobs and run time of running jobs using the same decay factor, so that 1 hour of recently-used time is equivalent to 0.1 hours after the specified number of hours has elapsed.

When **HIST\_HOURS=0**, CPU time and run time accumulated by running jobs is not decayed.

If undefined, the cluster-wide value from the `lsb.params` parameter of the same name is used.

## Default

Not defined.

# HJOB\_LIMIT

## Syntax

**HJOB\_LIMIT**=*integer*

## Description

Per-host job slot limit.

Maximum number of job slots that this queue can use on any host. This limit is configured per host, regardless of the number of processors it may have.

This may be useful if the queue dispatches jobs that require a node-locked license. If there is only one node-locked license per host then the system should not dispatch more than one job to the host even if it is a multiprocessor host.

## Example

The following runs a maximum of one job on each of hostA, hostB, and hostC:

```
Begin Queue
```

```
...
```

```
HJOB_LIMIT = 1
```

```
HOSTS=hostA hostB hostC
```

```
...
```

```
End Queue
```

## Default

Unlimited

# HOSTS

## Syntax

**HOSTS**=*host\_list* | none

- *host\_list* is a space-separated list of the following items:
  - *host\_name*[*@cluster\_name*][*!*] | *+pref\_level*
  - *host\_partition*[*+pref\_level*]
  - *host\_group*[*!*] | *+pref\_level*
  - *compute\_unit*[*!*] | *+pref\_level*
  - [~]*host\_name*
  - [~]*host\_group*
  - [~]*compute\_unit*
- The list can include the following items only once:
  - all *@cluster\_name*
  - others[*+pref\_level*]
  - all
  - all remote

- The `none` keyword is only used with the MultiCluster job forwarding model, to specify a remote-only queue.

## Description

A space-separated list of hosts on which jobs from this queue can be run.

If compute units, host groups, or host partitions are included in the list, the job can run on any host in the unit, group, or partition. All the members of the host list should either belong to a single host partition or not belong to any host partition. Otherwise, job scheduling may be affected.

Some items can be followed by a plus sign (+) and a positive number to indicate the preference for dispatching a job to that host. A higher number indicates a higher preference. If a host preference is not given, it is assumed to be 0. If there are multiple candidate hosts, LSF dispatches the job to the host with the highest preference; hosts at the same level of preference are ordered by load.

If compute units, host groups, or host partitions are assigned a preference, each host in the unit, group, or partition has the same preference.

Use the keyword `others` to include all hosts not explicitly listed.

Use the keyword `all` to include all hosts not explicitly excluded.

Use the keyword `all@cluster_name hostgroup_name` or `all remote hostgroup_name` to include lease in hosts.

Use the not operator (-) to exclude hosts from the `all` specification in the queue. This is useful if you have a large cluster but only want to exclude a few hosts from the queue definition.

The not operator can only be used with the `all` keyword. It is *not* valid with the keywords `others` and `none`.

The not operator (-) can be used to exclude host groups.

For parallel jobs, specify first execution host candidates when you want to ensure that a host has the required resources or runtime environment to handle processes that run on the first execution host.

To specify one or more hosts, host groups, or compute units as first execution host candidates, add the exclamation point (!) symbol after the name.

Follow these guidelines when you specify first execution host candidates:

- If you specify a compute unit or host group, you must first define the unit or group in the file `lsb.hosts`.
- Do not specify a dynamic host group as a first execution host.
- Do not specify "all," "allremote," or "others," or a host partition as a first execution host.
- Do not specify a preference (+) for a host identified by (!) as a first execution host candidate.
- For each parallel job, specify enough regular hosts to satisfy the CPU requirement for the job. Once LSF selects a first execution host for the current job, the other first execution host candidates
  - Become unavailable to the current job
  - Remain available to other jobs as either regular or first execution hosts
- You cannot specify first execution host candidates when you use the `brun` command.

---

### Restriction:

If you have enabled EGO, host groups and compute units are not honored.

With MultiCluster resource leasing model, use the format *host\_name@cluster\_name* to specify a borrowed host. LSF does not validate the names of remote hosts. The keyword *others* indicates all local hosts not explicitly listed. The keyword *all* indicates all local hosts not explicitly excluded. Use the keyword *all remote* to specify all hosts borrowed from all remote clusters. Use *all@cluster\_name* to specify the group of all hosts borrowed from one remote cluster. You cannot specify a host group or partition that includes remote resources, unless it uses the keyword *all remote* to include all remote hosts. You cannot specify a compute unit that includes remote resources.

With MultiCluster resource leasing model, the not operator (*~*) can be used to exclude local hosts or host groups. You cannot use the not operator (*~*) with remote hosts.

---

### Restriction:

Hosts that participate in queue-based fairshare cannot be in a host partition.

---

## Behavior with host intersection

Host preferences specified by *bsub -m* combine intelligently with the queue specification and advance reservation hosts. The jobs run on the hosts that are both specified at job submission and belong to the queue or have advance reservation.

### Example 1

```
HOSTS=hostA+1 hostB hostC+1 hostD+3
```

This example defines three levels of preferences: run jobs on *hostD* as much as possible, otherwise run on either *hostA* or *hostC* if possible, otherwise run on *hostB*. Jobs should not run on *hostB* unless all other hosts are too busy to accept more jobs.

### Example 2

```
HOSTS=hostD+1 others
```

Run jobs on *hostD* as much as possible, otherwise run jobs on the least-loaded host available.

With MultiCluster resource leasing model, this queue does not use borrowed hosts.

### Example 3

```
HOSTS=all ~hostA
```

Run jobs on all hosts in the cluster, except for *hostA*.

With MultiCluster resource leasing model, this queue does not use borrowed hosts.

### Example 4

```
HOSTS=Group1 ~hostA hostB hostC
```

Run jobs on *hostB*, *hostC*, and all hosts in *Group1* except for *hostA*.

With MultiCluster resource leasing model, this queue uses borrowed hosts if *Group1* uses the keyword *all remote*.

## Example 5

```
HOSTS=hostA! hostB+ hostC hostgroup1!
```

Runs parallel jobs using either hostA or a host defined in hostgroup1 as the first execution host. If the first execution host cannot run the entire job due to resource requirements, runs the rest of the job on hostB. If hostB is too busy to accept the job, or if hostB does not have enough resources to run the entire job, runs the rest of the job on hostC.

## Example 6

```
HOSTS=computeunit1! hostB hostC
```

Runs parallel jobs using a host in computeunit1 as the first execution host. If the first execution host cannot run the entire job due to resource requirements, runs the rest of the job on other hosts in computeunit1 followed by hostB and finally hostC.

## Example 7

```
HOSTS=hostgroup1! computeunitA computeunitB computeunitC
```

Runs parallel jobs using a host in hostgroup1 as the first execution host. If additional hosts are required, runs the rest of the job on other hosts in the same compute unit as the first execution host, followed by hosts in the remaining compute units in the order they are defined in the lsb.hosts ComputeUnit section.

## Default

all (the queue can use all hosts in the cluster, and every host has equal preference)

With MultiCluster resource leasing model, this queue can use all local hosts, but no borrowed hosts.

# IGNORE\_DEADLINE

## Syntax

```
IGNORE_DEADLINE=Y
```

## Description

If Y, disables deadline constraint scheduling (starts all jobs regardless of deadline constraints).

# IMPT\_JOBKLG

## Syntax

```
IMPT_JOBKLG=integer | infinite
```

## Description

MultiCluster job forwarding model only. Specifies the MultiCluster pending job limit for a receive-jobs queue. This represents the maximum number of MultiCluster jobs that can be pending in the queue; once the limit has been reached, the queue stops accepting jobs from remote clusters.

Use the keyword *infinite* to make the queue accept an unlimited number of pending MultiCluster jobs.

## Default

50

# INTERACTIVE

## Syntax

**INTERACTIVE=YES | NO | ONLY**

## Description

YES causes the queue to accept both interactive and non-interactive batch jobs, NO causes the queue to reject interactive batch jobs, and ONLY causes the queue to accept interactive batch jobs and reject non-interactive batch jobs.

Interactive batch jobs are submitted via `bsub -I`.

## Default

YES. The queue accepts both interactive and non-interactive jobs.

# INTERRUPTIBLE\_BACKFILL

## Syntax

**INTERRUPTIBLE\_BACKFILL=*seconds***

## Description

Configures interruptible backfill scheduling policy, which allows reserved job slots to be used by low priority small jobs that are terminated when the higher priority large jobs are about to start.

There can only be one interruptible backfill queue. It should be the lowest priority queue in the cluster.

Specify the minimum number of seconds for the job to be considered for backfilling. This minimal time slice depends on the specific job properties; it must be longer than at least one useful iteration of the job. Multiple queues may be created if a site has jobs of distinctively different classes.

An interruptible backfill job:

- Starts as a regular job and is killed when it exceeds the queue runtime limit, or
- Is started for backfill whenever there is a backfill time slice longer than the specified minimal time, and killed before the slot-reservation job is about to start

The queue `RUNLIMIT` corresponds to a maximum time slice for backfill, and should be configured so that the wait period for the new jobs submitted to the queue is acceptable to users. 10 minutes of runtime is a common value.

You should configure `REQUEUE_EXIT_VALUES` for interruptible backfill queues.

`BACKFILL` and `RUNLIMIT` must be configured in the queue. The queue is disabled if `BACKFILL` and `RUNLIMIT` are not configured.

## Assumptions and limitations:

- The interruptible backfill job holds the slot-reserving job start until its calculated start time, in the same way as a regular backfill job. The interruptible backfill job are not preempted in any way other than being killed when its time come.

- While the queue is checked for the consistency of interruptible backfill, backfill and runtime specifications, the requeue exit value clause is not verified, nor executed automatically. Configure requeue exit values according to your site policies.
- The interruptible backfill job must be able to do at least one unit of useful calculations and save its data within the minimal time slice, and be able to continue its calculations after it has been restarted
- Interruptible backfill paradigm does not explicitly prohibit running parallel jobs, distributed across multiple nodes; however, the chance of success of such job is close to zero.

## Default

Not defined. No interruptible backfilling.

# JOB\_ACCEPT\_INTERVAL

## Syntax

**JOB\_ACCEPT\_INTERVAL**=*integer*

## Description

The number you specify is multiplied by the value of `lsb.params.MBD_SLEEP_TIME` (60 seconds by default). The result of the calculation is the number of seconds to wait after dispatching a job to a host, before dispatching a second job to the same host.

If 0 (zero), a host may accept more than one job in each dispatch turn. By default, there is no limit to the total number of jobs that can run on a host, so if this parameter is set to 0, a very large number of jobs might be dispatched to a host all at once. This can overload your system to the point that it is unable to create any more processes. It is not recommended to set this parameter to 0.

`JOB_ACCEPT_INTERVAL` set at the queue level (`lsb.queues`) overrides `JOB_ACCEPT_INTERVAL` set at the cluster level (`lsb.params`).

---

### Note:

The parameter `JOB_ACCEPT_INTERVAL` only applies when there are running jobs on a host. A host running a short job which finishes before `JOB_ACCEPT_INTERVAL` has elapsed is free to accept a new job without waiting.

---

## Default

Not defined. The queue uses `JOB_ACCEPT_INTERVAL` defined in `lsb.params`, which has a default value of 1.

# JOB\_ACTION\_WARNING\_TIME

## Syntax

**JOB\_ACTION\_WARNING\_TIME**=[*hour*:]*minute*

## Description

Specifies the amount of time before a job control action occurs that a job warning action is to be taken. For example, 2 minutes before the job reaches runtime limit or termination deadline, or the queue's run window is closed, an URG signal is sent to the job.

Job action warning time is not normalized.

A job action warning time must be specified with a job warning action in order for job warning to take effect.

The warning time specified by the `bsub -wt` option overrides `JOB_ACTION_WARNING_TIME` in the queue. `JOB_ACTION_WARNING_TIME` is used as the default when no command line option is specified.

## Example

```
JOB_ACTION_WARNING_TIME=2
```

## Default

Not defined

# JOB\_CONTROLS

## Syntax

```
JOB_CONTROLS=SUSPEND[signal | command] CHKPNT RESUME[signal | command] TERMINATE  
[signal | command] CHKPNT
```

- *signal* is a UNIX signal name (for example, `SIGTSTP` or `SIGTERM`). The specified signal is sent to the job. The same set of signals is not supported on all UNIX systems. To display a list of the symbolic names of the signals (without the `SIG` prefix) supported on your system, use the `kill -l` command.
- *command* specifies a `/bin/sh` command line to be invoked.

---

### Restriction:

Do not quote the command line inside an action definition. Do not specify a signal followed by an action that triggers the same signal. For example, do not specify `JOB_CONTROLS=TERMINATE[ kill ]` or `JOB_CONTROLS=TERMINATE[ brequeue ]`. This causes a deadlock between the signal and the action.

- `CHKPNT` is a special action, which causes the system to checkpoint the job. Only valid for `SUSPEND` and `TERMINATE` actions:
  - If the `SUSPEND` action is `CHKPNT`, the job is checkpointed and then stopped by sending the `SIGSTOP` signal to the job automatically.
  - If the `TERMINATE` action is `CHKPNT`, then the job is checkpointed and killed automatically.

## Description

Changes the behavior of the `SUSPEND`, `RESUME`, and `TERMINATE` actions in LSF.

- The contents of the configuration line for the action are run with `/bin/sh -c` so you can use shell features in the command.
- The standard input, output, and error of the command are redirected to the `NULL` device, so you cannot tell directly whether the command runs correctly. The default null device on UNIX is `/dev/null`.
- The command is run as the user of the job.
- All environment variables set for the job are also set for the command action. The following additional environment variables are set:

- `LSB_JOBPGIDS`: a list of current process group IDs of the job
- `LSB_JOBPIIDS`: a list of current process IDs of the job
- For the `SUSPEND` action command, the following environment variables are also set:
  - `LSB_SUSP_REASONS`: an integer representing a bitmap of suspending reasons as defined in `lsbatch.h`. The suspending reason can allow the command to take different actions based on the reason for suspending the job.
  - `LSB_SUSP_SUBREASONS`: an integer representing the load index that caused the job to be suspended. When the suspending reason `SUSP_LOAD_REASON` (suspended by load) is set in `LSB_SUSP_REASONS`, `LSB_SUSP_SUBREASONS` set to one of the load index values defined in `lsf.h`. Use `LSB_SUSP_REASONS` and `LSB_SUSP_SUBREASONS` together in your custom job control to determine the exact load threshold that caused a job to be suspended.
- If an additional action is necessary for the `SUSPEND` command, that action should also send the appropriate signal to the application. Otherwise, a job can continue to run even after being suspended by LSF. For example, `JOB_CONTROLS=SUSPEND[kill $LSB_JOBPIIDS; command]`
- If you set preemption with the signal `SIGTSTP` you use Platform License Scheduler, define `LIC_SCHED_PREEMPT_STOP=Y` in `lsf.conf` for License Scheduler preemption to work.

## Default

On UNIX, by default, `SUSPEND` sends `SIGTSTP` for parallel or interactive jobs and `SIGSTOP` for other jobs. `RESUME` sends `SIGCONT`. `TERMINATE` sends `SIGINT`, `SIGTERM` and `SIGKILL` in that order.

On Windows, actions equivalent to the UNIX signals have been implemented to do the default job control actions. Job control messages replace the `SIGINT` and `SIGTERM` signals, but only customized applications are able to process them. Termination is implemented by the `TerminateProcess()` system call.

## JOB\_IDLE

### Syntax

`JOB_IDLE=number`

### Description

Specifies a threshold for idle job exception handling. The value should be a number between 0.0 and 1.0 representing CPU time/runtime. If the job idle factor is less than the specified threshold, LSF invokes `LSF_SERVERDIR/eadmin` to trigger the action for a job idle exception.

The minimum job run time before `mbatchd` reports that the job is idle is defined as `DETECT_IDLE_JOB_AFTER` in `lsb.params`.

### Valid values

Any positive number between 0.0 and 1.0

### Example

```
JOB_IDLE=0.10
```

A job idle exception is triggered for jobs with an idle value (CPU time/runtime) less than 0.10.

### Default

Not defined. No job idle exceptions are detected.

# JOB\_OVERRUN

## Syntax

```
JOB_OVERRUN=run_time
```

## Description

Specifies a threshold for job overrun exception handling. If a job runs longer than the specified run time, LSF invokes LSF\_SERVERDIR/eadmin to trigger the action for a job overrun exception.

## Example

```
JOB_OVERRUN=5
```

A job overrun exception is triggered for jobs running longer than 5 minutes.

## Default

Not defined. No job overrun exceptions are detected.

# JOB\_STARTER

## Syntax

```
JOB_STARTER=starter [starter] ["%USRCMD"] [starter]
```

## Description

Creates a specific environment for submitted jobs prior to execution.

*starter* is any executable that can be used to start the job (i.e., can accept the job as an input argument). Optionally, additional strings can be specified.

By default, the user commands run after the job starter. A special string, %USRCMD, can be used to represent the position of the user's job in the job starter command line. The %USRCMD string and any additional commands must be enclosed in quotation marks (" ").

If your job starter script runs on a Windows execution host and includes symbols (like & or |), you can use the JOB\_STARTER\_EXTEND=preservestarter parameter in lsf.conf and set JOB\_STARTER=preservestarter in lsb.queues. A customized userstarter can also be used.

## Example

```
JOB_STARTER=csh -c "%USRCMD; sleep 10"
```

In this case, if a user submits a job

```
% bsub myjob arguments
```

the command that actually runs is:

```
% csh -c "myjob arguments; sleep 10"
```

## Default

Not defined. No job starter is used.

# JOB\_UNDERRUN

## Syntax

```
JOB_UNDERRUN=run_time
```

## Description

Specifies a threshold for job underrun exception handling. If a job exits before the specified number of minutes, LSF invokes `LSF_SERVERDIR/eadmin` to trigger the action for a job underrun exception.

## Example

```
JOB_UNDERRUN=2
```

A job underrun exception is triggered for jobs running less than 2 minutes.

## Default

Not defined. No job underrun exceptions are detected.

# JOB\_WARNING\_ACTION

## Syntax

```
JOB_WARNING_ACTION=signal
```

## Description

Specifies the job action to be taken before a job control action occurs. For example, 2 minutes before the job reaches runtime limit or termination deadline, or the queue's run window is closed, an URG signal is sent to the job.

A job warning action must be specified with a job action warning time in order for job warning to take effect.

If `JOB_WARNING_ACTION` is specified, LSF sends the warning action to the job before the actual control action is taken. This allows the job time to save its result before being terminated by the job control action.

The warning action specified by the `bsub -wa` option overrides `JOB_WARNING_ACTION` in the queue. `JOB_WARNING_ACTION` is used as the default when no command line option is specified.

## Example

```
JOB_WARNING_ACTION=URG
```

## Default

Not defined

# *load\_index*

## Syntax

```
load_index=loadSched[/loadStop]
```

Specify `io`, `it`, `ls`, `mem`, `pg`, `r15s`, `r1m`, `r15m`, `swp`, `tmp`, `ut`, or a non-shared custom external load index. Specify multiple lines to configure thresholds for multiple load indices.

Specify `io`, `it`, `ls`, `mem`, `pg`, `r15s`, `r1m`, `r15m`, `swp`, `tmp`, `ut`, or a non-shared custom external load index as a column. Specify multiple columns to configure thresholds for multiple load indices.

## Description

Scheduling and suspending thresholds for the specified dynamic load index.

The `loadSched` condition must be satisfied before a job is dispatched to the host. If a `RESUME_COND` is not specified, the `loadSched` condition must also be satisfied before a suspended job can be resumed.

If the `loadStop` condition is satisfied, a job on the host is suspended.

The `loadSched` and `loadStop` thresholds permit the specification of conditions using simple AND/OR logic. Any load index that does not have a configured threshold has no effect on job scheduling.

LSF does not suspend a job if the job is the only batch job running on the host and the machine is interactively idle (`it > 0`).

The `r15s`, `r1m`, and `r15m` CPU run queue length conditions are compared to the effective queue length as reported by `lsl load -E`, which is normalized for multiprocessor hosts. Thresholds for these parameters should be set at appropriate levels for single processor hosts.

## Example

```
MEM=100/10
```

```
SWAP=200/30
```

These two lines translate into a `loadSched` condition of

```
mem>=100 && swap>=200
```

and a `loadStop` condition of

```
mem < 10 || swap < 30
```

## Default

Not defined

# LOCAL\_MAX\_PREEEXEC\_RETRY

## Syntax

```
LOCAL_MAX_PREEEXEC_RETRY=integer
```

## Description

The maximum number of times to attempt the pre-execution command of a job on the local cluster.

## Valid values

```
0 < MAX_PREEEXEC_RETRY < INFINIT_INT
```

`INFINIT_INT` is defined in `lsf.h`.

## Default

Not defined. The number of preexec retry times is unlimited

## MANDATORY\_EXTSCHED

### Syntax

**MANDATORY\_EXTSCHED**=*external\_scheduler\_options*

### Description

Specifies mandatory external scheduling options for the queue.

-extsched options on the bsub command are merged with MANDATORY\_EXTSCHED options, and MANDATORY\_EXTSCHED options override any conflicting job-level options set by -extsched.

### Default

Not defined

## MAX\_JOB\_PREEMPT

### Syntax

**MAX\_JOB\_PREEMPT**=*integer*

### Description

The maximum number of times a job can be preempted. Applies to queue-based preemption only.

### Valid values

$0 < \text{MAX\_JOB\_PREEMPT} < \text{INFINIT\_INT}$

INFINIT\_INT is defined in lsf.h.

### Default

Not defined. The number of preemption times is unlimited.

## MAX\_JOB\_REQUEUE

### Syntax

**MAX\_JOB\_REQUEUE**=*integer*

### Description

The maximum number of times to requeue a job automatically.

### Valid values

$0 < \text{MAX\_JOB\_REQUEUE} < \text{INFINIT\_INT}$

INFINIT\_INT is defined in lsf.h.

### Default

Not defined. The number of requeue times is unlimited

# MAX\_PREEEXEC\_RETRY

## Syntax

**MAX\_PREEEXEC\_RETRY**=*integer*

## Description

Use REMOTE\_MAX\_PREEEXEC\_RETRY instead. This parameter is maintained for backwards compatibility.

MultiCluster job forwarding model only. The maximum number of times to attempt the pre-execution command of a job from a remote cluster.

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

## Valid values

$0 < \text{MAX\_PREEEXEC\_RETRY} < \text{INFINIT\_INT}$

INFINIT\_INT is defined in lsf.h.

## Default

5

# MAX\_RSCHED\_TIME

## Syntax

**MAX\_RSCHED\_TIME**=*integer* | *infinite*

## Description

MultiCluster job forwarding model only. Determines how long a MultiCluster job stays pending in the execution cluster before returning to the submission cluster. The remote timeout limit in seconds is:

`MAX_RSCHED_TIME * MBD_SLEEP_TIME=timeout`

Specify *infinite* to disable remote timeout (jobs always get dispatched in the correct FCFS order because MultiCluster jobs never get rescheduled, but MultiCluster jobs can be pending in the receive-jobs queue forever instead of being rescheduled to a better queue).

---

### Note:

apply to the queue in the submission cluster (only). This parameter is ignored by the receiving queue.

---

Remote timeout limit never affects advance reservation jobs

Jobs that use an advance reservation always behave as if remote timeout is disabled.

## Default

20 (20 minutes by default)

## MAX\_SLOTS\_IN\_POOL

### Syntax

**MAX\_SLOTS\_IN\_POOL**=*integer*

### Description

Queue-based fairshare only. Maximum number of job slots available in the slot pool the queue belongs to for queue based fairshare.

Defined in the first queue of the slot pool. Definitions in subsequent queues have no effect.

When defined together with other slot limits (QJOB\_LIMIT, HJOB\_LIMIT or UJOB\_LIMIT in `lsb.queues` or queue limits in `lsb.resources`) the lowest limit defined applies.

When MAX\_SLOTS\_IN\_POOL, SLOT\_RESERVE, and BACKFILL are defined for the same queue, jobs in the queue cannot backfill using slots reserved by other jobs in the same queue.

### Valid values

MAX\_SLOTS\_IN\_POOL can be any number from 0 to INFINIT\_INT, where INFINIT\_INT is defined in `lsf.h`.

### Default

Not defined

## MAX\_TOTAL\_TIME\_PREEMPT

### Syntax

**MAX\_TOTAL\_TIME\_PREEMPT**=*integer*

### Description

The accumulated preemption time in minutes after which a job cannot be preempted again, where *minutes* is wall-clock time, not normalized time.

Setting the parameter of the same name in `lsb.applications` overrides this parameter; setting this parameter overrides the parameter of the same name in `lsb.params`.

### Valid values

Any positive integer greater than or equal to one (1)

### Default

Unlimited

## MEMLIMIT

### Syntax

**MEMLIMIT**=[*default\_limit*] *maximum\_limit*

## Description

The per-process (hard) process resident set size limit (in KB) for all of the processes belonging to a job from this queue (see `getrlimit(2)`).

Sets the maximum amount of physical memory (resident set size, RSS) that may be allocated to a process.

By default, if a default memory limit is specified, jobs submitted to the queue without a job-level memory limit are killed when the default memory limit is reached.

If you specify only one limit, it is the maximum, or hard, memory limit. If you specify two limits, the first one is the default, or soft, memory limit, and the second one is the maximum memory limit.

LSF has two methods of enforcing memory usage:

- OS Memory Limit Enforcement
- LSF Memory Limit Enforcement

## OS memory limit enforcement

OS memory limit enforcement is the default MEMLIMIT behavior and does not require further configuration. OS enforcement usually allows the process to eventually run to completion. LSF passes MEMLIMIT to the OS that uses it as a guide for the system scheduler and memory allocator. The system may allocate more memory to a process if there is a surplus. When memory is low, the system takes memory from and lowers the scheduling priority (re-nice) of a process that has exceeded its declared MEMLIMIT. Only available on systems that support `RLIMIT_RSS` for `setrlimit()`.

Not supported on:

- Sun Solaris 2.x
- Windows

## LSF memory limit enforcement

To enable LSF memory limit enforcement, set `LSB_MEMLIMIT_ENFORCE` in `lsf.conf` to `y`. LSF memory limit enforcement explicitly sends a signal to kill a running process once it has allocated memory past MEMLIMIT.

You can also enable LSF memory limit enforcement by setting `LSB_JOB_MEMLIMIT` in `lsf.conf` to `y`. The difference between `LSB_JOB_MEMLIMIT` set to `y` and `LSB_MEMLIMIT_ENFORCE` set to `y` is that with `LSB_JOB_MEMLIMIT`, only the per-job memory limit enforced by LSF is enabled. The per-process memory limit enforced by the OS is disabled. With `LSB_MEMLIMIT_ENFORCE` set to `y`, both the per-job memory limit enforced by LSF and the per-process memory limit enforced by the OS are enabled.

Available for all systems on which LSF collects total memory usage.

## Example

The following configuration defines a queue with a memory limit of 5000 KB:

```
Begin Queue
QUEUE_NAME = default
DESCRIPTION = Queue with memory limit of 5000 kbytes
MEMLIMIT = 5000
End Queue
```

## Default

Unlimited

## MIG

### Syntax

**MIG**=*minutes*

### Description

Enables automatic job migration and specifies the migration threshold for checkpointable or rerunnable jobs, in minutes.

LSF automatically migrates jobs that have been in the SSUSP state for more than the specified number of minutes. Specify a value of 0 to migrate jobs immediately upon suspension. The migration threshold applies to all jobs running on the host.

Job-level command line migration threshold overrides threshold configuration in application profile and queue. Application profile configuration overrides queue level configuration.

When a host migration threshold is specified, and is lower than the value for the job, the queue, or the application, the host value is used..

Members of a chunk job can be migrated. Chunk jobs in WAIT state are removed from the job chunk and put into PEND state.

Does not affect MultiCluster jobs that are forwarded to a remote cluster.

## Default

Not defined. LSF does not migrate checkpointable or rerunnable jobs automatically.

## NEW\_JOB\_SCHED\_DELAY

### Syntax

**NEW\_JOB\_SCHED\_DELAY**=*seconds*

### Description

The number of seconds that a new job waits, before being scheduled. A value of zero (0) means the job is scheduled without any delay.

## Default

2 seconds

## NICE

### Syntax

**NICE**=*integer*

## Description

Adjusts the UNIX scheduling priority at which jobs from this queue execute.

The default value of 0 (zero) maintains the default scheduling priority for UNIX interactive jobs. This value adjusts the run-time priorities for batch jobs on a queue-by-queue basis, to control their effect on other batch or interactive jobs. See the `nicel(1)` manual page for more details.

On Windows, this value is mapped to Windows process priority classes as follows:

- `nicel>=0` corresponds to a priority class of IDLE
- `nicel<0` corresponds to a priority class of NORMAL

Platform LSF on Windows does not support HIGH or REAL-TIME priority classes.

This value is overwritten by the NICE setting in `lsb.applications`, if defined.

## Default

0 (zero)

## NO\_PREEMPT\_INTERVAL

## Syntax

`NO_PREEMPT_INTERVAL=minutes`

## Description

Prevents preemption of jobs for the specified number of minutes of uninterrupted run time, where *minutes* is wall-clock time, not normalized time. `NO_PREEMPT_INTERVAL=0` allows immediate preemption of jobs as soon as they start or resume running.

Setting the parameter of the same name in `lsb.applications` overrides this parameter; setting this parameter overrides the parameter of the same name in `lsb.params`.

## Default

0

## NQS\_QUEUES

## Syntax

`NQS_QUEUES=NQS_queue_name@NQS_host_name ...`

## Description

Makes the queue an NQS forward queue.

*NQS\_host\_name* is an NQS host name that can be the official host name or an alias name known to the LSF master host.

*NQS\_queue\_name* is the name of an NQS destination queue on this host. NQS destination queues are considered for job routing in the order in which they are listed here. If a queue accepts the job, it is routed to that queue. If no queue accepts the job, it remains pending in the NQS forward queue.

`lsb.nqsmaps` must be present for the LSF system to route jobs in this queue to NQS systems.

You must configure `LSB_MAX_NQS_QUEUES` in `lsf.conf` to specify the maximum number of NQS queues allowed in the LSF cluster. This is required for LSF to work with NQS.

Since many features of LSF are not supported by NQS, the following queue configuration parameters are ignored for NQS forward queues: `PJOB_LIMIT`, `POLICIES`, `RUN_WINDOW`, `DISPATCH_WINDOW`, `RUNLIMIT`, `HOSTS`, `MIG`. The application-level `RUNTIME` parameter in `lsb.applications` is also ignored. In addition, scheduling load threshold parameters are ignored because NQS does not provide load information about hosts.

## Default

Not defined

# PJOB\_LIMIT

## Syntax

**PJOB\_LIMIT**=*float*

## Description

Per-processor job slot limit for the queue.

Maximum number of job slots that this queue can use on any processor. This limit is configured per processor, so that multiprocessor hosts automatically run more jobs.

## Default

Unlimited

# POST\_EXEC

## Syntax

**POST\_EXEC**=*command*

## Description

Enables post-execution processing at the queue level. The `POST_EXEC` command runs on the execution host after the job finishes. Post-execution commands can be configured at the application and queue levels. Application-level post-execution commands run *before* queue-level post-execution commands.

The `POST_EXEC` command uses the same environment variable values as the job, and, by default, runs under the user account of the user who submits the job. To run post-execution commands under a different user account (such as `root` for privileged operations), configure the parameter `LSB_PRE_POST_EXEC_USER` in `lsf.sudoers`.

When a job exits with one of the queue's `REQUEUE_EXIT_VALUES`, LSF requeues the job and sets the environment variable `LSB_JOBPEND`. The post-execution command runs after the requeued job finishes.

When the post-execution command is run, the environment variable `LSB_JOBEXIT_STAT` is set to the exit status of the job. If the execution environment for the job cannot be set up, `LSB_JOBEXIT_STAT` is set to 0 (zero).

The command path can contain up to 4094 characters for UNIX and Linux, or up to 255 characters for Windows, including the directory, file name, and expanded values for `%J` (*job\_ID*) and `%I` (*index\_ID*).

For UNIX:

- The pre- and post-execution commands run in the `/tmp` directory under `/bin/sh -c`, which allows the use of shell features in the commands. The following example shows valid configuration lines:  

```
PRE_EXEC= /usr/share/lsf/misc/testq_pre >> /tmp/pre.out
```

```
POST_EXEC= /usr/share/lsf/misc/testq_post | grep -v "Hey!"
```
- LSF sets the `PATH` environment variable to  

```
PATH= /bin /usr/bin /sbin /usr/sbin'
```
- The `stdin`, `stdout`, and `stderr` are set to `/dev/null`
- To allow UNIX users to define their own post-execution commands, an LSF administrator specifies the environment variable `$USER_POSTEXEC` as the `POST_EXEC` command. A user then defines the post-execution command:  

```
setenv USER_POSTEXEC /path_name
```

**Note:**

The path name for the post-execution command must be an absolute path. Do not define `POST_EXEC=$USER_POSTEXEC` when `LSB_PRE_POST_EXEC_USER=root`.

For Windows:

- The pre- and post-execution commands run under `cmd.exe /c`
- The standard input, standard output, and standard error are set to `NULL`
- The `PATH` is determined by the setup of the LSF Service

**Note:**

For post-execution commands that execute on a Windows Server 2003, x64 Edition platform, users must have read and execute privileges for `cmd.exe`.

## Default

Not defined. No post-execution commands are associated with the queue.

## PRE\_EXEC

### Syntax

```
PRE_EXEC=command
```

## Description

Enables pre-execution processing at the queue level. The `PRE_EXEC` command runs on the execution host before the job starts. If the `PRE_EXEC` command exits with a non-zero exit code, LSF requeues the job to the front of the queue.

Pre-execution commands can be configured at the queue, application, and job levels and run in the following order:

1. The queue-level command
2. The application-level or job-level command. If you specify a command at both the application and job levels, the job-level command overrides the application-level command; the application-level command is ignored.

The `PRE_EXEC` command uses the same environment variable values as the job, and runs under the user account of the user who submits the job. To run pre-execution commands under a different user account

(such as `root` for privileged operations), configure the parameter `LSB_PRE_POST_EXEC_USER` in `lsf.sudoers`.

The command path can contain up to 4094 characters for UNIX and Linux, or up to 255 characters for Windows, including the directory, file name, and expanded values for `%J` (*job\_ID*) and `%I` (*index\_ID*).

For UNIX:

- The pre- and post-execution commands run in the `/tmp` directory under `/bin/sh -c`, which allows the use of shell features in the commands. The following example shows valid configuration lines:
 

```
PRE_EXEC= /usr/share/lsf/misc/testq_pre >> /tmp/pre.out
POST_EXEC= /usr/share/lsf/misc/testq_post | grep -v "Hey!"
```
- LSF sets the `PATH` environment variable to
 

```
PATH='/bin /usr/bin /sbin /usr/sbin'
```
- The `stdin`, `stdout`, and `stderr` are set to `/dev/null`

For Windows:

- The pre- and post-execution commands run under `cmd.exe /c`
- The standard input, standard output, and standard error are set to `NULL`
- The `PATH` is determined by the setup of the LSF Service

---

#### Note:

For pre-execution commands that execute on a Windows Server 2003, x64 Edition platform, users must have read and execute privileges for `cmd.exe`.

---

## Default

Not defined. No pre-execution commands are associated with the queue.

# PREEMPTION

## Syntax

```
PREEMPTION=PREEMPTIVE[[low_queue_name[+pref_level]...] PREEMPTION=PREEMPTABLE
[[hi_queue_name...]] PREEMPTION=PREEMPTIVE[[low_queue_name[+pref_level]...]
PREEMPTABLE[[hi_queue_name...]]
```

## Description

### PREEMPTIVE

Enables preemptive scheduling and defines this queue as preemptive. Jobs in this queue preempt jobs from the specified lower-priority queues or from all lower-priority queues if the parameter is specified with no queue names. `PREEMPTIVE` can be combined with `PREEMPTABLE` to specify that jobs in this queue can preempt jobs in lower-priority queues, and can be preempted by jobs in higher-priority queues.

### PREEMPTABLE

Enables preemptive scheduling and defines this queue as preemptable. Jobs in this queue can be preempted by jobs from specified higher-priority queues, or from all higher-priority queues, even if the higher-priority queues are not preemptive. `PREEMPTIVE` can

be combined with `PREEMPTIVE` to specify that jobs in this queue can be preempted by jobs in higher-priority queues, and can preempt jobs in lower-priority queues.

***low\_queue\_name***

Specifies the names of lower-priority queues that can be preempted.

To specify multiple queues, separate the queue names with a space, and enclose the list in a single set of square brackets.

***+pref\_level***

Specifies to preempt this queue before preempting other queues. When multiple queues are indicated with a preference level, an order of preference is indicated: queues with higher relative preference levels are preempted before queues with lower relative preference levels set.

***hi\_queue\_name***

Specifies the names of higher-priority queues that can preempt jobs in this queue.

To specify multiple queues, separate the queue names with a space and enclose the list in a single set of square brackets.

## Example: configure selective, ordered preemption across queues

The following example defines four queues, as follows:

- `high`
  - Has the highest relative priority of 99
  - Jobs from this queue can preempt jobs from all other queues
- `medium`
  - Has the second-highest relative priority at 10
  - Jobs from this queue can preempt jobs from `normal` and `low` queues, beginning with jobs from `low`, as indicated by the preference (+1)
- `normal`
  - Has the second-lowest relative priority, at 5
  - Jobs from this queue can preempt jobs from `low`, and can be preempted by jobs from both `high` and `medium` queues
- `low`
  - Has the lowest relative priority, which is also the default priority, at 1

- Jobs from this queue can be preempted by jobs from all preemptive queues, even though it does not have the PREEMPTABLE keyword set

```

Begin Queue
QUEUE_NAME=hi gh
PREEMPTION=PREEMPTIVE
PRIORITY=99
End Queue

Begin Queue
QUEUE_NAME=medi um
PREEMPTION=PREEMPTIVE[normal low+1]
PRIORITY=10
End Queue

Begin Queue
QUEUE_NAME=normal
PREEMPTION=PREEMPTIVE[low]
PREEMPTABLE[hi gh medi um]
PRIORITY=5
End Queue

Begin Queue
QUEUE_NAME=low
PRIORITY=1
End Queue

```

## PRIORITY

### Syntax

**PRIORITY**=*integer*

### Description

Specifies the relative queue priority for dispatching jobs. A higher value indicates a higher job-dispatching priority, relative to other queues.

LSF schedules jobs from one queue at a time, starting with the highest-priority queue. If multiple queues have the same priority, LSF schedules all the jobs from these queues in first-come, first-served order.

LSF queue priority is independent of the UNIX scheduler priority system for time-sharing processes. In LSF, the NICE parameter is used to set the UNIX time-sharing priority for batch jobs.

#### **integer**

Specify a number greater than or equal to 1, where 1 is the lowest priority.

### Default

1

# PROCESLIMIT

## Syntax

**PROCESLIMIT**=[*default\_limit*] *maximum\_limit*

## Description

Limits the number of concurrent processes that can be part of a job.

By default, if a default process limit is specified, jobs submitted to the queue without a job-level process limit are killed when the default process limit is reached.

If you specify only one limit, it is the maximum, or hard, process limit. If you specify two limits, the first one is the default, or soft, process limit, and the second one is the maximum process limit.

## Default

Unlimited

# PROCLIMIT

## Syntax

**PROCLIMIT**=[*minimum\_limit* [*default\_limit*]] *maximum\_limit*

## Description

Maximum number of slots that can be allocated to a job. For parallel jobs, the maximum number of processors that can be allocated to the job.

Job-level processor limits (`bsub -n`) override queue-level PROCLIMIT. Job-level limits must fall within the maximum and minimum limits of the application profile and the queue. Application-level PROCLIMIT in `lsb.appl icat i ons` overrides queue-level specification.

Optionally specifies the minimum and default number of job slots.

All limits must be positive numbers greater than or equal to 1 that satisfy the following relationship:

$1 \leq \textit{minimum} \leq \textit{default} \leq \textit{maximum}$

You can specify up to three limits in the PROCLIMIT parameter:

Jobs that request fewer slots than the minimum PROCLIMIT or more slots than the maximum PROCLIMIT cannot use the queue and are rejected. If the job requests minimum and maximum job slots, the maximum slots requested cannot be less than the minimum PROCLIMIT, and the minimum slots requested cannot be more than the maximum PROCLIMIT.

## Default

Unlimited, the default number of slots is 1

# QJOB\_LIMIT

## Syntax

**QJOB\_LIMIT**=*integer*

## Description

Job slot limit for the queue. Total number of job slots that this queue can use.

## Default

Unlimited

## QUEUE\_GROUP

### Syntax

**QUEUE\_GROUP**=*queue1, queue2 ...*

## Description

Configures absolute priority scheduling (APS) across multiple queues.

When APS is enabled in the queue with APS\_PRIORITY, the FAIRSHARE\_QUEUES parameter is ignored. The QUEUE\_GROUP parameter replaces FAIRSHARE\_QUEUES, which is obsolete in LSF 7.0.

## Default

Not defined

## QUEUE\_NAME

### Syntax

**QUEUE\_NAME**=*string*

## Description

*Required.* Name of the queue.

Specify any ASCII string up to 59 characters long. You can use letters, digits, underscores (\_) or dashes (-). You cannot use blank spaces. You cannot specify the reserved name default.

## Default

You must specify this parameter to define a queue. The default queue automatically created by LSF is named default.

## RCVJOBS\_FROM

### Syntax

**RCVJOBS\_FROM**=*cluster\_name ...* | **allclusters**

## Description

MultiCluster only. Defines a MultiCluster receive-jobs queue.

Specify cluster names, separated by a space. The administrator of each remote cluster determines which queues in that cluster forward jobs to the local cluster.

Use the keyword **allclusters** to specify any remote cluster.

## Example

```
RCVJOBS_FROM=cluster2 cluster4 cluster6
```

This queue accepts remote jobs from clusters 2, 4, and 6.

## REMOTE\_MAX\_PREEEXEC\_RETRY

### Syntax

```
REMOTE_MAX_PREEEXEC_RETRY=integer
```

### Description

MultiCluster job forwarding model only. Applies to the execution cluster. Define the maximum number of times to attempt the pre-execution command of a job from the remote cluster.

### Valid values

0 - INFINIT\_INT

INFINIT\_INT is defined in `lsf.h`.

### Default

5

## REQUEUE\_EXIT\_VALUES

### Syntax

```
REQUEUE_EXIT_VALUES=[exit_code ...] [EXCLUDE(exit_code ...)]
```

### Description

Enables automatic job requeue and sets the `LSB_EXIT_REQUEUE` environment variable. Use spaces to separate multiple exit codes. Application-level exit values override queue-level values. Job-level exit values (`bsub - Q`) override application-level and queue-level values.

*exit\_code* has the following form:

```
"[all] [~number ...] | [number ...]"
```

The reserved keyword `all` specifies all exit codes. Exit codes are typically between 0 and 255. Use a tilde (~) to exclude specified exit codes from the list.

Jobs are requeued to the head of the queue. The output from the failed run is not saved, and the user is not notified by LSF.

Define an exit code as `EXCLUDE(exit_code)` to enable exclusive job requeue, ensuring the job does not rerun on the samehost. Exclusive job requeue does not work for parallel jobs.

For MultiCluster jobs forwarded to a remote execution cluster, the exit values specified in the submission cluster with the `EXCLUDE` keyword are treated as if they were non-exclusive.

You can also requeue a job if the job is terminated by a signal.

If a job is killed by a signal, the exit value is `128+signal_value`. The sum of 128 and the signal value can be used as the exit code in the parameter `REQUEUE_EXIT_VALUES`.

For example, if you want a job to rerun if it is killed with a signal 9 (SIGKILL), the exit value would be  $128+9=137$ . You can configure the following requeue exit value to allow a job to be requeue if it was kill by signal 9:

```
REQUEUE_EXIT_VALUES=137
```

If `mbat chd` is restarted, it does not remember the previous hosts from which the job exited with an exclusive requeue exit code. In this situation, it is possible for a job to be dispatched to hosts on which the job has previously exited with an exclusive exit code.

You should configure `REQUEUE_EXIT_VALUES` for interruptible backfill queues (`INTERRUPTIBLE_BACKFILL=seconds`).

## Example

```
REQUEUE_EXIT_VALUES=30 EXCLUDE(20)
```

means that jobs with exit code 30 are requeued, jobs with exit code 20 are requeued exclusively, and jobs with any other exit code are not requeued.

## Default

Not defined. Jobs are not requeued.

# RERUNNABLE

## Syntax

```
RERUNNABLE=yes | no
```

## Description

If yes, enables automatic job rerun (restart).

Rerun is disabled when `RERUNNABLE` is set to no. The yes and no arguments are not case sensitive.

For MultiCluster jobs, the setting in the submission queue is used, and the setting in the execution queue is ignored.

Members of a chunk job can be rerunnable. If the execution host becomes unavailable, rerunnable chunk job members are removed from the job chunk and dispatched to a different execution host.

## Default

no

# RESOURCE\_RESERVE

## Syntax

```
RESOURCE_RESERVE=MAX_RESERVE_TIME[integer]
```

## Description

Enables processor reservation and memory reservation for pending jobs for the queue. Specifies the number of dispatch turns (`MAX_RESERVE_TIME`) over which a job can reserve job slots and memory.

Overrides the `SLOT_RESERVE` parameter. If both `RESOURCE_RESERVE` and `SLOT_RESERVE` are defined in the same queue, an error is displayed when the cluster is reconfigured, and `SLOT_RESERVE`

is ignored. Job slot reservation for parallel jobs is enabled by `RESOURCE_RESERVE` if the LSF scheduler plugin module names for both resource reservation and parallel batch jobs (`schmod_parallel` and `schmod_reserve`) are configured in the `lsb.modules` file: The `schmod_parallel` name *must* come before `schmod_reserve` in `lsb.modules`.

If a job has not accumulated enough memory or job slots to start by the time `MAX_RESERVE_TIME` expires, it releases all its reserved job slots or memory so that other pending jobs can run. After the reservation time expires, the job cannot reserve memory or slots for one scheduling session, so other jobs have a chance to be dispatched. After one scheduling session, the job can reserve available memory and job slots again for another period specified by `MAX_RESERVE_TIME`.

If `BACKFILL` is configured in a queue, and a run limit is specified with `-W` on `bsub` or with `RUNLIMIT` in the queue, backfill jobs can use the accumulated memory reserved by the other jobs in the queue, as long as the backfill job can finish before the predicted start time of the jobs with the reservation.

Unlike slot reservation, which only applies to parallel jobs, memory reservation and backfill on memory apply to sequential and parallel jobs.

## Example

```
RESOURCE_RESERVE=MAX_RESERVE_TIME[ 5]
```

This example specifies that jobs have up to 5 dispatch turns to reserve sufficient job slots or memory (equal to 5 minutes, by default).

## Default

Not defined. No job slots or memory is reserved.

## RES\_REQ

### Syntax

```
RES_REQ=res_req
```

## Description

Resource requirements used to determine eligible hosts. Specify a resource requirement string as usual. The resource requirement string lets you specify conditions in a more flexible manner than using the load thresholds. Resource requirement strings can be simple (applying to the entire job) or compound (applying to the specified number of slots).

When a compound resource requirement is set for a queue, it will be ignored unless it is the only resource requirement specified (no resource requirements are set at the job-level or application-level).

When a simple resource requirement is set for a queue and a compound resource requirement is set at the job-level or application-level, the queue-level requirements merge as they do for simple resource requirements. However, any job-based resources defined in the queue only apply to the first term of the merged compound resource requirements.

When `LSF_STRICT_RESREQ=Y` is configured in `lsf.conf`, resource requirement strings in select sections must conform to a more strict syntax. The strict resource requirement syntax only applies to the `select` section. It does not apply to the other resource requirement sections (`order`, `rusage`, `same`, `span`, or `cu`). When `LSF_STRICT_RESREQ=Y` in `lsf.conf`, LSF rejects resource requirement strings where an `rusage` section contains a non-consumable resource.

For simple resource requirements, the `select` sections from all levels must be satisfied and the same sections from all levels are combined. `cu`, `order`, and `span` sections at the job-level overwrite those at the application-level which overwrite those at the queue-level. Multiple `rusage` definitions are merged, with the job-level `rusage` taking precedence over the application-level, and application-level taking precedence over the queue-level.

The simple resource requirement `rusage` section can specify additional requests. To do this, use the `OR (| |)` operator to separate additional `rusage` strings. Multiple `-R` options cannot be used with multi-phase `rusage` resource requirements.

---

### Note:

Compound resource requirements do not support use of the `| |` operator within `rusage` sections, multiple `-R` options, or the `cu` section.

The `RES_REQ` consumable resource requirements must satisfy any limits set by the parameter `RESRSV_LIMIT` in `lsb.queues`, or the `RES_REQ` will be ignored.

When both the `RES_REQ` and `RESRSV_LIMIT` are set in `lsb.queues` for a consumable resource, the queue-level `RES_REQ` no longer acts as a hard limit for the merged `RES_REQ` `rusage` values from the job and application levels. In this case only the limits set by `RESRSV_LIMIT` must be satisfied, and the queue-level `RES_REQ` acts as a default value.

---

For example:

Queue-level RES_REQ:	<pre>RES_REQ=rusage[mem=200:lic=1] ...</pre>
	<p>For the job submission:</p> <pre>bsub -R' rusage[mem=100]' ...</pre>
	<p>the resulting requirement for the job is</p> <pre>rusage[mem=100:lic=1]</pre>
	<p>where <code>mem=100</code> specified by the job overrides <code>mem=200</code> specified by the queue. However, <code>lic=1</code> from queue is kept, since job does not specify it.</p>
Queue-level RES_REQ with decay and duration defined:	<pre>RES_REQ=rusage[mem=200:duration=20:decay=1] ...</pre>
	<p>For a job submission with no decay or duration:</p> <pre>bsub -R' rusage[mem=100]' ...</pre>
	<p>the resulting requirement for the job is:</p> <pre>rusage[mem=100:duration=20:decay=1]</pre>
	<p>Queue-level duration and decay are merged with the job-level specification, and <code>mem=100</code> for the job overrides <code>mem=200</code> specified by the queue. However, <code>duration=20</code> and <code>decay=1</code> from queue are kept, since job does not specify them.</p>
Queue-level RES_REQ with multi-phase job-level rusage:	<pre>RES_REQ=rusage[mem=200:duration=20:decay=1] ...</pre>
	<p>For a job submission with no decay or duration:</p> <pre>bsub -R' rusage[mem=(300 200 100):duration=(10 10 10)]' ...</pre>
	<p>the resulting requirement for the job is:</p> <pre>rusage[mem=(300 200 100):duration=(10 10 10)]</pre>

Multi-phase rusage values in the job submission override the single phase specified by the queue.

- If RESRSV\_LIMIT is defined in l sb. queues and has a maximum memory limit of 300 MB or greater, this job will be accepted.
- If RESRSV\_LIMIT is defined in l sb. queues and has a maximum memory limit of less than 300 MB, this job will be rejected.
- If RESRSV\_LIMIT is not defined in l sb. queues and the queue-level RES\_REQ value of 200 MB acts as a ceiling, this job will be rejected.

Queue-level  
multi-phase  
rusage  
RES\_REQ:

```
RES_REQ=rusage[mem=(350 200):duration=(20):decay=(1)] ...
```

For a single phase job submission with no decay or duration:

```
bsub -q q_name -R' rusage[mem=100: swap=150]' ...
```

the resulting requirement for the job is:

```
rusage[mem=100: swap=150]
```

The job-level rusage string overrides the queue-level multi-phase rusage string.

The order section defined at the job level overwrites any resource requirements specified at the application level or queue level. The order section defined at the application level overwrites any resource requirements specified at the queue level. The default order string is r15s:pg.

If RES\_REQ is defined at the queue level and there are no load thresholds defined, the pending reasons for each individual load index are not displayed by bj obs.

The span section defined at the queue level is ignored if the span section is also defined at the job level or in an application profile.

---

#### Note:

Define span[hosts=-1] in the application profile or bsub -R resource requirement string to override the span section setting in the queue.

---

Resource requirements determined by the queue no longer apply to a running job after running badmi n reconfi g. For example, if you change the RES\_REQ parameter in a queue and reconfigure the cluster, the previous queue-level resource requirements for running jobs are lost.

## Default

select[type==local] order[r15s:pg]. If this parameter is defined and a host model or Boolean resource is specified, the default type is any.

## RESRSV\_LIMIT

### Syntax

```
RESRSV_LIMIT=[res1={min1,} max1] [res2={min2,} max2]...
```

Where *res* is a consumable resource name, *min* is an optional minimum value and *max* is the maximum allowed value. Both *max* and *min* must be float numbers between 0 and 2147483647, and *min* cannot be greater than *max*.

### Description

Sets a range of allowed values for RES\_REQ resources.

Queue-level RES\_REQ rusage values (set in lsb. queues) must be in the range set by RESRSV\_LIMIT, or the queue-level RES\_REQ is ignored. Merged RES\_REQ rusage values from the job and application levels must be in the range of RESRSV\_LIMIT, or the job is rejected.

Changes made to the rusage values of running jobs using bmod -R cannot exceed the maximum values of RESRSV\_LIMIT, but can be lower than the minimum values.

When both the RES\_REQ and RESRSV\_LIMIT are set in lsb. queues for a consumable resource, the queue-level RES\_REQ no longer acts as a hard limit for the merged RES\_REQ rusage values from the job and application levels. In this case only the limits set by RESRSV\_LIMIT must be satisfied, and the queue-level RES\_REQ acts as a default value.

For MultiCluster, jobs must satisfy the RESRSV\_LIMIT range set for the send-jobs queue in the submission cluster. After the job is forwarded the resource requirements are also checked against the RESRSV\_LIMIT range set for the receive-jobs queue in the execution cluster.

---

**Note:**

Only consumable resource limits can be set in RESRSV\_LIMIT. Other resources will be ignored.

---

## Default

Not defined.

If *max* is defined and optional *min* is not, the default for *min* is 0.

## RESUME\_COND

### Syntax

**RESUME\_COND**=*res\_req*

Use the select section of the resource requirement string to specify load thresholds. All other sections are ignored.

### Description

LSF automatically resumes a suspended (SSUSP) job in this queue if the load on the host satisfies the specified conditions.

If RESUME\_COND is not defined, then the loadSched thresholds are used to control resuming of jobs. The loadSched thresholds are ignored, when resuming jobs, if RESUME\_COND is defined.

### Default

Not defined. The loadSched thresholds are used to control resuming of jobs.

## RUN\_JOB\_FACTOR

### Syntax

**RUN\_JOB\_FACTOR**=*number*

### Description

Used only with fairshare scheduling. Job slots weighting factor.

In the calculation of a user's dynamic share priority, this factor determines the relative importance of the number of job slots reserved and in use by a user.

If undefined, the cluster-wide value from the `lsb.params` parameter of the same name is used.

## Default

Not defined.

# RUN\_TIME\_DECAY

## Syntax

**RUN\_TIME\_DECAY=Y | y | N | n**

## Description

Used only with fairshare scheduling. Enables decay for run time at the same rate as the decay set by `HIST_HOURS` for cumulative CPU time and historical run time.

In the calculation of a user's dynamic share priority, this factor determines whether run time is decayed.

If undefined, the cluster-wide value from the `lsb.params` parameter of the same name is used.

## Restrictions

Running `badadmin reconfig` or restarting `mbatchd` during a job's run time results in the decayed run time being recalculated.

When a suspended job using run time decay is resumed, the decay time is based on the elapsed time.

## Default

Not defined

# RUN\_TIME\_FACTOR

## Syntax

**RUN\_TIME\_FACTOR=*number***

## Description

Used only with fairshare scheduling. Run time weighting factor.

In the calculation of a user's dynamic share priority, this factor determines the relative importance of the total run time of a user's running jobs.

If undefined, the cluster-wide value from the `lsb.params` parameter of the same name is used.

## Default

Not defined.

# RUN\_WINDOW

## Syntax

**RUN\_WINDOW**=*time\_window* ...

## Description

Time periods during which jobs in the queue are allowed to run.

When the window closes, LSF suspends jobs running in the queue and stops dispatching jobs from the queue. When the window reopens, LSF resumes the suspended jobs and begins dispatching additional jobs.

## Default

Not defined. Queue is always active.

# RUNLIMIT

## Syntax

**RUNLIMIT**=[*default\_limit*] *maximum\_limit*

where *default\_limit* and *maximum\_limit* are:

[*hour*:]*minute*[/*host\_name* | /*host\_model*]

## Description

The maximum run limit and optionally the default run limit. The name of a host or host model specifies the runtime normalization host to use.

By default, jobs that are in the RUN state for longer than the specified maximum run limit are killed by LSF. You can optionally provide your own termination job action to override this default.

Jobs submitted with a job-level run limit (bsub -W) that is less than the maximum run limit are killed when their job-level run limit is reached. Jobs submitted with a run limit greater than the maximum run limit are rejected by the queue.

If a default run limit is specified, jobs submitted to the queue without a job-level run limit are killed when the default run limit is reached. The default run limit is used with backfill scheduling of parallel jobs.

---

### Note:

If you want to provide an estimated run time for scheduling purposes without killing jobs that exceed the estimate, define the RUNTIME parameter in an application profile instead of a run limit (see lsb. applications for details).

---

If you specify only one limit, it is the maximum, or hard, run limit. If you specify two limits, the first one is the default, or soft, run limit, and the second one is the maximum run limit. The number of minutes may be greater than 59. Therefore, three and a half hours can be specified either as 3:30, or 210.

The run limit is 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 run limit you specify is the normalized run time. This is done so that the job does approximately the same amount of processing, even if it is sent to host with a faster or slower CPU. Whenever a normalized run time is given, the actual time on the execution host is the specified time multiplied by the CPU factor of the normalization host then divided by the CPU factor of the execution host.

If `ABS_RUNLIMIT=Y` is defined in `lsb.params`, the runtime limit is not normalized by the host CPU factor. Absolute wall-clock run time is used for all jobs submitted to a queue with a run limit configured.

Optionally, you can supply a host name or a host model name defined in LSF. You must insert `'/'` between the run limit and the host name or model name. (See `lsinfo(1)` to get host model information.)

If no host or host model is given, LSF uses the default runtime normalization host defined at the queue level (`DEFAULT_HOST_SPEC` in `lsb.queues`) if it has been configured; otherwise, LSF uses the default CPU time normalization host defined at the cluster level (`DEFAULT_HOST_SPEC` in `lsb.params`) if it has been configured; otherwise, the host with the largest CPU factor (the fastest host in the cluster).

For MultiCluster jobs, if no other CPU time normalization host is defined and information about the submission host is not available, LSF uses the host with the largest CPU factor (the fastest host in the cluster).

Jobs submitted to a chunk job queue are not chunked if `RUNLIMIT` is greater than 30 minutes.

`RUNLIMIT` is required for queues configured with `INTERRUPTIBLE_BACKFILL`.

## Default

Unlimited

# SLA\_GUARANTEES\_IGNORE

## Syntax

`SLA_GUARANTEES_IGNORE=Y|y|N|n`

## Description

Applies to SLA guarantees only.

`SLA_GUARANTEES_IGNORE=Y` allows jobs in the queue access to all guaranteed resources. As a result, some guarantees might not be honored.

---

### Note:

Using `SLA_GUARANTEES_IGNORE=Y` defeats the purpose of guaranteeing resources. This should be used sparingly for low traffic queues only.

---

## Default

Not defined (N). The queue must honor resource guarantees when dispatching jobs.

# SLOT\_POOL

## Syntax

`SLOT_POOL=pool_name`

## Description

Name of the pool of job slots the queue belongs to for queue-based fairshare. A queue can only belong to one pool. All queues in the pool must share the same set of hosts.

## Valid values

Specify any ASCII string up to 60 characters long. You can use letters, digits, underscores (\_) or dashes (-). You cannot use blank spaces.

## Default

Not defined. No job slots are reserved.

# SLOT\_RESERVE

## Syntax

```
SLOT_RESERVE=MAX_RESERVE_TIME[integer]
```

## Description

Enables processor reservation for the queue and specifies the reservation time. Specify the keyword `MAX_RESERVE_TIME` and, in square brackets, the number of `MBD_SLEEP_TIME` cycles over which a job can reserve job slots. `MBD_SLEEP_TIME` is defined in `lsb.params`; the default value is 60 seconds.

If a job has not accumulated enough job slots to start before the reservation expires, it releases all its reserved job slots so that other jobs can run. Then, the job cannot reserve slots for one scheduling session, so other jobs have a chance to be dispatched. After one scheduling session, the job can reserve job slots again for another period specified by `SLOT_RESERVE`.

`SLOT_RESERVE` is overridden by the `RESOURCE_RESERVE` parameter.

If both `RESOURCE_RESERVE` and `SLOT_RESERVE` are defined in the same queue, job slot reservation and memory reservation are enabled and an error is displayed when the cluster is reconfigured. `SLOT_RESERVE` is ignored.

Job slot reservation for parallel jobs is enabled by `RESOURCE_RESERVE` if the LSF scheduler plugin module names for both resource reservation and parallel batch jobs (`schmod_parallel` and `schmod_reserve`) are configured in the `lsb.modules` file: The `schmod_parallel` name *must* come before `schmod_reserve` in `lsb.modules`.

If `BACKFILL` is configured in a queue, and a run limit is specified at the job level (`bsub -W`), application level (`RUNLIMIT` in `lsb.applications`), or queue level (`RUNLIMIT` in `lsb.queues`), or if an estimated run time is specified at the application level (`RUNTIME` in `lsb.applications`), backfill parallel jobs can use job slots reserved by the other jobs, as long as the backfill job can finish before the predicted start time of the jobs with the reservation.

Unlike memory reservation, which applies both to sequential and parallel jobs, slot reservation applies only to parallel jobs.

## Example

```
SLOT_RESERVE=MAX_RESERVE_TIME[5]
```

This example specifies that parallel jobs have up to 5 cycles of `MBD_SLEEP_TIME` (5 minutes, by default) to reserve sufficient job slots to start.

## Default

Not defined. No job slots are reserved.

# SLOT\_SHARE

## Syntax

**SLOT\_SHARE**=*integer*

## Description

Share of job slots for queue-based fairshare. Represents the percentage of running jobs (job slots) in use from the queue. SLOT\_SHARE must be greater than zero (0) and less than or equal to 100.

The sum of SLOT\_SHARE for all queues in the pool does not need to be 100%. It can be more or less, depending on your needs.

## Default

Not defined

# SNDJOBS\_TO

## Syntax

**SNDJOBS\_TO**=*queue\_name@cluster\_name ...*

## Description

Defines a MultiCluster send-jobs queue.

Specify remote queue names, in the form *queue\_name@cluster\_name*, separated by a space.

This parameter is ignored if lsb. queues HOSTS specifies remote (borrowed) resources.

## Example

```
SNDJOBS_TO=queue2@cluster2 queue3@cluster2 queue3@cluster3
```

# STACKLIMIT

## Syntax

**STACKLIMIT**=*integer*

## Description

The per-process (hard) stack segment size limit (in KB) for all of the processes belonging to a job from this queue (see `getrlimit(2)`).

## Default

Unlimited

## STOP\_COND

### Syntax

**STOP\_COND**=*res\_req*

Use the `select` section of the resource requirement string to specify load thresholds. All other sections are ignored.

### Description

LSF automatically suspends a running job in this queue if the load on the host satisfies the specified conditions.

- LSF does not suspend the only job running on the host if the machine is interactively idle (`it > 0`).
- LSF does not suspend a forced job (`brun -f`).
- LSF does not suspend a job because of paging rate if the machine is interactively idle.

If `STOP_COND` is specified in the queue and there are no load thresholds, the suspending reasons for each individual load index is not displayed by `bj obs`.

### Example

```
STOP_COND= select[( !cs && it < 5) || (cs && mem < 15 && swp < 50)]
```

In this example, assume “`cs`” is a Boolean resource indicating that the host is a computer server. The stop condition for jobs running on computer servers is based on the availability of swap memory. The stop condition for jobs running on other kinds of hosts is based on the idle time.

## SWAPLIMIT

### Syntax

**SWAPLIMIT**=*integer*

### Description

The amount of total virtual memory limit (in KB) for a job from this queue.

This limit applies to the whole job, no matter how many processes the job may contain.

The action taken when a job exceeds its `SWAPLIMIT` or `PROCESSLIMIT` is to send `SIGQUIT`, `SIGINT`, `SIGTERM`, and `SIGKILL` in sequence. For `CPULIMIT`, `SIGXCPU` is sent before `SIGINT`, `SIGTERM`, and `SIGKILL`.

### Default

Unlimited

## TERMINATE\_WHEN

### Syntax

**TERMINATE\_WHEN**=[LOAD] [PREEMPT] [WINDOW]

## Description

Configures the queue to invoke the TERMINATE action instead of the SUSPEND action in the specified circumstance.

- **LOAD**: kills jobs when the load exceeds the suspending thresholds.
- **PREEMPT**: kills jobs that are being preempted.
- **WINDOW**: kills jobs if the run window closes.

If the TERMINATE\_WHEN job control action is applied to a chunk job, sbatchd kills the chunk job element that is running and puts the rest of the waiting elements into pending state to be rescheduled later.

## Example

Set TERMINATE\_WHEN to WINDOW to define a night queue that kills jobs if the run window closes:

```
Begin Queue
NAME           = night
RUN_WINDOW    = 20:00-08:00
TERMINATE_WHEN = WINDOW
JOB_CONTROLS  = TERMINATE[kill -KILL $LS_JOBPGIDS; mail -s "job $LSB_JOBID killed by
queue run window" $USER < /dev/null]
End Queue
```

# THREADLIMIT

## Syntax

**THREADLIMIT**=[*default\_limit*] *maximum\_limit*

## Description

Limits the number of concurrent threads that can be part of a job. Exceeding the limit causes the job to terminate. The system sends the following signals in sequence to all processes belongs to the job: SIGINT, SIGTERM, and SIGKILL.

By default, if a default thread limit is specified, jobs submitted to the queue without a job-level thread limit are killed when the default thread limit is reached.

If you specify only one limit, it is the maximum, or hard, thread limit. If you specify two limits, the first one is the default, or soft, thread limit, and the second one is the maximum thread limit.

Both the default and the maximum limits must be positive integers. The default limit must be less than the maximum limit. The default limit is ignored if it is greater than the maximum limit.

## Examples

```
THREADLIMIT=6
```

No default thread limit is specified. The value 6 is the default and maximum thread limit.

```
THREADLIMIT=6 8
```

The first value (6) is the default thread limit. The second value (8) is the maximum thread limit.

## Default

Unlimited

## UJOB\_LIMIT

### Syntax

**UJOB\_LIMIT=integer**

### Description

Per-user job slot limit for the queue. Maximum number of job slots that each user can use in this queue.

UJOB\_LIMIT must be within or greater than the range set by PROCLIMIT or bsub -n (if either is used), or jobs are rejected.

### Default

Unlimited

## USE\_PAM\_CREDS

### Syntax

**USE\_PAM\_CREDS=y | n**

### Description

If USE\_PAM\_CREDS=y, applies PAM limits to a queue when its job is dispatched to a Linux host using PAM. PAM limits are system resource limits defined in `limits.conf`.

When USE\_PAM\_CREDS is enabled, PAM limits override others. For example, the PAM limit is used even if queue-level soft limit is less than PAM limit. However, it still cannot exceed queue's hard limit.

If the execution host does not have PAM configured and this parameter is enabled, the job fails.

For parallel jobs, only takes effect on the first execution host.

USE\_PAM\_CREDS only applies on the following platforms:

- linux2.6-glibc2.3-ia64
- linux2.6-glibc2.3-ppc64
- linux2.6-glibc2.3-sn-iph
- linux2.6-glibc2.3-x86
- linux2.6-glibc2.3-x86\_64

Overrides MEMLIMIT\_TYPE=Process.

Overridden (for CPU limit only) by LSB\_JOB\_CPULIMIT=y.

Overridden (for memory limits only) by LSB\_JOB\_MEMLIMIT=y.

### Default

n

## USE\_PRIORITY\_IN\_POOL

### Syntax

**USE\_PRIORITY\_IN\_POOL= y | Y | n | N**

## Description

Queue-based fairshare only. After job scheduling occurs for each queue, this parameter enables LSF to dispatch jobs to any remaining slots in the pool in first-come first-served order across queues.

## Default

N

## USERS

## Syntax

```
USERS=all [~user_name ...] [~user_group ...] | [user_name ...] [user_group [~user_group ...] ...]
```

## Description

A space-separated list of user names or user groups that can submit jobs to the queue. LSF cluster administrators are automatically included in the list of users. LSF cluster administrators can submit jobs to this queue, or switch (b`swi t ch`) any user's jobs into this queue.

If user groups are specified, each user in the group can submit jobs to this queue. If FAIRSHARE is also defined in this queue, only users defined by both parameters can submit jobs, so LSF administrators cannot use the queue if they are not included in the share assignments.

User names must be valid login names. To specify a Windows user account, include the domain name in uppercase letters (*DOMAIN\_NAME\user\_name*).

User group names can be LSF user groups or UNIX and Windows user groups. To specify a Windows user group, include the domain name in uppercase letters (*DOMAIN\_NAME\user\_group*).

Use the keyword `all` to specify all users or user groups in a cluster.

Use the not operator (`~`) to exclude users from the `all` specification or from user groups. This is useful if you have a large number of users but only want to exclude a few users or groups from the queue definition.

The not operator (`~`) can only be used with the `all` keyword or to exclude users from user groups.

---

### Caution:

The not operator does not exclude LSF administrators from the queue definition.

---

## Default

`all` (all users can submit jobs to the queue)

## Examples

- `USERS=user1 user2`
- `USERS=all ~user1 ~user2`
- `USERS=all ~ugroup1`
- `USERS=groupA ~user3 ~user4`

## Automatic time-based configuration

Variable configuration is used to automatically change LSF configuration based on time windows. You define automatic configuration changes in `lsb.queues` by using if-else constructs and time expressions. After you change the files, reconfigure the cluster with the `badm n reconf i g` command.

The expressions are evaluated by LSF every 10 minutes based on `mbatchd` start time. When an expression evaluates true, LSF dynamically changes the configuration based on the associated configuration statements. Reconfiguration is done in real time without restarting `mbatchd`, providing continuous system availability.

### Example

```
Begin Queue
...
#i f t i m e ( 8 : 3 0 - 1 8 : 3 0 )
INTERACTIVE = ONLY # i n t e r a c t i v e o n l y d u r i n g d a y s h i f t # e n d i f
...
End Queue
```

# lsb.resources

The `lsb.resources` file contains configuration information for resource allocation limits, exports, and resource usage limits. This file is optional.

The `lsb.resources` file is stored in the directory `LSB_CONFDIR/cluster_name/confdir`, where `LSB_CONFDIR` is defined in `lsf.conf`.

## Changing lsb.resources configuration

After making any changes to `lsb.resources`, run `badm n reconfi g` to reconfigure `mbatchd`.

## Limit section

The Limit section sets limits for the maximum amount of the specified resources that must be available for different classes of jobs to start, and which resource consumers the limits apply to. Limits are enforced during job resource allocation.

---

### Tip:

For limits to be enforced, jobs must specify `rusage` resource requirements (`bsub -R` or `RES_REQ` in `lsb.queues`).

---

The `blimits` command displays view current usage of resource allocation limits configured in Limit sections in `lsb.resources`:

## Limit section structure

Each set of limits is defined in a Limit section enclosed by `Begin Limit` and `End Limit`.

A Limit section has two formats:

- Vertical tabular
- Horizontal

The file can contain sections in both formats. In either format, you must configure a limit for at least one consumer and one resource. The Limit section cannot be empty.

## Vertical tabular format

Use the vertical format for simple configuration conditions involving only a few consumers and resource limits.

The first row consists of an optional NAME and the following keywords for:

- Resource types:
  - SLOTS or SLOTS\_PER\_PROCESSOR
  - MEM (MB or percentage)
  - SWP (MB or percentage)
  - TMP (MB or percentage)
  - LICENSE
  - JOBS
  - RESOURCE
- Consumer types:

- USERS or PER\_USER
- QUEUES or PER\_QUEUE
- HOSTS or PER\_HOST
- PROJECTS or PER\_PROJECT
- LIC\_PROJECTS or PER\_LIC\_PROJECT

Each subsequent row describes the configuration information for resource consumers and the limits that apply to them. Each line must contain an entry for each keyword. Use empty parentheses () or a dash (-) to indicate an empty field. Fields cannot be left blank.

---

**Tip:**

Multiple entries must be enclosed in parentheses. For RESOURCE and LICENSE limits, resource and license names must be enclosed in parentheses.

---

## Horizontal format

Use the horizontal format to give a name for your limits and to configure more complicated combinations of consumers and resource limits.

The first line of the Limit section gives the name of the limit configuration.

Each subsequent line in the Limit section consists of keywords identifying the resource limits:

- Job slots and per-processor job slots
- Memory (MB or percentage)
- Swap space (MB or percentage)
- Tmp space (MB or percentage)
- Software licenses
- Running and suspended (RUN, SSUSP, USUSP) jobs
- Other shared resources

and the resource *consumers* to which the limits apply:

- Users and user groups
- Hosts and host groups
- Queues
- Projects
- License Projects

## Example: Vertical tabular format

In the following limit configuration:

- Jobs from user1 and user3 are limited to 2 job slots on host A
- Jobs from user2 on queue normal are limited to 20 MB of memory

- The short queue can have at most 200 running and suspended jobs

```

Begin Limit
NAME      USERS          QUEUES  HOSTS    SLOTS  MEM  SWP  TMP  JOBS
limit1   (user1 user3)  -       hostA    2      -   -   -   -
-        user2          normal  -        -      20  -   -   -
-        -             short   -        -      -   -   -   200
End Limit

```

Jobs that do not match these limits; that is, all users except user 1 and user 3 running jobs on host A and all users except user 2 submitting jobs to queue normal, have no limits.

## Example: Horizontal format

All users in user group ugroup1 except user 1 using queue1 and queue2 and running jobs on hosts in host group hgroup1 are limited to 2 job slots per processor on each host:

```

Begin Limit
# ugroup1 except user1 uses queue1 and queue2 with 2 job slots
# on each host in hgroup1
NAME          = limit1
# Resources
SLOTS_PER_PROCESSOR = 2
#Consumers
QUEUES        = queue1 queue2
USERS         = ugroup1 ~user1
PER_HOST      = hgroup1
End Limit

```

## Compatibility with lsb.queues, lsb.users, and lsb.hosts

The Limit section of lsb.resources does not support the keywords or format used in lsb.users, lsb.hosts, and lsb.queues. However, your existing job slot limit configuration in these files will continue to apply.

Job slot limits are the only type of limit you can configure in lsb.users, lsb.hosts, and lsb.queues. You cannot configure limits for user groups, host groups, license projects, and projects in lsb.users, lsb.hosts, and lsb.queues. You should not configure any new resource allocation limits in lsb.users, lsb.hosts, and lsb.queues. Use lsb.resources to configure all new resource allocation limits, including job slot limits. Limits on running and suspended jobs can only be set in lsb.resources.

Existing limits in lsb.users, lsb.hosts, and lsb.queues with the same scope as a new limit in lsb.resources, but with a different value are ignored. The value of the new limit in lsb.resources is used. Similar limits with different scope enforce the most restrictive limit.

## Parameters

- HOSTS
- JOBS
- LICENSE
- LIC\_PROJECTS

- MEM
- NAME
- PER\_HOST
- PER\_LIC\_PROJECT
- PER\_PROJECT
- PER\_QUEUE
- PER\_USER
- PROJECTS
- QUEUES
- RESOURCE
- SLOTS
- SLOTS\_PER\_PROCESSOR
- SWP
- TMP
- USERS

## HOSTS

### Syntax

**HOSTS=**all [-]host\_name ... | all [-]host\_group ...

**HOSTS**

( [-] | all [-]host\_name ... | all [-]host\_group ... )

### Description

A space-separated list of hosts, host groups defined in lsb. hosts on which limits are enforced. Limits are enforced on all hosts or host groups listed.

If a group contains a subgroup, the limit also applies to each member in the subgroup recursively.

To specify a per-host limit, use the PER\_HOST keyword. Do not configure HOSTS and PER\_HOST limits in the same Limit section.

If you specify MEM, TMP, or SWP as a percentage, you must specify PER\_HOST and list the hosts that the limit is to be enforced on. You cannot specify HOSTS.

In horizontal format, use only one HOSTS line per Limit section.

Use the keyword all to configure limits that apply to all hosts in a cluster.

Use the not operator (-) to exclude hosts from the all specification in the limit. This is useful if you have a large cluster but only want to exclude a few hosts from the limit definition.

In vertical tabular format, multiple host names must be enclosed in parentheses.

In vertical tabular format, use empty parentheses () or a dash (-) to indicate an empty field. Fields cannot be left blank.

### Default

all (limits are enforced on all hosts in the cluster).

### Example 1

```
HOSTS=Group1 ~hostA hostB hostC
```

Enforces limits on host B, host C, and all hosts in Group1 except for host A.

## Example 2

```
HOSTS=all ~group2 ~hostA
```

Enforces limits on all hosts in the cluster, except for host A and the hosts in group2.

## Example 3

```
HOSTS SWP (all ~hostK ~hostM) 10
```

Enforces a 10 MB swap limit on all hosts in the cluster, except for host K and host M

# JOBS

## Syntax

**JOBS**=*integer*

**JOBS**

- | *integer*

## Description

Maximum number of running or suspended (RUN, SSUSP, USUSP) jobs available to resource consumers. Specify a positive integer greater than or equal 0. Job limits can be defined in both vertical and horizontal limit formats.

With MultiCluster resource lease model, this limit applies only to local hosts being used by the local cluster. The job limit for hosts exported to a remote cluster is determined by the host export policy, not by this parameter. The job limit for borrowed hosts is determined by the host export policy of the remote cluster.

If SLOTS are configured in the Limit section, the most restrictive limit is applied.

If HOSTS are configured in the Limit section, JOBS is the number of running and suspended jobs on a host. If preemptive scheduling is used, the suspended jobs are not counted against the job limit.

Use this parameter to prevent a host from being overloaded with too many jobs, and to maximize the throughput of a machine.

If only QUEUES are configured in the Limit section, JOBS is the maximum number of jobs that can run in the listed queues.

If only USERS are configured in the Limit section, JOBS is the maximum number of jobs that the users or user groups can run.

If only HOSTS are configured in the Limit section, JOBS is the maximum number of jobs that can run on the listed hosts.

If only PROJECTS are configured in the Limit section, JOBS is the maximum number of jobs that can run under the listed projects.

If only LIC\_PROJECTS are configured in the Limit section, JOBS is the maximum number of jobs that can run under the listed license projects.

Use QUEUES or PER\_QUEUE, USERS or PER\_USER, HOSTS or PER\_HOST, LIC\_PROJECTS or PER\_LIC\_PROJECT, and PROJECTS or PER\_PROJECT in combination to further limit jobs available to resource consumers.

In horizontal format, use only one JOBS line per Limit section.

In vertical format, use empty parentheses () or a dash (-) to indicate the default value (no limit). Fields cannot be left blank.

## Default

No limit

## Example

```
JOBS=20
```

# LICENSE

## Syntax

```
LICENSE=[license_name,integer] [[license_name,integer] ...]
```

```
LICENSE
```

```
( [license_name,integer] [[license_name,integer] ...] )
```

## Description

Maximum number of specified software licenses available to resource consumers. The value must be a positive integer greater than or equal to zero.

Software licenses must be defined as decreasing numeric shared resources in lsf. shared.

You cannot specify RESOURCE and LICENSE in the same Limit section.

In horizontal format, use only one LICENSE line per Limit section.

In vertical tabular format, license entries must be enclosed in parentheses.

In vertical tabular format, use empty parentheses () or a dash (-) to indicate the default value (no limit). Fields cannot be left blank.

## Default

None

## Examples

```
LICENSE=[verilog, 4] [spice, 2]
Begin Limit
LICENSE                PER_HOST
([verilog, 1])         (all ~hostA)
([verilog, 1] [spice, 2]) (hostA)
End Limit
```

# LIC\_PROJECTS

## Syntax

```
LIC_PROJECTS=all [~]lic_project_name ...
```

```
LIC_PROJECTS
```

( [-] | all [-]lic\_project\_name ... )

## Description

A space-separated list of license project names on which limits are enforced. Limits are enforced on all license projects listed.

To specify a per-project limit on license projects, use the PER\_LIC\_PROJECT keyword. Do not configure LIC\_PROJECTS and PER\_LIC\_PROJECT limits in the same Limit section.

In horizontal format, use only one LIC\_PROJECTS line per Limit section.

Use the keyword all to configure limits that apply to all license projects in a cluster.

Use the not operator (-) to exclude license projects from the all specification in the limit. This is useful if you have a large number of license projects but only want to exclude a few license projects from the limit definition.

In vertical tabular format, multiple license project names must be enclosed in parentheses.

In vertical tabular format, use empty parentheses () or a dash (-) to indicate an empty field. Fields cannot be left blank.

LIC\_PROJECTS limits do not apply to taskman jobs.

---

### Note:

Jobs submitted without -Lp and without license requirements are not associated with a default license project. The keyword all is not the same as dash (-) for LIC\_PROJECTS.

---

## Default

None. If no limit is specified for LIC\_PROJECTS or PER\_LIC\_PROJECT, no limit is enforced on any license project.

## Example

```
LIC_PROJECTS=proj A proj B
```

## MEM

## Syntax

**MEM**=integer[%]

**MEM**

- | integer[%]

## Description

Maximum amount of memory available to resource consumers. Specify a value in MB or a percentage (%) as a positive integer greater than or equal 0. If you specify a percentage, you must also specify PER\_HOST and list the hosts that the limit is to be enforced on.

The Limit section is ignored if MEM is specified as a percentage:

- Without PER\_HOST, or
- With HOSTS

In horizontal format, use only one MEM line per Limit section.

In vertical tabular format, use empty parentheses () or a dash (-) to indicate the default value (no limit). Fields cannot be left blank.

If only QUEUES are configured in the Limit section, MEM must be an integer value. MEM is the maximum amount of memory available to the listed queues.

If only USERS are configured in the Limit section, MEM must be an integer value. MEM is the maximum amount of memory that the users or user groups can use.

If only HOSTS are configured in the Limit section, MEM must be an integer value. It cannot be a percentage. MEM is the maximum amount of memory available to the listed hosts.

If only PROJECTS are configured in the Limit section, MEM must be an integer value. MEM is the maximum amount of memory available to the listed projects.

If only LIC\_PROJECTS are configured in the Limit section, MEM must be an integer value. MEM is the maximum amount of memory available to the listed license projects.

Use QUEUES or PER\_QUEUE, USERS or PER\_USER, HOSTS or PER\_HOST, LIC\_PROJECTS or PER\_LIC\_PROJECT, and PROJECTS or PER\_PROJECT in combination to further limit memory available to resource consumers.

## Default

No limit

## Example

```
MEM=20
```

## NAME

### Syntax

**NAME**=*limit\_name*

**NAME**

- | *limit\_name*

## Description

Name of the Limit section

Specify any ASCII string 40 characters or less. You can use letters, digits, underscores (\_) or dashes (-). You cannot use blank spaces.

If duplicate limit names are defined, the Limit section is ignored. If value of NAME is not defined in vertical format, or defined as (-), bl i mt i s displays *NONAME.nnn*.

## Default

None. In horizontal format, you must provide a name for the Limit section. NAME is optional in the vertical format.

## Example

```
NAME=short_l i m i t s
```

## PER\_HOST

### Syntax

```
PER_HOST=all [-]host_name ... | all [-]host_group ...
```

```
PER_HOST
```

```
( [-] | all [-]host_name ... | all [-]host_group ... )
```

### Description

A space-separated list of host or host groups defined in `lsb.hosts` on which limits are enforced. Limits are enforced on each host or individually to each host of the host group listed. If a group contains a subgroup, the limit also applies to each member in the subgroup recursively.

Do not configure `PER_HOST` and `HOSTS` limits in the same Limit section.

In horizontal format, use only one `PER_HOST` line per Limit section.

If you specify `MEM`, `TMP`, or `SWP` as a percentage, you must specify `PER_HOST` and list the hosts that the limit is to be enforced on. You cannot specify `HOSTS`.

Use the keyword `all` to configure limits that apply to each host in a cluster. If host groups are configured, the limit applies to each member of the host group, not the group as a whole.

Use the not operator (`~`) to exclude hosts or host groups from the `all` specification in the limit. This is useful if you have a large cluster but only want to exclude a few hosts from the limit definition.

In vertical tabular format, multiple host names must be enclosed in parentheses.

In vertical tabular format, use empty parentheses `()` or a dash `(-)` to indicate an empty field. Fields cannot be left blank.

### Default

None. If no limit is specified for `PER_HOST` or `HOSTS`, no limit is enforced on any host or host group.

### Example

```
PER_HOST=hostA hgroup1 ~hostC
```

## PER\_LIC\_PROJECT

### Syntax

```
PER_LIC_PROJECT=all [-]lic_project_name ...
```

```
PER_LIC_PROJECT
```

```
( [-] | all [-]lic_project_name ... )
```

### Description

A space-separated list of license project names on which limits are enforced. Limits are enforced on each license project listed.

Do not configure `PER_LIC_PROJECT` and `LIC_PROJECTS` limits in the same Limit section.

In horizontal format, use only one `PER_LIC_PROJECT` line per Limit section.

Use the keyword `all` to configure limits that apply to each license project in a cluster.

Use the not operator (`~`) to exclude license projects from the `all` specification in the limit.

In vertical tabular format, multiple license project names must be enclosed in parentheses.

In vertical tabular format, use empty parentheses `()` or a dash `(-)` to indicate an empty field. Fields cannot be left blank.

`PER_LIC_PROJECT` limits do not apply to `taskman` jobs.

## Default

None. If no limit is specified for `PER_LIC_PROJECT` or `LIC_PROJECTS`, no limit is enforced on any license project.

## Example

```
PER_LIC_PROJECT=proj 1 proj 2
```

## PER\_PROJECT

### Syntax

```
PER_PROJECT=all [-]project_name ...
```

```
PER_PROJECT
```

```
( [-] | all [-]project_name ... )
```

### Description

A space-separated list of project names on which limits are enforced. Limits are enforced on each project listed.

Do not configure `PER_PROJECT` and `PROJECTS` limits in the same Limit section.

In horizontal format, use only one `PER_PROJECT` line per Limit section.

Use the keyword `all` to configure limits that apply to each project in a cluster.

Use the not operator (`~`) to exclude projects from the `all` specification in the limit.

In vertical tabular format, multiple project names must be enclosed in parentheses.

In vertical tabular format, use empty parentheses `()` or a dash `(-)` to indicate an empty field. Fields cannot be left blank.

## Default

None. If no limit is specified for `PER_PROJECT` or `PROJECTS`, no limit is enforced on any project.

## Example

```
PER_PROJECT=proj 1 proj 2
```

## PER\_QUEUE

### Syntax

```
PER_QUEUE=all [-]queue_name ..
```

**PER\_QUEUE**

( [-] | **all** [-] *queue\_name* ... )

## Description

A space-separated list of queue names on which limits are enforced. Limits are enforced on jobs submitted to each queue listed.

Do not configure PER\_QUEUE and QUEUES limits in the same Limit section.

In horizontal format, use only one PER\_QUEUE line per Limit section.

Use the keyword **all** to configure limits that apply to each queue in a cluster.

Use the not operator (~) to exclude queues from the **all** specification in the limit. This is useful if you have a large number of queues but only want to exclude a few queues from the limit definition.

In vertical tabular format, multiple queue names must be enclosed in parentheses.

In vertical tabular format, use empty parentheses () or a dash (-) to indicate an empty field. Fields cannot be left blank.

## Default

None. If no limit is specified for PER\_QUEUE or QUEUES, no limit is enforced on any queue.

## Example

```
PER_QUEUE=pri or i t y ni ght
```

## PER\_USER

### Syntax

**PER\_USER**=**all** [-] *user\_name* ... | **all** [-] *user\_group* ...

**PER\_USER**

( [-] | **all** [-] *user\_name* ... | **all** [-] *user\_group* ... )

## Description

A space-separated list of user names or user groups on which limits are enforced. Limits are enforced on each user or individually to each user in the user group listed. If a user group contains a subgroup, the limit also applies to each member in the subgroup recursively.

User names must be valid login names. User group names can be LSF user groups or UNIX and Windows user groups. Note that for LSF and UNIX user groups, the groups must be specified in a UserGroup section in `lsb.users` first.

Do not configure PER\_USER and USERS limits in the same Limit section.

In horizontal format, use only one PER\_USER line per Limit section.

Use the keyword **all** to configure limits that apply to each user in a cluster. If user groups are configured, the limit applies to each member of the user group, not the group as a whole.

Use the not operator (~) to exclude users or user groups from the **all** specification in the limit. This is useful if you have a large number of users but only want to exclude a few users from the limit definition.

In vertical tabular format, multiple user names must be enclosed in parentheses.

In vertical tabular format, use empty parentheses () or a dash (-) to indicate an empty field. Fields cannot be left blank.

## Default

None. If no limit is specified for PER\_USER or USERS, no limit is enforced on any user or user group.

## Example

```
PER_USER=user1 user2 ugroup1 ~user3
```

# PROJECTS

## Syntax

**PROJECTS=all** [-]project\_name ...

**PROJECTS**

( [-] | **all** [-]project\_name ... )

## Description

A space-separated list of project names on which limits are enforced. Limits are enforced on all projects listed.

To specify a per-project limit, use the PER\_PROJECT keyword. Do not configure PROJECTS and PER\_PROJECT limits in the same Limit section.

In horizontal format, use only one PROJECTS line per Limit section.

Use the keyword `all` to configure limits that apply to all projects in a cluster.

Use the not operator (~) to exclude projects from the `all` specification in the limit. This is useful if you have a large number of projects but only want to exclude a few projects from the limit definition.

In vertical tabular format, multiple project names must be enclosed in parentheses.

In vertical tabular format, use empty parentheses () or a dash (-) to indicate an empty field. Fields cannot be left blank.

## Default

`all` (limits are enforced on all projects in the cluster)

## Example

```
PROJECTS=proj A proj B
```

# QUEUES

## Syntax

**QUEUES=all** [-]queue\_name ...

**QUEUES**

( [-] | **all** [-]queue\_name ... )

## Description

A space-separated list of queue names on which limits are enforced. Limits are enforced on all queues listed.

The list must contain valid queue names defined in `lsb.queues`.

To specify a per-queue limit, use the `PER_QUEUE` keyword. Do not configure `QUEUES` and `PER_QUEUE` limits in the same Limit section.

In horizontal format, use only one `QUEUES` line per Limit section.

Use the keyword `all` to configure limits that apply to all queues in a cluster.

Use the not operator (`~`) to exclude queues from the `all` specification in the limit. This is useful if you have a large number of queues but only want to exclude a few queues from the limit definition.

In vertical tabular format, multiple queue names must be enclosed in parentheses.

In vertical tabular format, use empty parentheses `()` or a dash `(-)` to indicate an empty field. Fields cannot be left blank.

## Default

`all` (limits are enforced on all queues in the cluster)

## Example

```
QUEUES=normal night
```

# RESOURCE

## Syntax

**RESOURCE**=[*shared\_resource,integer*] [[*shared\_resource,integer*] ...]

**RESOURCE**

( [[*shared\_resource,integer*] [[*shared\_resource,integer*] ...] )

## Description

Maximum amount of any user-defined shared resource available to consumers.

You cannot specify `RESOURCE` and `LICENSE` in the same Limit section; you can use `RESOURCE` to configure software licenses.

In horizontal format, use only one `RESOURCE` line per Limit section.

In vertical tabular format, resource names must be enclosed in parentheses.

In vertical tabular format, use empty parentheses `()` or a dash `(-)` to indicate an empty field. Fields cannot be left blank.

## Default

None

## Examples

```
RESOURCE=[stat_shared, 4]
Begin Limit
RESOURCE                PER_HOST
([stat_shared, 4])      (all ~hostA)
([dyn_rsrc, 1] [stat_rsrc, 2]) (hostA)
End Limit
```

## SLOTS

### Syntax

**SLOTS**=*integer*

**SLOTS**

- | *integer*

### Description

Maximum number of job slots available to resource consumers. Specify a positive integer greater than or equal 0.

With MultiCluster resource lease model, this limit applies only to local hosts being used by the local cluster. The job slot limit for hosts exported to a remote cluster is determined by the host export policy, not by this parameter. The job slot limit for borrowed hosts is determined by the host export policy of the remote cluster.

If JOBS are configured in the Limit section, the most restrictive limit is applied.

If HOSTS are configured in the Limit section, SLOTS is the number of running and suspended jobs on a host. If preemptive scheduling is used, the suspended jobs are not counted as using a job slot.

To fully use the CPU resource on multiprocessor hosts, make the number of job slots equal to or greater than the number of processors.

Use this parameter to prevent a host from being overloaded with too many jobs, and to maximize the throughput of a machine.

Use “!” to make the number of job slots equal to the number of CPUs on a host.

If the number of CPUs in a host changes dynamically, `mbat chd` adjusts the maximum number of job slots per host accordingly. Allow the `mbat chd` up to 10 minutes to get the number of CPUs for a host. During this period the value of SLOTS is 1.

If only QUEUES are configured in the Limit section, SLOTS is the maximum number of job slots available to the listed queues.

If only USERS are configured in the Limit section, SLOTS is the maximum number of job slots that the users or user groups can use.

If only HOSTS are configured in the Limit section, SLOTS is the maximum number of job slots that are available to the listed hosts.

If only PROJECTS are configured in the Limit section, SLOTS is the maximum number of job slots that are available to the listed projects.

If only LIC\_PROJECTS are configured in the Limit section, SLOTS is the maximum number of job slots that are available to the listed license projects.

Use QUEUES or PER\_QUEUE, USERS or PER\_USER, HOSTS or PER\_HOST, LIC\_PROJECTS or PER\_LIC\_PROJECT, and PROJECTS or PER\_PROJECT in combination to further limit job slots per processor available to resource consumers.

In horizontal format, use only one SLOTS line per Limit section.

In vertical format, use empty parentheses () or a dash (-) to indicate the default value (no limit). Fields cannot be left blank.

## Default

No limit

## Example

```
SLOTS=20
```

# SLOTS\_PER\_PROCESSOR

## Syntax

**SLOTS\_PER\_PROCESSOR**=*number*

**SLOTS\_PER\_PROCESSOR**

- | *number*

## Description

Per processor job slot limit, based on the number of processors on each host affected by the limit.

Maximum number of job slots that each resource consumer can use per processor. This job slot limit is configured per processor so that multiprocessor hosts will automatically run more jobs.

You must also specify PER\_HOST and list the hosts that the limit is to be enforced on. The Limit section is ignored if SLOTS\_PER\_PROCESSOR is specified:

- Without PER\_HOST, or
- With HOSTS

In vertical format, use empty parentheses () or a dash (-) to indicate the default value (no limit). Fields cannot be left blank.

To fully use the CPU resource on multiprocessor hosts, make the number of job slots equal to or greater than the number of processors.

Use this parameter to prevent a host from being overloaded with too many jobs, and to maximize the throughput of a machine.

This number can be a fraction such as 0.5, so that it can also serve as a per-CPU limit on multiprocessor machines. This number is rounded up to the nearest integer equal to or greater than the total job slot limits for a host. For example, if SLOTS\_PER\_PROCESSOR is 0.5, on a 4-CPU multiprocessor host, users can only use up to 2 job slots at any time. On a single-processor machine, users can use 1 job slot.

Use "!" to make the number of job slots equal to the number of CPUs on a host.

If the number of CPUs in a host changes dynamically, `mbat chd` adjusts the maximum number of job slots per host accordingly. Allow the `mbat chd` up to 10 minutes to get the number of CPUs for a host. During this period the number of CPUs is 1.

If only `QUEUES` and `PER_HOST` are configured in the Limit section, `SLOTS_PER_PROCESSOR` is the maximum amount of job slots per processor available to the listed queues for any hosts, users, license projects, or projects.

If only `USERS` and `PER_HOST` are configured in the Limit section, `SLOTS_PER_PROCESSOR` is the maximum amount of job slots per processor that the users or user groups can use on any hosts, queues, license projects, or projects.

If only `PER_HOST` is configured in the Limit section, `SLOTS_PER_PROCESSOR` is the maximum amount of job slots per processor available to the listed hosts for any users, queues, license projects, or projects.

If only `PROJECTS` and `PER_HOST` are configured in the Limit section, `SLOTS_PER_PROCESSOR` is the maximum amount of job slots per processor available to the listed projects for any users, queues, license projects, or hosts.

Use `QUEUES` or `PER_QUEUE`, `USERS` or `PER_USER`, `PER_HOST`, `LIC_PROJECTS` or `PER_LIC_PROJECT`, and `PROJECTS` or `PER_PROJECT` in combination to further limit job slots per processor available to resource consumers.

## Default

No limit

## Example

```
SLOTS_PER_PROCESSOR=2
```

## SWP

### Syntax

**SWP**=*integer*[%]

**SWP**

- | *integer*[%]

## Description

Maximum amount of swap space available to resource consumers. Specify a value in MB or a percentage (%) as a positive integer greater than or equal 0. If you specify a percentage, you must also specify `PER_HOST` and list the hosts that the limit is to be enforced on.

The Limit section is ignored if `SWP` is specified as a percentage:

- Without `PER_HOST`, or
- With `HOSTS`

In horizontal format, use only one `SWP` line per Limit section.

In vertical format, use empty parentheses () or a dash (-) to indicate the default value (no limit). Fields cannot be left blank.

If only `QUEUES` are configured in the Limit section, `SWP` must be an integer value. `SWP` is the maximum amount of swap space available to the listed queues for any hosts, users, license projects, or projects.

If only **USERS** are configured in the Limit section, **SWP** must be an integer value. **SWP** is the maximum amount of swap space that the users or user groups can use on any hosts, queues, license projects, or projects.

If only **HOSTS** are configured in the Limit section, **SWP** must be an integer value. **SWP** is the maximum amount of swap space available to the listed hosts for any users, queues, license projects, or projects.

If only **PROJECTS** are configured in the Limit section, **SWP** must be an integer value. **SWP** is the maximum amount of swap space available to the listed projects for any users, queues, license projects, or hosts.

If only **LIC\_PROJECTS** are configured in the Limit section, **SWP** must be an integer value. **SWP** is the maximum amount of swap space available to the listed projects for any users, queues, projects, or hosts.

Use **QUEUES** or **PER\_QUEUE**, **USERS** or **PER\_USER**, **HOSTS** or **PER\_HOST**, **LIC\_PROJECTS** or **PER\_LIC\_PROJECT**, and **PROJECTS** or **PER\_PROJECT** in combination to further limit swap space available to resource consumers.

## Default

No limit

## Example

```
SWP=60
```

## TMP

### Syntax

**TMP**=*integer*[%]

**TMP**

- | *integer*[%]

## Description

Maximum amount of **t mp** space available to resource consumers. Specify a value in MB or a percentage (%) as a positive integer greater than or equal 0. If you specify a percentage, you must also specify **PER\_HOST** and list the hosts that the limit is to be enforced on.

The Limit section is ignored if **TMP** is specified as a percentage:

- Without **PER\_HOST**, or
- With **HOSTS**

In horizontal format, use only one **TMP** line per Limit section.

In vertical format, use empty parentheses () or a dash (-) to indicate the default value (no limit). Fields cannot be left blank.

If only **QUEUES** are configured in the Limit section, **TMP** must be an integer value. **TMP** is the maximum amount of **t mp** space available to the listed queues for any hosts, users, license projects, or projects.

If only **USERS** are configured in the Limit section, **TMP** must be an integer value. **TMP** is the maximum amount of **t mp** space that the users or user groups can use on any hosts, queues, license projects, or projects.

If only **HOSTS** are configured in the Limit section, **TMP** must be an integer value. **TMP** is the maximum amount of **t mp** space available to the listed hosts for any users, queues, license projects, or projects.

If only **PROJECTS** are configured in the Limit section, **TMP** must be an integer value. **TMP** is the maximum amount of **tmp** space available to the listed projects for any users, queues, license projects, or hosts.

If only **LIC\_PROJECTS** are configured in the Limit section, **TMP** must be an integer value. **TMP** is the maximum amount of **tmp** space available to the listed projects for any users, queues, projects, or hosts.

Use **QUEUES** or **PER\_QUEUE**, **USERS** or **PER\_USER**, **HOSTS** or **PER\_HOST**, **LIC\_PROJECTS** or **PER\_LIC\_PROJECT**, and **PROJECTS** or **PER\_PROJECT** in combination to further limit **tmp** space available to resource consumers.

## Default

No limit

## Example

```
TMP=20%
```

## USERS

### Syntax

```
USERS=all [-]user_name ... | all [-]user_group ...
```

**USERS**

```
( [-] | all [-]user_name ... | all [-]user_group ... )
```

### Description

A space-separated list of user names or user groups on which limits are enforced. Limits are enforced on all users or groups listed. Limits apply to a group as a whole.

If a group contains a subgroup, the limit also applies to each member in the subgroup recursively.

User names must be valid login names. User group names can be LSF user groups or UNIX and Windows user groups.

To specify a per-user limit, use the **PER\_USER** keyword. Do not configure **USERS** and **PER\_USER** limits in the same Limit section.

In horizontal format, use only one **USERS** line per Limit section.

Use the keyword **all** to configure limits that apply to all users or user groups in a cluster.

Use the not operator (**~**) to exclude users or user groups from the **all** specification in the limit. This is useful if you have a large number of users but only want to exclude a few users or groups from the limit definition.

In vertical format, multiple user names must be enclosed in parentheses.

In vertical format, use empty parentheses **()** or a dash **(-)** to indicate an empty field. Fields cannot be left blank.

## Default

**all** (limits are enforced on all users in the cluster)

## Example

```
USERS=user1 user2
```

## GuaranteedResourcePool section

Defines resource pools for use in resource-based service level agreements (SLAs) with guarantee goals. One guaranteed resource pool can be shared between many guarantee SLAs, and a single SLA can have shares in multiple resource pools.

To use guaranteed resources, configure SLAs with **GOALS=[GUARANTEE]** in the `lsb.servicelasses` file.

## GuaranteedResourcePool section structure

Each resource pool is defined in a `GuaranteedResourcePool` section and enclosed by `Begin GuaranteedResourcePool` and `End GuaranteedResourcePool`.

You must configure a `NAME`, `TYPE` and `DISTRIBUTION` for each `GuaranteedResourcePool` section.

The order of `GuaranteedResourcePool` sections is important, as the sections are evaluated in the order configured. Each host can only be in one `GuaranteedResourcePool` section; ensure all `GuaranteedResourcePool` sections (except the last one) define the `HOSTS` parameter, so they do not contain the default of all hosts.

## Example GuaranteedResourcePool sections

```
Begin GuaranteedResourcePool
NAME = linuxGuarantee
TYPE = slots
HOSTS = linux_group
DISTRIBUTION = [sla1, 25] [sla2, 30]
LOAN_POLICIES=QUEUES[all] DURATION[15]
DESCRIPTION = This is the resource pool for the hostgroup linux_group, with 25 slots
guaranteed to sla1 and 30 slots guaranteed to sla2. Resources are loaned to jobs from
any queue with runtimes of up to 15 minutes.
End GuaranteedResourcePool

Begin GuaranteedResourcePool
NAME = solarisGuarantee
TYPE = slots
HOSTS = solaris_group
DISTRIBUTION = [sla1, 25]
LOAN_POLICIES=QUEUES[short_jobs] DURATION[15]
DESCRIPTION = This is the resource pool for the hostgroup solaris_group using the queue
solaris, with 25 slots guaranteed to sla1. Resources are loaned to jobs for up to 15
minutes.
End GuaranteedResourcePool
```

```

Begin GuaranteedResourcePool
NAME = license2pool
TYPE = resource[f2]
DISTRIBUTION = [sla1, 25%] [sla2, 25%]
LOAN_POLICIES=QUEUES[all] DURATION[10]
DESCRIPTION = This is the resource pool for all f2 licenses managed by Platform License Scheduler, with 25% guaranteed to each of SLA1 and SLA2. Resources are loaned to jobs from any queue with runtimes of up to 10 minutes.
End GuaranteedResourcePool

```

## Parameters

- NAME
- TYPE
- HOSTS
- RES\_SELECT
- DISTRIBUTION
- POLICIES
- DESCRIPTION
- SLOTS\_PER\_HOST

## NAME

### Syntax

**NAME**=*name*

### Description

The name of the guarantee.

### Default

None. You must provide a name for the guarantee.

## TYPE

### Syntax

**TYPE**=**slots** | **hosts** | **resource**[*shared\_resource*]

### Description

The type of the guarantee.

Specify **slots** to have slots on hosts within the resource pool reserved by SLA guarantee jobs.

Specify **hosts** to have entire hosts reserved by SLA guarantee jobs.

Specify **resource**[ *shared\_resource*] to guarantee licenses managed by Platform License Scheduler to SLA guarantee jobs. Only one License Scheduler shared resource may be specified for a guaranteed resource pool, and it must be numeric, consumable, decreasing, and available from all hosts in the cluster.

## Default

None. You must specify the type of guarantee.

## HOSTS

### Syntax

**HOSTS**=[~]*host\_name* | [~]*host\_group* | **all** | **allremote** | **all@cluster\_name...**

### Description

A space-separated list of hosts or host groups defined in `lsb.hosts`, on which the guarantee is enforced.

Use the keyword `all` to include all hosts in a cluster. Use the not operator (`~`) to exclude hosts from the `all` specification in the guarantee.

Use host groups for greater flexibility, since host groups have additional configuration options.

Ensure all `GuaranteedResourcePool` sections (except the last one) define the `HOSTS` parameter, so they do not contain the default of all hosts.

## Default

`all` (the guarantee is made over all hosts)

## RES\_SELECT

### Syntax

**RES\_SELECT**=*res\_req*

### Description

Resource requirement string all hosts used in the resource pool must satisfy. For example, **RES\_SELECT=type==LINUX86**

Only static host attributes can be used in `RES_SELECT`. Do not use consumable resources or dynamic resources.

## Default

None. `RES_SELECT` is optional.

## DISTRIBUTION

### Syntax

**DISTRIBUTION**=(*[sla\_name, shares]...*)

### Description

Resource distribution among SLAs, where *shares* can be absolute numbers or a percentage of the resources in the pool. The outer brackets are optional.

When configured as a percentage, the total can exceed 100% but each assigned share cannot exceed 100%. For example:

**DISTRIBUTION=[SLA1,50%] [SLA2, 50%] [SLA3,50%]** is an acceptable configuration even though the total shares assigned sum to 150%.

**DISTRIBUTION=[SLA1,120%]** is not an acceptable configuration, since the share for SLA1 is greater than 100%.

Each SLA (service level agreement) must be configured in lsb. serviceclasses, with **GOALS=[GUARANTEE]**.

## Default

None. You must provide a distribution for the resource pool.

## LOAN\_POLICIES

### Syntax

**LOAN\_POLICIES=QUEUES[*queue\_name* ...|all] [CLOSE\_ON\_DEMAND] [DURATION[*minutes*]]**

### Description

Setting **LOAN\_POLICIES=QUEUES[all]** enables loaning of unused resources within the resource pool by jobs from any queue. When not enabled, resources are reserved regardless of use by SLA jobs.

**QUEUES[*queue\_name*]** loans only to jobs from the specified queue or queues.

**DURATION[*minutes*]** only allows jobs to borrow the resources if the job run limit (or estimated run time) is no larger than *minutes*. Loans limited by job duration make the guaranteed resources available within the time specified by *minutes*. Jobs running longer than the estimated run time will run to completion regardless of the actual run time.

**CLOSE\_ON\_DEMAND** halts loans within the resource pool when SLAs with unmet guarantees have pending demand for resources in the pool. This is useful when running large parallel jobs which may need to wait for sufficient resources to become available.

## Default

None. LOAN\_POLICIES is optional.

## DESCRIPTION

### Syntax

**DESCRIPTION=*description***

### Description

A description of the guarantee resource pool.

## Default

None. DESCRIPTION is optional.

## SLOTS\_PER\_HOST

### Syntax

**SLOTS\_PER\_HOST=*number***

## Description

The maximum number of slots each host within the guaranteed resource pool contributes to the pool.

When defined, SLOTS\_PER\_HOST limits the number of slots any SLA jobs with guarantees in the guaranteed resource pool can use on each host. This limit applies to all hosts in the pool, regardless of guarantees.

## Default

None. SLOTS\_PER\_HOST is optional.

## HostExport section

Defines an export policy for a host or a group of related hosts. Defines how much of each host's resources are exported, and how the resources are distributed among the consumers.

Each export policy is defined in a separate HostExport section, so it is normal to have multiple HostExport sections in `lsb.resources`.

## HostExport section structure

Use empty parentheses () or a dash (-) to specify the default value for an entry. Fields cannot be left blank.

## Example HostExport section

```
Begin HostExport PER_HOST= hostA hostB SLOTS= 4 DISTRIBUTION= [cluster1, 1] [cluster2,
3] MEM= 100 SWP= 100 End HostExport
```

## Parameters

- PER\_HOST
- RES\_SELECT
- NHOSTS
- DISTRIBUTION
- MEM
- SLOTS
- SWAP
- TYPE

## PER\_HOST

### Syntax

**PER\_HOST**=*host\_name...*

### Description

Required when exporting special hosts.

Determines which hosts to export. Specify one or more LSF hosts by name. Separate names by space.

## RES\_SELECT

### Syntax

**RES\_SELECT**=*res\_req*

## Description

Required when exporting workstations.

Determines which hosts to export. Specify the selection part of the resource requirement string (without quotes or parentheses), and LSF will automatically select hosts that meet the specified criteria. For this parameter, if you do not specify the required host type, the default is `type==any`.

When `LSF_STRICT_RESREQ=Y` is configured in `lsf.conf`, resource requirement strings in `select` sections must conform to a more strict syntax. The strict resource requirement syntax only applies to the `select` section. It does not apply to the other resource requirement sections (`order`, `rusage`, `same`, `span`, or `cu`). When `LSF_STRICT_RESREQ=Y` in `lsf.conf`, LSF rejects resource requirement strings where an `rusage` section contains a non-consumable resource.

The criteria is only evaluated once, when a host is exported.

## NHOSTS

### Syntax

**NHOSTS**=*integer*

### Description

Required when exporting workstations.

Maximum number of hosts to export. If there are not this many hosts meeting the selection criteria, LSF exports as many as it can.

## DISTRIBUTION

### Syntax

**DISTRIBUTION**=(*[cluster\_name, number\_shares]...*)

### Description

Required. Specifies how the exported resources are distributed among consumer clusters.

The syntax for the distribution list is a series of share assignments. The syntax of each share assignment is the cluster name, a comma, and the number of shares, all enclosed in square brackets, as shown. Use a space to separate multiple share assignments. Enclose the full distribution list in a set of round brackets.

*cluster\_name*

Specify the name of a remote cluster that will be allowed to use the exported resources. If you specify a local cluster, the assignment is ignored.

*number\_shares*

Specify a positive integer representing the number of shares of exported resources assigned to the cluster.

The number of shares assigned to a cluster is only meaningful when you compare it to the number assigned to other clusters, or to the total number. The total number of shares is just the sum of all the shares assigned in each share assignment.

## MEM

### Syntax

**MEM**=*megabytes*

### Description

Used when exporting special hosts. Specify the amount of memory to export on each host, in MB.

### Default

- (provider and consumer clusters compete for available memory)

## SLOTS

### Syntax

**SLOTS**=*integer*

### Description

Required when exporting special hosts. Specify the number of job slots to export on each host.

To avoid overloading a partially exported host, you can reduce the number of job slots in the configuration of the local cluster.

## SWAP

### Syntax

**SWAP**=*megabytes*

### Description

Used when exporting special hosts. Specify the amount of swap space to export on each host, in MB.

### Default

- (provider and consumer clusters compete for available swap space)

## TYPE

### Syntax

**TYPE**=**shared**

### Description

Changes the lease type from exclusive to shared.

If you export special hosts with a shared lease (using PER\_HOST), you cannot specify multiple consumer clusters in the distribution policy.

### Default

Undefined (the lease type is exclusive; exported resources are never available to the provider cluster)

## SharedResourceExport section

Optional. Requires HostExport section. Defines an export policy for a shared resource. Defines how much of the shared resource is exported, and the distribution among the consumers.

The shared resource must be available on hosts defined in the HostExport sections.

## SharedResourceExport section structure

All parameters are required.

## Example SharedResourceExport section

```
Begin SharedResourceExport
NAME= AppLicense
NINSTANCES= 10
DISTRIBUTION= ([C1, 30] [C2, 70])
End SharedResourceExport
```

## Parameters

- NAME
- NINSTANCES
- DISTRIBUTION

## NAME

### Syntax

**NAME**=*shared\_resource\_name*

### Description

Shared resource to export. This resource must be available on the hosts that are exported to the specified clusters; you cannot export resources without hosts.

## NINSTANCES

### Syntax

**NINSTANCES**=*integer*

### Description

Maximum quantity of shared resource to export. If the total number available is less than the requested amount, LSF exports all that are available.

## DISTRIBUTION

### Syntax

**DISTRIBUTION**=(*[cluster\_name, number\_shares]...*)

### Description

Specifies how the exported resources are distributed among consumer clusters.

The syntax for the distribution list is a series of share assignments. The syntax of each share assignment is the cluster name, a comma, and the number of shares, all enclosed in square brackets, as shown. Use a space to separate multiple share assignments. Enclose the full distribution list in a set of round brackets.

*cluster\_name*

Specify the name of a cluster allowed to use the exported resources.

*number\_shares*

Specify a positive integer representing the number of shares of exported resources assigned to the cluster.

The number of shares assigned to a cluster is only meaningful when you compare it to the number assigned to other clusters, or to the total number. The total number of shares is the sum of all the shares assigned in each share assignment.

## ResourceReservation section

By default, only LSF administrators or root can add or delete advance reservations.

The ResourceReservation section defines an advance reservation policy. It specifies:

- Users or user groups that can create reservations
- Hosts that can be used for the reservation
- Time window when reservations can be created

Each advance reservation policy is defined in a separate ResourceReservation section, so it is normal to have multiple ResourceReservation sections in `lsb.resources`.

## Example ResourceReservation section

Only user1 and user2 can make advance reservations on hostA and hostB. The reservation time window is between 8:00 a.m. and 6:00 p.m. every day:

```
Begin ResourceReservation
NAME          = dayPolicy
USERS         = user1 user2      # optional
HOSTS         = hostA hostB     # optional
TIME_WINDOW  = 8:00-18:00      # weekly recurring reservation
End ResourceReservation
```

user1 can add the following reservation for user user2 to use on hostA every Friday between 9:00 a.m. and 11:00 a.m.:

```
% user1@hostB> brsvadd -m "hostA" -n 1 -u "user2" -t "5:9:0-5:11:0" Reservation
"user2#2" is created
```

Users can only delete reservations they created themselves. In the example, only user user1 can delete the reservation; user2 cannot. Administrators can delete any reservations created by users.

## Parameters

- HOSTS
- NAME
- TIME\_WINDOW
- USERS

## HOSTS

### Syntax

**HOSTS**=[~]*host\_name* | [~]*host\_group* | **all** | **allremote** | **all@cluster\_name** ...

### Description

A space-separated list of hosts, host groups defined in `lsb.hosts` on which administrators or users specified in the `USERS` parameter can create advance reservations.

The hosts can be local to the cluster or hosts leased from remote clusters.

If a group contains a subgroup, the reservation configuration applies to each member in the subgroup recursively.

Use the keyword `all` to configure reservation policies that apply to all local hosts in a cluster not explicitly excluded. This is useful if you have a large cluster but you want to use the not operator (`~`) to exclude a few hosts from the list of hosts where reservations can be created.

Use the keyword `allremote` to specify all hosts borrowed from all remote clusters.

---

#### Tip:

You cannot specify host groups or host partitions that contain the `allremote` keyword.

---

Use `all@cluster_name` to specify the group of all hosts borrowed from one remote cluster. You cannot specify a host group or partition that includes remote resources.

With MultiCluster resource leasing model, the not operator (`~`) can be used to exclude local hosts or host groups. You cannot use the not operator (`~`) with remote hosts.

### Examples

```
HOSTS=hgroup1 ~hostA hostB hostC
```

Advance reservations can be created on `hostB`, `hostC`, and all hosts in `hgroup1` except for `hostA`.

```
HOSTS=all ~group2 ~hostA
```

Advance reservations can be created on all hosts in the cluster, except for `hostA` and the hosts in `group2`.

### Default

`all allremote` (users can create reservations on all server hosts in the local cluster, and all leased hosts in a remote cluster).

## NAME

### Syntax

**NAME**=*text*

### Description

Required. Name of the ResourceReservation section

Specify any ASCII string 40 characters or less. You can use letters, digits, underscores (\_) or dashes (-). You cannot use blank spaces.

## Example

```
NAME=reservation1
```

## Default

None. You must provide a name for the ResourceReservation section.

## TIME\_WINDOW

### Syntax

```
TIME_WINDOW=time_window...
```

### Description

Optional. Time window for users to create advance reservations. The time for reservations that users create must fall within this time window.

Use the same format for *time\_window* as the recurring reservation option (-t) of `brsvadd`. To specify a time window, specify two time values separated by a hyphen (-), with no space in between:

```
time_window = begin_time-end_time
```

### Time format

Times are specified in the format:

```
[day: ]hour[:minute]
```

where all fields are numbers with the following ranges:

- *day of the week*: 0-6 (0 is Sunday)
- *hour*: 0-23
- *minute*: 0-59

Specify a time window one of the following ways:

- *hour-hour*
- *hour.minute-hour.minute*
- *day.hour.minute-day.hour.minute*

The default value for minute is 0 (on the hour); the default value for day is every day of the week.

You must specify at least the hour. Day of the week and minute are optional. Both the start time and end time values must use the same syntax. If you do not specify a minute, LSF assumes the first minute of the hour (: 00). If you do not specify a day, LSF assumes every day of the week. If you do specify the day, you must also specify the minute.

You can specify multiple time windows, but they cannot overlap. For example:

```
timeWindow(8: 00- 14: 00 18: 00- 22: 00)
```

is correct, but

```
timeWindow(8: 00- 14: 00 11: 00- 15: 00)
```

is not valid.

## Example

```
TIME_WINDOW=8:00-14:00
```

Users can create advance reservations with begin time (`brsvadd -b`), end time (`brsvadd -e`), or time window (`brsvadd -t`) on any day between 8:00 a.m. and 2:00 p.m.

## Default

Undefined (any time)

## USERS

### Syntax

```
USERS=[~]user_name | [~]user_group ... | all
```

### Description

A space-separated list of user names or user groups who are allowed to create advance reservations. Administrators, root, and all users or groups listed can create reservations.

If a group contains a subgroup, the reservation policy applies to each member in the subgroup recursively.

User names must be valid login names. User group names can be LSF user groups or UNIX and Windows user groups.

Use the keyword `all` to configure reservation policies that apply to all users or user groups in a cluster. This is useful if you have a large number of users but you want to exclude a few users or groups from the reservation policy.

Use the not operator (`~`) to exclude users or user groups from the list of users who can create reservations.

---

#### Caution:

The not operator does not exclude LSF administrators from the policy.

## Example

```
USERS=user1 user2
```

## Default

`all` (all users in the cluster can create reservations)

## ReservationUsage section

To enable greater flexibility for reserving numeric resources that are reserved by jobs, configure the `ReservationUsage` section in `lsb.resources` to reserve resources like license tokens per resource as `PER_JOB`, `PER_SLOT`, or `PER_HOST`. For example:

## Example ReservationUsage section

```

Begin Reservati onUsage
RESOURCE          METHOD          RESERVE
licenseX          PER_JOB          Y
licenseY          PER_HOST          N
licenseZ          PER_SLOT          N
End Reservati onUsage

```

## Parameters

- RESOURCE
- METHOD
- RESERVE

## RESOURCE

The name of the resource to be reserved. User-defined numeric resources can be reserved, but only if they are shared (they are not specific to one host).

The following built-in resources can be configured in the ReservationUsage section and reserved:

- mem
- tmp
- swp

Any custom resource can also be reserved if it is shared (defined in the Resource section of `lsf.shared`) or host based (listed in the Host section of the `lsf.cluster` file in the resource column).

## METHOD

The resource reservation method. One of:

- PER\_JOB
- PER\_HOST
- PER\_SLOT

The cluster-wide RESOURCE\_RESERVE\_PER\_SLOT parameter in `lsb.params` is obsolete.

RESOURCE\_RESERVE\_PER\_SLOT parameter still controls resources not configured in `lsb.resources`. Resources not reserved in `lsb.resources` are reserved per job.

PER\_HOST reservation means that for the parallel job, LSF reserves one instance of a for each host. For example, some application licenses are charged only once no matter how many applications are running provided those applications are running on the same host under the same user.

Use no method ("-") when setting mem, swp, or tmp as RESERVE=Y.

## RESERVE

Reserves the resource for pending jobs that are waiting for another resource to become available.

For example, job A requires resources X, Y, and Z to run, but resource Z is a high demand or scarce resource. This job pends until Z is available. In the meantime, other jobs requiring only X and Y resources run. If X and Y are set as reservable resources (the RESERVE parameter is set to "Y"), as soon as Z resource is available, job A runs. If they are not, job A may never be able to run because all resources are never available at the same time.

---

**Restriction:**

Only the following built-in resources can be defined as reservable:

- mem
  - swp
  - tmp
- 

Use no method ("-") when setting mem, swp, or tmp as RESERVE=Y.

When submitting a job, the queue must have RESOURCE\_RESERVE defined.

Backfill of the reservable resources is also supported when you submit a job with reservable resources to a queue with BACKFILL defined.

Valid values are Y and N. If not specified, resources are not reserved.

## Assumptions and limitations

- Per-resource configuration defines resource usage for individual resources, but it does not change any existing resource limit behavior (PER\_JOB, PER\_SLOT).
- In a MultiCluster environment, you should configure resource usage in the scheduling cluster (submission cluster in lease model or receiving cluster in job forward model).

## Automatic time-based configuration

Variable configuration is used to automatically change LSF configuration based on time windows. You define automatic configuration changes in `lsb.resources` by using if-else constructs and time expressions. After you change the files, reconfigure the cluster with the `badm n reconfi g` command.

The expressions are evaluated by LSF every 10 minutes based on `mbat chd` start time. When an expression evaluates true, LSF dynamically changes the configuration based on the associated configuration statements. Reconfiguration is done in real time without restarting `mbat chd`, providing continuous system availability.

## Example

```

# limit usage of hosts in 'license1' group and time
# based configuration
# - 10 jobs can run from normal queue
# - any number can run from short queue between 18:30
#   and 19:30
# all other hours you are limited to 100 slots in the
# short queue
# - each other queue can run 30 jobs

Begin Limit
PER_QUEUE          HOSTS          SLOTS      # Example
normal             license1      10
# if time(18:30-19:30)
short             license1      -
#else
short             license1      100
#endif
(all ~normal ~short) license1      30
End Limit

```

# lsb.serviceclasses

The `lsb.serviceclasses` file defines the service-level agreements (SLAs) in an LSF cluster as *service classes*, which define the properties of the SLA.

This file is optional.

You can configure as many service class sections as you need.

Use `bsl a` to display the properties of service classes configured in `lsb.serviceclasses` and dynamic information about the state of each configured service class.

By default, `lsb.serviceclasses` is installed in `LSB_CONFDIR/cluster_name/confidir`.

## Changing lsb.serviceclasses configuration

After making any changes to `lsb.serviceclasses`, run `badm in reconfi g` to reconfigure `mbat chd`.

## lsb.serviceclasses structure

Each service class definition begins with the line `Begin Servi ceCl ass` and ends with the line `End Servi ceCl ass`.

## Syntax

```

Begin Servi ceCl ass
NAME           = string
PRIORITY       = integer
GOALS          = [throughput | velocity | deadline] [\... ]
CONTROL_ACTION = VIOLATION_PERIOD[minutes] CMD [action]
USER_GROUP     = all | [user_name] [user_group] ...
DESCRIPTION    = text
End Servi ceCl ass

Begin Servi ceCl ass
NAME           = string
GOALS          = guarantee
ACCESS_CONTROL = [QUEUES[ queue ... ]] [USERS[ [user_name] [user_group] ... ]]
               [FAIRSHARE_GROUPS[ user_group ... ]] [APPS[ app_name ... ]] [PROJECTS[ proj_name ... ]]
               [LIC_PROJECTS[ license_proj ... ]]
AUTO_ATTACH    = Y | y | N | n
DESCRIPTION    = text
End Servi ceCl ass

```

You must specify:

- Service class name
- Goals

Service classes with guarantee goals cannot have `PRIORITY`, `CONTROL_ACTION` or `USER_GROUP` defined.

To configure EGO-enabled SLA scheduling, you must specify an existing EGO consumer name to allow the SLA to get host allocations from EGO.

All other parameters are optional.

## Example

```

Begin ServiceClass
NAME=Sooke
PRIORITY=20
GOALS=[DEADLINE timeWindow (8:30-16:00)]
DESCRIPTION="working hours"
End ServiceClass

Begin ServiceClass
NAME=Newmarket
GOALS=[GUARANTEE]
ACCESS_CONTROL = QUEUES[batch] FAIRSHARE_GROUPS[team2]
AUTO_ATTACH = Y
DESCRIPTION="guarantee for team2 batch jobs"
End ServiceClass

```

## Parameters

- ACCESS\_CONTROL
- AUTO\_ATTACH
- CONSUMER
- CONTROL\_ACTION
- DESCRIPTION
- EGO\_RES\_REQ
- GOALS
- MAX\_HOST\_IDLE\_TIME
- NAME
- PRIORITY
- USER\_GROUP

## ACCESS\_CONTROL

### Syntax

```

ACCESS_CONTROL=[QUEUES[ queue ...]] [USERS[ [user_name] [user_group] ...]]
[FAIRSHARE_GROUPS[user_group ...]] [APPS[app_name ...]] [PROJECTS[proj_name...]]
[LIC_PROJECTS[lic_proj...]]

```

### Description

Guarantee SLAs (with **GOALS=[GUARANTEE]**) only.

Restricts access to a guarantee SLA. If more than one restriction is configured, all must be satisfied.

- QUEUES restricts access to the queues listed; the queue is specified for jobs at submission using `bsub -q`.
- USERS restricts access to jobs submitted by the users or user groups specified.

User names must be valid login names. To specify a Windows user account, include the domain name in uppercase letters (*DOMAIN\_NAME\user\_name*). User group names can be LSF user groups or UNIX and Windows user groups. To specify a Windows user group, include the domain name in uppercase letters (*DOMAIN\_NAME\user\_group*).

- FAIRSHARE\_GROUPS restricts access to the fairshare groups listed; the fairshare group is specified for jobs at submission using `bsub -G`.
- APPS restricts access to the application profiles listed; the application profile is specified for jobs at submission using `bsub -app`.
- PROJECTS restricts access to the projects listed; the project is specified for jobs at submission using `bsub -P`.
- LIC\_PROJECTS restricts access to the license projects listed; the license project is specified for jobs at submission using `bsub -Lp`.

## Example

```
ACCESS_CONTROL = QUEUES[normal short] USERS[ug1]
```

Jobs submitted to the queues `normal` or `short` by users in usergroup `ug1` are the only jobs accepted by the guarantee SLA.

## Default

None. Access to the guarantee SLA is not restricted.

# AUTO\_ATTACH

## Syntax

```
AUTO_ATTACH=Y | y | N | n
```

## Description

Guarantee SLAs (with **GOALS=[GUARANTEE]**) only. Used with `ACCESS_CONTROL`.

Enabling `AUTO_ATTACH` when a guarantee SLA has `ACCESS_CONTROL` configured results in submitted jobs automatically attaching to the guarantee SLA if they have access. If a job can access multiple guarantee SLAs with `AUTO_ATTACH` enabled, the job is automatically attached to the first accessible SLA based on configuration order in the `lsb.serviceclasses` file.

During restart or reconfiguration, automatic attachments to guarantee SLAs are checked and jobs may be attached to a different SLA. During live reconfiguration (using the `bconf` command) automatic attachments are not checked, and jobs remain attached to the same guarantee SLAs regardless of configuration changes.

## Example

```

Begin ServiceClass
...
NAME = Maple
GOALS = [GUARANTEE]
ACCESS_CONTROL = QUEUES[priority] USERS[ug1]
AUTO_ATTACH = Y
...
End ServiceClass

```

All jobs submitted to the priority queue by users in user group ug1 and submitted without an SLA specified are automatically attached to the service class Maple.

## Default

N

# CONSUMER

## Syntax

**CONSUMER**=*ego\_consumer\_name*

## Description

For EGO-enabled SLA service classes, the name of the EGO consumer from which hosts are allocated to the SLA. This parameter is not mandatory, but must be configured for the SLA to receive hosts from EGO.

Guarantee SLAs (with **GOALS**=[**GUARANTEE**]) cannot have CONSUMER set. If defined, it will be ignored.

---

### Important:

CONSUMER must specify the name of a valid consumer in EGO. If a default SLA is configured with ENABLE\_DEFAULT\_EGO\_SLA in lsb.params, all services classes configured in lsb.serviceclasses must specify a consumer name.

---

## Default

None

# CONTROL\_ACTION

## Syntax

**CONTROL\_ACTION**=**VIOLATION\_PERIOD**[*minutes*] **CMD** [*action*]

## Description

Optional. Configures a control action to be run if the SLA goal is delayed for a specified number of minutes.

If the SLA goal is delayed for longer than VIOLATION\_PERIOD, the action specified by CMD is invoked. The violation period is reset and if the SLA is still active when the violation period expires again, the action runs again. If the SLA has multiple active goals that are in violation, the action is run for each of them.

Guarantee SLAs (with **GOALS=[GUARANTEE]**) cannot have CONTROL\_ACTION set. If defined, it will be ignored.

## Example

```
CONTROL_ACTION=VIOLATION_PERIOD[10] CMD [echo `date`: SLA is in violation >> ! /tmp/sl_a_violation.log]
```

## Default

None

# DESCRIPTION

## Syntax

**DESCRIPTION**=*text*

## Description

Optional. Description of the service class. Use `bsl a` to display the description text.

This description should clearly describe the features of the service class to help users select the proper service class for their jobs.

The text can include any characters, including white space. The text can be extended to multiple lines by ending the preceding line with a backslash (`\`).

## Default

None

# EGO\_RES\_REQ

## Syntax

**EGO\_RES\_REQ**=*res\_req*

## Description

For EGO-enabled SLA service classes, the EGO resource requirement that specifies the characteristics of the hosts that EGO will assign to the SLA.

Must be a valid EGO resource requirement. The EGO resource requirement string supports the `select` section, but the format is different from LSF resource requirements.

Guarantee SLAs (with **GOALS=[GUARANTEE]**) cannot have EGO\_RES\_REQ set. If defined, it will be ignored.

## Example

```
EGO_RES_REQ=select (linux && maxmem > 100)
```

## Default

None

# GOALS

## Syntax

```
GOALS=[throughput | velocity | deadline] [\
```

```
[throughput | velocity | deadline] ...]
```

```
GOALS=[guarantee]
```

## Description

*Required.* Defines the service-level goals for the service class. A service class can have more than one goal, each active at different times of the day and days of the week. Outside of the time window, the SLA is inactive and jobs are scheduled as if no service class is defined. LSF does not enforce any service-level goal for an inactive SLA.

The time windows of multiple service-level goals can overlap. In this case, the largest number of jobs is run.

An active SLA can have a status of `On time` if it is meeting the goal, and a status `Delayed`, if it is missing its goals.

A service-level goal defines:

*throughput* — expressed as *finished* jobs per hour and an optional time window when the goal is active. *throughput* has the form:

```
GOALS=[THROUGHPUT num_jobs timeWindow [(time_window)]]
```

If no time window is configured, `THROUGHPUT` can be the only goal in the service class. The service class is always active, and `bsl a` displays `ACTIVE WINDOW: Always Open`.

*velocity* — expressed as *concurrently* running jobs and an optional time window when the goal is active. *velocity* has the form:

```
GOALS=[VELOCITY num_jobs timeWindow [(time_window)]]
```

If no time window is configured, `VELOCITY` can be the only goal in the service class. The service class is always active, and `bsl a` displays `ACTIVE WINDOW: Always Open`.

*deadline* — indicates that all jobs in the service class should complete by the end of the specified time window. The time window is required for a deadline goal. *deadline* has the form:

```
GOALS=[DEADLINE timeWindow (time_window)]
```

*guarantee* — indicates the SLA has guaranteed resources defined in `lsb.resources` and is able to guarantee resources, depending on the scavenging policies configured. Guarantee goals cannot be combined with any other goals, and do not accept time windows.

```
GOALS=[GUARANTEE]
```

---

### Restriction:

EGO-enabled SLA service classes only support velocity goals. Deadline, throughput, and guarantee goals are not supported. The configured

velocity value for EGO-enabled SLA service classes is considered to be a *minimum* number of jobs that should be in run state from the SLA

## Time window format

The time window of an SLA goal has the standard form:

```
begin_time-end_time
```

Times are specified in the format:

```
[day: ]hour[:minute]
```

where all fields are numbers with the following ranges:

- *day of the week*: 0-6 (0 is Sunday)
- *hour*: 0-23
- *minute*: 0-59

Specify a time window one of the following ways:

- *hour-hour*
- *hour.minute-hour.minute*
- *day:hour.minute-day:hour.minute*

The default value for minute is 0 (on the hour); the default value for day is every day of the week.

You must specify at least the hour. Day of the week and minute are optional. Both the start time and end time values must use the same syntax. If you do not specify a minute, LSF assumes the first minute of the hour (: 00). If you do not specify a day, LSF assumes every day of the week. If you do specify the day, you must also specify the minute.

You can specify multiple time windows, but they cannot overlap. For example:

```
timeWindow(8: 00- 14: 00 18: 00- 22: 00)
```

is correct, but

```
timeWindow(8: 00- 14: 00 11: 00- 15: 00)
```

is not valid.

### Tip:

To configure a time window that is always open, use the timeWindow keyword with empty parentheses.

## Examples

```
GOALS=[ THROUGHPUT 2 timeWindow () ]
```

```
GOALS=[ THROUGHPUT 10 timeWindow (8: 30- 16: 30) ]
```

```
GOALS=[ VELOCITY 5 timeWindow () ]
```

```
GOALS=[ DEADLINE timeWindow (16: 30- 8: 30) ]\
```

```
    [ VELOCITY 10 timeWindow (8: 30- 16: 30) ]
```

```
GOALS=[ GUARANTEE]
```

## MAX\_HOST\_IDLE\_TIME

### Syntax

```
MAX_HOST_IDLE_TIME=seconds
```

## Description

For EGO-enabled SLA service classes, number of seconds that the SLA will hold its idle hosts before LSF releases them to EGO. Each SLA can configure a different idle time. Do not set this parameter to a small value, or LSF may release hosts too quickly.

Guarantee SLAs (with **GOALS=[GUARANTEE]**) cannot have MAX\_HOST\_IDLE\_TIME set. If defined, it will be ignored.

## Default

120 seconds

## NAME

## Syntax

**NAME**=*string*

## Description

*Required.* A unique name that identifies the service class.

Specify any ASCII string 60 characters or less. You can use letters, digits, underscores (\_) or dashes (-). You cannot use blank spaces.

---

### Important:

The name you use cannot be the same as an existing host partition, user group name, or fairshare queue name.

---

## Example

```
NAME=Tofino
```

## Default

None. You must provide a unique name for the service class.

## PRIORITY

## Syntax

**PRIORITY**=*integer*

## Description

*Required (time-based SLAs only).* The service class priority. A higher value indicates a higher priority, relative to other service classes. Similar to queue priority, service classes access the cluster resources in priority order.

LSF schedules jobs from one service class at a time, starting with the highest-priority service class. If multiple service classes have the same priority, LSF runs all the jobs from these service classes in first-come, first-served order.

Service class priority in LSF is completely independent of the UNIX scheduler's priority system for time-sharing processes. In LSF, the NICE parameter is used to set the UNIX time-sharing priority for batch jobs.

Guarantee SLAs (with **GOALS=[GUARANTEE]**) cannot have PRIORITY set. If defined, it will be ignored.

## Default

None.

# USER\_GROUP

## Syntax

```
USER_GROUP=all | [user_name] [user_group] ...
```

## Description

Optional. A space-separated list of user names or user groups who can submit jobs to the service class. Administrators, root, and all users or groups listed can use the service class.

Use the reserved word `al 1` to specify all LSF users. LSF cluster administrators are automatically included in the list of users, so LSF cluster administrators can submit jobs to any service class, or switch any user's jobs into this service class, even if they are not listed.

If user groups are specified in `l sb. users`, each user in the group can submit jobs to this service class. If a group contains a subgroup, the service class policy applies to each member in the subgroup recursively. If the group can define fairshare among its members, the SLA defined by the service class enforces the fairshare policy among the users of the SLA.

User names must be valid login names. User group names can be LSF user groups (in `l sb. users`) or UNIX and Windows user groups.

Guarantee SLAs (with **GOALS=[GUARANTEE]**) cannot have `USER_GROUP` set. If defined, it will be ignored.

## Example

```
USER_GROUP=user1 user2 ugroup1
```

## Default

`al 1` (all users in the cluster can submit jobs to the service class)

## Examples

- The resource-based service class AccountingSLA guarantees hosts to the user group `accountingUG` for jobs submitted to the queue `longj obs`. Jobs submitted to this queue by this

usergroup without an SLA specified will be automatically attached to the SLA. The guaranteed resource pools used by the SLA are configured in `lsb.resources`.

```

Begin ServiceClass
NAME=AccountingSLA
GOALS=[ GUARANTEE]
DESCRIPTION="Guaranteed hosts for the accounting department "
ACCESS_CONTROL = QUEUES[longjobs] USERS[accountingUG]
AUTO_ATTACH = Y
End ServiceClass

```

- The service class `Sooke` defines one deadline goal that is active during working hours between 8:30 AM and 4:00 PM. All jobs in the service class should complete by the end of the specified time window. Outside of this time window, the SLA is inactive and jobs are scheduled without any goal being enforced:

```

Begin ServiceClass
NAME=Sooke
PRIORITY=20
GOALS=[ DEADLINE timeWindow (8:30-16:00) ]
DESCRIPTION="working hours"
End ServiceClass

```

- The service class `Nanaimo` defines a deadline goal that is active during the weekends and at nights.

```

Begin ServiceClass
NAME=Nanaimo
PRIORITY=20
GOALS=[ DEADLINE timeWindow (5:18:00-1:8:30 20:00-8:30) ]
DESCRIPTION="weekend nighttime regression tests"
End ServiceClass

```

- The service class `Sidney` defines a throughput goal of 6 jobs per hour that is always active:

```

Begin ServiceClass
NAME=Sidney
PRIORITY=20
GOALS=[ THROUGHPUT 6 timeWindow () ]
DESCRIPTION="constant throughput"
End ServiceClass

```

- The service class `Tofino` defines two velocity goals in a 24 hour period. The first goal is to have a maximum of 10 concurrently running jobs during business hours (9:00 a.m. to 5:00 p.m). The second goal is a maximum of 30 concurrently running jobs during off-hours (5:30 p.m. to 8:30 a.m.)

```

Begin ServiceClass
NAME=Tofino
PRIORITY=20
GOALS=[ VELOCITY 10 timeWindow (9:00-17:00) ] \
      [ VELOCITY 30 timeWindow (17:30-8:30) ]
DESCRIPTION="day and night velocity"
End ServiceClass

```

- The service class Duncan defines a velocity goal that is active during working hours (9:00 a.m. to 5:30 p.m.) and a deadline goal that is active during off-hours (5:30 p.m. to 9:00 a.m.) Only users user1 and user2 can submit jobs to this service class.

```
Begin ServiceClass
```

```
NAME=Duncan
```

```
PRIORITY=23
```

```
USER_GROUP=user1 user2
```

```
GOALS=[ VELOCITY 8 timeWindow (9:00-17:30) ] \
```

```
    [ DEADLINE timeWindow (17:30-9:00) ]
```

```
DESCRIPTION="Daytime/Nighttime SLA"
```

```
End ServiceClass
```

- The service class Tevere defines a combination similar to Duncan, but with a deadline goal that takes effect overnight and on weekends. During the working hours in weekdays the velocity goal favors a mix of short and medium jobs.

```
Begin ServiceClass
```

```
NAME=Tevere
```

```
PRIORITY=20
```

```
GOALS=[ VELOCITY 100 timeWindow (9:00-17:00) ] \
```

```
    [ DEADLINE timeWindow (17:30-8:30 5:17:30-1:8:30) ]
```

```
DESCRIPTION="nine to five"
```

```
End ServiceClass
```

# lsb.users

The `lsb.users` file is used to configure user groups, hierarchical fairshare for users and user groups, and job slot limits for users and user groups. It is also used to configure account mappings in a MultiCluster environment.

This file is optional.

The `lsb.users` file is stored in the directory `LSB_CONFDIR/cluster_name/configdir`, where `LSB_CONFDIR` is defined in `lsf.conf`.

## Changing lsb.users configuration

After making any changes to `lsb.users`, run `badminton reconfig` to reconfigure `mbatchd`.

## UserGroup section

Optional. Defines user groups.

The name of the user group can be used in other user group and queue definitions, as well as on the command line. Specifying the name of a user group in the `GROUP_MEMBER` section has exactly the same effect as listing the names of all users in the group.

The total number of user groups cannot be more than 1024.

## Structure

The first line consists of two mandatory keywords, `GROUP_NAME` and `GROUP_MEMBER`. The `USER_SHARES` and `GROUP_ADMIN` keywords are optional. Subsequent lines name a group and list its membership and optionally its share assignments and administrator.

Each line must contain one entry for each keyword. Use empty parentheses `()` or a dash `-` to specify the default value for an entry.

---

### Restriction:

If specifying a specific user name for a user group, that entry must precede all user groups.

---

## Examples of a UserGroup section

Example 1:

```

Begin UserGroup
GROUP_NAME    GROUP_MEMBER                GROUP_ADMIN
groupA        (user1 user2 user3 user4) (user5[full])
groupB        (user7 user8 user9)         (groupA[usershares])
groupC        (groupA user5)             (groupA)
groupD        (!) ()
End UserGroup

```

**Example 2:**

```
Begin UserGroup
```

GROUP_NAME	GROUP_MEMBER	GROUP_ADMIN
groupA	(user1 user2 user3 user4)	(user5)
groupB	(groupA user5)	(groupA)
groupC	(!)	()

```
End UserGroup
```

**Example 2:**

```
Begin UserGroup
```

GROUP_NAME	GROUP_MEMBER	USER_SHARES
groupB	(user1 user2)	()
groupC	(user3 user4)	([User3, 3] [User4, 4])
groupA	(GroupB GroupC user5)	([User5, 1] [default, 10])

```
End UserGroup
```

## GROUP\_NAME

An alphanumeric string representing the user group name. You cannot use the reserved name `all` or a `"/` in a group name.

## GROUP\_MEMBER

User group members are the users who belong to the group. You can specify both user names and user group names.

User and user group names can appear on multiple lines because users can belong to multiple groups.

**Note:**

When a user belongs to more than one group, any of the administrators specified for any of the groups the user belongs to can control that users' jobs. Limit administrative control by defining

**STRICT\_UG\_CONTROL=Y** in `lsb.params` and submitting jobs with the `-G` option, specifying which user group the job is submitted with.

User groups may be defined recursively but must not create a loop.

## Syntax

```
(user_name | user_group ...) | (all) | (!)
```

Enclose the entire group member list in parentheses. Use space to separate multiple names.

You can combine user names and user group names in the same list.

## Valid values

- `all`

The reserved name `all` specifies all users in the cluster.

- `!`

An exclamation mark (`!`) indicates an externally-defined user group, which the `egroup` executable retrieves.

- *user\_name*

User names must be valid login names.

To specify a Windows user account, include the domain name in uppercase letters (*DOMAIN\_NAME \user\_name*).

- *user\_group*

User group names can be LSF user groups defined previously in this section, or UNIX and Windows user groups.

If you specify a name that is both a UNIX user group and also a UNIX user, append a backslash to make sure it is interpreted as a group (*user\_group/*).

To specify a Windows user group, include the domain name in uppercase letters (*DOMAIN\_NAME \user\_group*).

## GROUP\_ADMIN

User group administrators can administer the jobs of group members. You can specify both user names and user group names.

- If you specify a user group as an administrator for another user group, all members of the first user group become administrators for the second user group.
- You can also specify that all users of a group are also administrators of that same group.
- Users can be administrators for more than one user group at the same time.

---

### Note:

When a user belongs to more than one group, any of the administrators specified for any of the groups the user belongs to can control that users' jobs. Define **STRICT\_UG\_CONTROL=Y** in *lsb.params* to limit user group administrator control to the user group specified by -G at job submission.

---

By default a user group administrator has privileges equivalent to those of a job owner, and is allowed to control any job belonging to member users of the group they administer. A user group administrator can also resume jobs stopped by the LSF administrator or queue administrator if the job belongs to a member of their user group.

Optionally, you can specify additional user group administrator rights for each user group administrator.

User group administrator rights are inherited. For example, if *admin2* has full rights for user group *ugA* and user group *ugB* is a member of *ugA*, *admin2* also has full rights for user group *ugB*.

---

### Restriction:

Unlike a job owner, a user group administrator cannot run *brestart* and *bread -a data\_file*.

---

To manage security concerns, you cannot specify user group administrators for any user group containing the keyword *all* as a member unless **STRICT\_UG\_CONTROL=Y** is defined in *lsb.params*.

## Syntax

*(user\_name | user\_name[admin\_rights] | user\_group | user\_group[admin\_rights] ...)*

Enclose the entire group administrator list in parentheses. If you specify administrator rights for a user or group, enclose them in square brackets.

You can combine user names and user group names in the same list. Use space to separate multiple names.

## Valid values

- *user\_name*

User names must be valid login names.

To specify a Windows user account, include the domain name in uppercase letters (*DOMAIN\_NAME \user\_name*).

- *user\_group*

User group names can be LSF user groups defined previously in this section, or UNIX and Windows user groups.

If you specify a name that is both a UNIX user group and also a UNIX user, append a backslash to make sure it is interpreted as a group (*user\_group/*).

To specify a Windows user group, include the domain name in uppercase letters (*DOMAIN\_NAME \user\_group*).

- *admin\_rights*

- If no rights are specified, only default job control rights are given to user group administrators.
- `usershares`: user group administrators with `usershares` rights can adjust user shares using `bconf update`.
- `full`: user group administrators with `full` rights can use `bconf` to adjust both `usershares` and group members, delete the user group, and create new user groups.

User group administrators with `full` rights can only add a user group member to the user group if they also have `full` rights for the member user group.

User group administrators adding a new user group with `bconf create` are automatically added to `GROUP_ADMIN` with `full` rights for the new user group.

## Restrictions

- Wildcard and special characters are not supported (for example: \*, !, \$, #, &, ~)
- The reserved keywords `others`, `default`, `allremote` are not supported.
- User groups with the keyword `all` as a member can only have user group administrators configured if **STRICT\_UG\_CONTROL=Y** is defined in `lsb.params`.
- User groups with the keyword `all` as a member cannot be user group administrators.
- User groups and user groups administrator definitions cannot be recursive or create a loop.

## USER\_SHARES

Optional. Enables hierarchical fairshare and defines a share tree for users and user groups.

By default, when resources are assigned collectively to a group, the group members compete for the resources according to FCFS scheduling. You can use hierarchical fairshare to further divide the shares among the group members.

## Syntax

(*user, number\_shares*)

Specify the arguments as follows:

- Enclose the list in parentheses, even if you do not specify any user share assignments.

- Enclose each user share assignment in square brackets, as shown.
- Separate the list of share assignments with a space.
- *user*—Specify users or user groups. You can assign the shares to:
  - A single user (specify *user\_name*). To specify a Windows user account, include the domain name in uppercase letters (*DOMAIN\_NAME\user\_name*).
  - Users in a group, individually (specify *group\_name@*) or collectively (specify *group\_name*). To specify a Windows user group, include the domain name in uppercase letters (*DOMAIN\_NAME\group\_name*).
  - Users not included in any other share assignment, individually (specify the keyword *default* or *default@*) or collectively (specify the keyword *others*).

---

**Note:**

By default, when resources are assigned collectively to a group, the group members compete for the resources on a first-come, first-served (FCFS) basis. You can use hierarchical fairshare to further divide the shares among the group members. When resources are assigned to members of a group individually, the share assignment is recursive. Members of the group and of all subgroups always compete for the resources according to FCFS scheduling, regardless of hierarchical fairshare policies.

- *number\_shares*—Specify a positive integer representing the number of shares of the cluster resources assigned to the user. The number of shares assigned to each user is only meaningful when you compare it to the shares assigned to other users or to the total number of shares. The total number of shares is just the sum of all the shares assigned in each share assignment.

## User section

Optional. If this section is not defined, all users and user groups can run an unlimited number of jobs in the cluster.

This section defines the maximum number of jobs a user or user group can run concurrently in the cluster. This is to avoid situations in which a user occupies all or most of the system resources while other users' jobs are waiting.

## Structure

One field is mandatory: `USER_NAME`.

`MAX_JOBS`, `JL/P`, and `MAX_PEND_JOBS` are optional.

You must specify a dash (-) to indicate the default value (unlimited) if a user or user group is specified. Fields cannot be left blank.

## Example of a User section

```

Begin User
USER_NAME  MAX_JOBS  JL/P  MAX_PEND_JOBS
user1      10        -      1000
user2      4         -      -
user3      -         -      -
groupA     10        1      100000
groupA@    -         1      100
groupC     -         -      500
defaul t   6         1      10
End User

```

### USER\_NAME

User or user group for which job slot limits are defined.

Use the reserved user name `defaul t` to specify a job slot limit that applies to each user and user group not explicitly named. Since the limit specified with the keyword `defaul t` applies to user groups also, make sure you select a limit that is high enough, or explicitly define limits for user groups.

User group names can be the LSF user groups defined previously, and/or UNIX and Windows user groups. To specify a Windows user account or user group, include the domain name in uppercase letters (*DOMAIN\_NAME\user\_name* or *DOMAIN\_NAME\user\_group*).

Job slot limits apply to a group as a whole. Append the at sign (@) to a group name to make the job slot limits apply individually to each user in the group. If a group contains a subgroup, the job slot limit also applies to each member in the subgroup recursively.

If the group contains the keyword `al l` in the user list, the at sign (@) has no effect. To specify job slot limits for each user in a user group containing `al l`, use the keyword `defaul t`.

### MAX\_JOBS

Per-user or per-group job slot limit for the cluster. Total number of job slots that each user or user group can use in the cluster.

---

#### Note:

If a group contains the keyword `al l` as a member, all users and user groups are included in the group. The per-group job slot limit set for the group applies to the group as a whole, limiting the entire cluster even when `ENFORCE_ONE_UG_LIMIT` is set in `lsb.params`.

---

### JL/P

Per processor job slot limit per user or user group.

Total number of job slots that each user or user group can use per processor. This job slot limit is configured per processor so that multiprocessor hosts will automatically run more jobs.

This number can be a fraction such as 0.5, so that it can also serve as a per-host limit. This number is rounded up to the nearest integer equal to or greater than the total job slot limits for a host. For example, if JL/P is 0.5, on a 4-CPU multiprocessor host, the user can only use up to 2 job slots at any time. On a uniprocessor machine, the user can use 1 job slot.

## MAX\_PEND\_JOBS

Per-user or per-group pending job limit. This is the total number of pending job slots that each user or user group can have in the system. If a user is a member of multiple user groups, the user's pending jobs are counted towards the pending job limits of all groups from which the user has membership.

If `ENFORCE_ONE_UG_LIMITS` is set to `Y` in `lsb.params` and you submit a job while specifying a user group, only the limits for that user group (or any parent user group) apply to the job even if there are overlapping user group members.

## UserMap section

Optional. Used only in a MultiCluster environment with a non-uniform user name space. Defines system-level cross-cluster account mapping for users and user groups, which allows users to submit a job from a local host and run the job as a different user on a remote host. Both the local and remote clusters must have corresponding user account mappings configured.

## Structure

The following three fields are all required:

- LOCAL
- REMOTE
- DIRECTION

### LOCAL

A list of users or user groups in the local cluster. To specify a Windows user account or user group, include the domain name in uppercase letters (*DOMAIN\_NAME* \user\_name or *DOMAIN\_NAME*\user\_group). Separate multiple user names by a space and enclose the list in parentheses ( ):

```
(user4 user6)
```

### REMOTE

A list of remote users or user groups in the form *user\_name@cluster\_name* or *user\_group@cluster\_name*. To specify a Windows user account or user group, include the domain name in uppercase letters (*DOMAIN\_NAME*\user\_name@cluster\_name or *DOMAIN\_NAME*\user\_group@cluster\_name). Separate multiple user names by a space and enclose the list in parentheses ( ):

```
(user4@cluster2 user6@cluster2)
```

### DIRECTION

Specifies whether the user account runs jobs locally or remotely. Both directions must be configured on the local and remote clusters.

- The `export` keyword configures local users/groups to run jobs as remote users/groups.
- The `import` keyword configures remote users/groups to run jobs as local users/groups.

## Example of a UserMap section

On cluster1:

```
Begin UserMap
```

LOCAL	REMOTE	DIRECTION
user1	user2@cluster2	export
user3	user6@cluster2	export

```
End UserMap
```

On cluster2:

```
Begin UserMap
```

LOCAL	REMOTE	DIRECTION
user2	user1@cluster1	import
user6	user3@cluster1	import

```
End UserMap
```

Cluster1 configures user 1 to run jobs as user2 and user3 to run jobs as user6.

Cluster2 configures user 1 to run jobs as user2 and user3 to run jobs as user6.

## Automatic time-based configuration

Variable configuration is used to automatically change LSF configuration based on time windows. You define automatic configuration changes in `lsb.users` by using if-else constructs and time expressions. After you change the files, reconfigure the cluster with the `badminton reconfig` command.

The expressions are evaluated by LSF every 10 minutes based on `mbatchd` start time. When an expression evaluates true, LSF dynamically changes the configuration based on the associated configuration statements. Reconfiguration is done in real time without restarting `mbatchd`, providing continuous system availability.

### Example

From 12 - 1 p.m. daily, user `smith` has 10 job slots, but during other hours, user has only 5 job slots.

```
Begin User
```

USER_NAME	MAX_JOBS	JL/P
#if time (12-13)		
smith	10	-
#else		
smith	5	-
default	1	-
#endif		

```
End User
```

## lsf.acct

The `lsf.acct` file is the LSF task log file.

The LSF Remote Execution Server, RES (see `res(8)`), generates a record for each task completion or failure. If the RES task logging is turned on (see `lsadm n(8)`), it appends the record to the task log file `lsf.acct.<host_name>`.

## lsf.acct structure

The task log file is an ASCII file with one task record per line. The fields of each record are separated by blanks. The location of the file is determined by the `LSF_RES_ACCTDIR` variable defined in `lsf.conf`. If this variable is not defined, or the RES cannot access the log directory, the log file is created in `/tmp` instead.

## Fields

The fields in a task record are ordered in the following sequence:

### **pid (%d)**

Process ID for the remote task

### **userName (%s)**

User name of the submitter

### **exitStatus (%d)**

Task exit status

### **dispTime (%ld)**

Dispatch time – time at which the task was dispatched for execution

### **termTime (%ld)**

Completion time – time when task is completed/failed

### **fromHost (%s)**

Submission host name

### **execHost (%s)**

Execution host name

### **cwd (%s)**

Current working directory

### **cmdln (%s)**

Command line of the task

### **lsfRusage**

The following fields contain resource usage information for the job (see `getrusage(2)`). If the value of some field is unavailable (due to job exit or the difference among the

operating systems), -1 will be logged. Times are measured in seconds, and sizes are measured in KB.

**ru\_utime (%f)**

User time used

**ru\_stime (%f)**

System time used

**ru\_maxrss (%f)**

Maximum shared text size

**ru\_ixrss (%f)**

Integral of the shared text size over time (in KB seconds)

**ru\_ismrss (%f)**

Integral of the shared memory size over time (valid only on Ultrix)

**ru\_idrss (%f)**

Integral of the unshared data size over time

**ru\_isrss (%f)**

Integral of the unshared stack size over time

**ru\_minflt (%f)**

Number of page reclaims

**ru\_majflt (%f)**

Number of page faults

**ru\_nswap (%f)**

Number of times the process was swapped out

**ru\_inblock (%f)**

Number of block input operations

**ru\_oublock (%f)**

Number of block output operations

**ru\_ioch (%f)**

Number of characters read and written (valid only on HP-UX)

**ru\_msgsnd (%f)**

Number of System V IPC messages sent

**ru\_msgsgrcv (%f)**

Number of messages received

**ru\_nsignals (%f)**

Number of signals received

**ru\_nvcsw (%f)**

Number of voluntary context switches

**ru\_nivcsw (%f)**

Number of involuntary context switches

**ru\_exutime (%f)**

Exact user time used (valid only on ConvexOS)

# lsf.cluster

## Contents

- About lsf.cluster
- Parameters section
- ClusterAdmins section
- Host section
- ResourceMap section
- RemoteClusters section

## About lsf.cluster

This is the cluster configuration file. There is one for each cluster, called `lsf.cluster.cluster_name`. The `cluster_name` suffix is the name of the cluster defined in the Cluster section of `lsf.shared`. All LSF hosts are listed in this file, along with the list of LSF administrators and the installed LSF features.

The `lsf.cluster.cluster_name` file contains two types of configuration information:

- Cluster definition information — affects all LSF applications. Defines cluster administrators, hosts that make up the cluster, attributes of each individual host such as host type or host model, and resources using the names defined in `lsf.shared`.
- LIM policy information — affects applications that rely on LIM job placement policy. Defines load sharing and job placement policies provided by LIM.

## Changing lsf.cluster configuration

After making any changes to `lsf.cluster.cluster_name`, run the following commands:

- `lsadmin reconfig` to reconfigure LIM
- `badmin mbdrestart` to restart `mbatchd`
- `lsadmin limrestart` to restart LIM (on all changed non-master hosts)

## Location

This file is typically installed in the directory defined by `LSF_ENVDIR`.

## Structure

The `lsf.cluster.cluster_name` file contains the following configuration sections:

- Parameters section
- ClusterAdmins section
- Host section
- ResourceMap section
- RemoteClusters section

## Parameters

- `ADJUST_DURATION`
- `ELIM_ABORT_VALUE`
- `ELIM_POLL_INTERVAL`

- ELIMARGS
- EXINTERVAL
- FLOAT\_CLIENTS
- FLOAT\_CLIENTS\_ADDR\_RANGE
- HOST\_INACTIVITY\_LIMIT
- LSF\_ELIM\_BLOCKTIME
- LSF\_ELIM\_DEBUG
- LSF\_ELIM\_RESTARTS
- LSF\_HOST\_ADDR\_RANGE
- MASTER\_INACTIVITY\_LIMIT
- PROBE\_TIMEOUT
- PRODUCTS
- RETRY\_LIMIT

## ADJUST\_DURATION

### Syntax

ADJUST\_DURATION=*integer*

### Description

Integer reflecting a multiple of EXINTERVAL that controls the time period during which load adjustment is in effect.

The `lsl ace(1)` and `lsl oadadj(1)` commands artificially raise the load on a selected host. This increase in load decays linearly to 0 over time.

### Default

3

## ELIM\_ABORT\_VALUE

### Syntax

ELIM\_ABORT\_VALUE=*integer*

### Description

Integer that triggers an abort for an ELIM.

### Default

97 (triggers abort)

## ELIM\_POLL\_INTERVAL

### Syntax

ELIM\_POLL\_INTERVAL=*seconds*

## Description

Time interval, in seconds, that the LIM samples external load index information. If your `elim` executable is programmed to report values more frequently than every 5 seconds, set the `ELIM_POLL_INTERVAL` so that it samples information at a corresponding rate.

## Valid values

0.001 to 5

## Default

5 seconds

# ELIMARGS

## Syntax

`ELIMARGS=cmd_line_args`

## Description

Specifies command-line arguments required by an `elim` executable on startup. Used only when the external load indices feature is enabled.

## Default

Undefined

# EXINTERVAL

## Syntax

`EXINTERVAL=time_in_seconds`

## Description

Time interval, in seconds, at which the LIM daemons exchange load information

On extremely busy hosts or networks, or in clusters with a large number of hosts, load may interfere with the periodic communication between LIM daemons. Setting `EXINTERVAL` to a longer interval can reduce network load and slightly improve reliability, at the cost of slower reaction to dynamic load changes.

Note that if you define the time interval as less than 5 seconds, LSF automatically resets it to 5 seconds.

## Default

15 seconds

# FLOAT\_CLIENTS

## Syntax

`FLOAT_CLIENTS=number_of_floating_client_licenses`

## Description

Sets the size of your license pool in the cluster

When the master LIM starts, up to *number\_of\_floating\_client\_licenses* will be checked out for use as floating client licenses. If fewer licenses are available than specified by *number\_of\_floating\_client\_licenses*, only the available licenses will be checked out and used.

If `FLOAT_CLIENTS` is not specified in `lsf.cluster.cluster_name` or there is an error in either `license.dat` or in `lsf.cluster.cluster_name`, the floating LSF client license feature is disabled.

---

### Caution:

When the LSF floating client feature is enabled, any host can submit jobs to the cluster. You can limit which hosts can be LSF floating clients with the parameter `FLOAT_CLIENTS_ADDR_RANGE` in `lsf.cluster.cluster_name`.

---

## LSF Floating Client

Although an LSF Floating Client requires a license, `LSF_Float_Client` does not need to be added to the `PRODUCTS` line. `LSF_Float_Client` also cannot be added as a resource for specific hosts already defined in `lsf.cluster.cluster_name`. Should these lines be present, they are ignored by LSF.

## Default

Undefined

# FLOAT\_CLIENTS\_ADDR\_RANGE

## Syntax

`FLOAT_CLIENTS_ADDR_RANGE=IP_address...`

## Description

Optional. IP address or range of addresses of domains from which floating client hosts can submit requests. Multiple ranges can be defined, separated by spaces. The IP address can have either a dotted quad notation (IPv4) or IP Next Generation (IPv6) format. LSF supports both formats; you do not have to map IPv4 addresses to an IPv6 format.

---

### Note:

To use IPv6 addresses, you must define the parameter `LSF_ENABLE_SUPPORT_IPV6` in `lsf.conf`.

---

If the value of `FLOAT_CLIENT_ADDR_RANGE` is undefined, there is no security and any hosts can be LSF floating clients.

If a value is defined, security is enabled. If there is an error in the configuration of this variable, by default, no hosts will be allowed to be LSF floating clients.

When this parameter is defined, client hosts that do not belong to the domain will be denied access.

If a requesting host belongs to an IP address that falls in the specified range, the host will be accepted to become a floating client.

IP addresses are separated by spaces, and considered "OR" alternatives.

If you define `FLOAT_CLIENT_ADDR_RANGE` with:

- No range specified, all IPv4 and IPv6 clients can submit requests.
- Only an IPv4 range specified, only IPv4 clients within the range can submit requests.
- Only an IPv6 range specified, only IPv6 clients within the range can submit requests.
- Both an IPv6 and IPv4 range specified, IPv6 and IPv4 clients within the ranges can submit requests.

The asterisk (\*) character indicates any value is allowed.

The dash (-) character indicates an explicit range of values. For example 1-4 indicates 1,2,3,4 are allowed.

Open ranges such as \*-30, or 10-\*, are allowed.

If a range is specified with fewer fields than an IP address such as 10.161, it is considered as 10.161.\*.\*.

Address ranges are validated at configuration time so they must conform to the required format. If any address range is not in the correct format, no hosts will be accepted as LSF floating clients, and an error message will be logged in the LIM log.

This parameter is limited to 2048 characters.

For IPv6 addresses, the double colon symbol (::) indicates multiple groups of 16-bits of zeros. You can also use (::) to compress leading and trailing zeros in an address filter, as shown in the following example:

```
FLOAT_CLIENTS_ADDR_RANGE=1080::8:800:20fc:*
```

This definition allows hosts with addresses 1080:0:0:0:8:800:20fc:\* (three leading zeros).

You cannot use the double colon (::) more than once within an IP address. You cannot use a zero before or after (::). For example, 1080:0::8:800:20fc:\* is not a valid address.

## Notes

After you configure `FLOAT_CLIENTS_ADDR_RANGE`, check the `lim.log`, `host_name` file to make sure this parameter is correctly set. If this parameter is not set or is wrong, this will be indicated in the log file.

## Examples

```
FLOAT_CLIENTS_ADDR_RANGE=100
```

All IPv4 and IPv6 hosts with a domain address starting with 100 will be allowed access.

- To specify only IPv4 hosts, set the value to **100.\***
- To specify only IPv6 hosts, set the value to **100:\***

```
FLOAT_CLIENTS_ADDR_RANGE=100-110.34.1-10.4-56
```

All client hosts belonging to a domain with an address having the first number between 100 and 110, then 34, then a number between 1 and 10, then, a number between 4 and 56 will be allowed access. Example: 100.34.9.45, 100.34.1.4, 102.34.3.20, etc. No IPv6 hosts are allowed.

```
FLOAT_CLIENTS_ADDR_RANGE=100.172.1.13 100.*.30-54 124.24-*.1.*-34
```

All client hosts belonging to a domain with the address 100.172.1.13 will be allowed access. All client hosts belonging to domains starting with 100, then any number, then a range of 30 to 54 will be allowed access. All client hosts belonging to domains starting with 124, then from 24 onward, then 1, then from 0 to 34 will be allowed access. No IPv6 hosts are allowed.

```
FLOAT_CLIENTS_ADDR_RANGE=12.23.45.*
```

All client hosts belonging to domains starting with 12.23.45 are allowed. No IPv6 hosts are allowed.

```
FLOAT_CLIENTS_ADDR_RANGE=100.*43
```

The \* character can only be used to indicate any value. In this example, an error will be inserted in the LIM log and no hosts will be accepted to become LSF floating clients. No IPv6 hosts are allowed.

```
FLOAT_CLIENTS_ADDR_RANGE=100.*43 100.172.1.13
```

Although one correct address range is specified, because \*43 is not correct format, the entire line is considered not valid. An error will be inserted in the LIM log and no hosts will be accepted to become LSF floating clients. No IPv6 hosts are allowed.

```
FLOAT_CLIENTS_ADDR_RANGE = 3ffe
```

All client IPv6 hosts with a domain address starting with 3ffe will be allowed access. No IPv4 hosts are allowed.

```
FLOAT_CLIENTS_ADDR_RANGE = 3ffe:ffe::88bb:*
```

Expands to 3ffe:ffe:0:0:0:88bb:\*. All IPv6 client hosts belonging to domains starting with 3ffe:ffe::88bb:\* are allowed. No IPv4 hosts are allowed.

```
FLOAT_CLIENTS_ADDR_RANGE = 3ffe-4fff:ffe::88bb:aa-ff 12.23.45.*
```

All IPv6 client hosts belonging to domains starting with 3ffe up to 4fff, then ffe::88bb, and ending with aa up to ff are allowed. All IPv4 client hosts belonging to domains starting with 12.23.45 are allowed.

```
FLOAT_CLIENTS_ADDR_RANGE = 3ffe-*:ffe::88bb:*-ff
```

All IPv6 client hosts belonging to domains starting with 3ffe up to ffff and ending with 0 up to ff are allowed. No IPv4 hosts are allowed.

## Default

Undefined. No security is enabled. Any host in any domain is allowed access to LSF floating client licenses.

## See also

LSF\_ENABLE\_SUPPORT\_IPV6

# HOST\_INACTIVITY\_LIMIT

## Syntax

```
HOST_INACTIVITY_LIMIT=integer
```

## Description

Integer that is multiplied by EXINTERVAL, the time period you set for the communication between the master and slave LIMs to ensure all parties are functioning.

A slave LIM can send its load information any time from EXINTERVAL to (HOST\_INACTIVITY\_LIMIT-1)\*EXINTERVAL seconds. A master LIM sends a master announce to each host at least every EXINTERVAL\*HOST\_INACTIVITY\_LIMIT seconds.

The HOST\_INACTIVITY\_LIMIT must be greater than or equal to 2.

Increase or decrease the host inactivity limit to adjust for your tolerance for communication between master and slaves. For example, if you have hosts that frequently become inactive, decrease the host inactivity limit. Note that to get the right interval, you may also have to adjust your EXINTERVAL.

## Default

5

# LSF\_ELIM\_BLOCKTIME

## Syntax

```
LSF_ELIM_BLOCKTIME=seconds
```

## Description

UNIX only; used when the external load indices feature is enabled.

Maximum amount of time the master external load information manager (MELIM) waits for a complete load update string from an `elim` executable. After the time period specified by `LSF_ELIM_BLOCKTIME`, the MELIM writes the last string sent by an `elim` in the LIM log file (`lim.log. host_name`) and restarts the `elim`.

Defining `LSF_ELIM_BLOCKTIME` also triggers the MELIM to restart `elim` executables if the `elim` does not write a complete load update string within the time specified for `LSF_ELIM_BLOCKTIME`.

## Valid values

Non-negative integers. For example, if your `elim` writes name-value pairs with 1 second intervals between them, and your `elim` reports 12 load indices, allow at least 12 seconds for the `elim` to finish writing the entire load update string. In this case, define `LSF_ELIM_BLOCKTIME` as 15 seconds or more.

A value of 0 indicates that the MELIM expects to receive the entire load string all at once.

If you comment out or delete `LSF_ELIM_BLOCKTIME`, the MELIM waits 2 seconds for a complete load update string.

## Default

4 seconds

## See also

`LSF_ELIM_RESTARTS` to limit how many times the ELIM can be restarted.

# LSF\_ELIM\_DEBUG

## Syntax

```
LSF_ELIM_DEBUG=y
```

## Description

UNIX only; used when the external load indices feature is enabled.

When this parameter is set to `y`, all external load information received by the load information manager (LIM) from the master external load information manager (MELIM) is logged in the LIM log file (`lim.log. host_name`).

Defining `LSF_ELIM_DEBUG` also triggers the MELIM to restart `elim` executables if the `elim` does not write a complete load update string within the time specified for `LSF_ELIM_BLOCKTIME`.

## Default

Undefined; external load information sent by an to the MELIM is not logged.

## See also

LSF\_ELIM\_BLOCKTIME to configure how long LIM waits before restarting the ELIM.

LSF\_ELIM\_RESTARTS to limit how many times the ELIM can be restarted.

# LSF\_ELIM\_RESTARTS

## Syntax

LSF\_ELIM\_RESTARTS=*integer*

## Description

UNIX only; used when the external load indices feature is enabled.

Maximum number of times the master external load information manager (MELIM) can restart `el i m` executables on a host. Defining this parameter prevents an ongoing restart loop in the case of a faulty `el i m`. The MELIM waits the LSF\_ELIM\_BLOCKTIME to receive a complete load update string before restarting the `el i m`. The MELIM does not restart any `el i m` executables that exit with ELIM\_ABORT\_VALUE.

---

### Important:

Either LSF\_ELIM\_BLOCKTIME or LSF\_ELIM\_DEBUG must also be defined; defining these parameters triggers the MELIM to restart `el i m` executables.

---

## Valid values

Non-negative integers.

## Default

Undefined; the number of `el i m` restarts is unlimited.

## See also

LSF\_ELIM\_BLOCKTIME, LSF\_ELIM\_DEBUG

# LSF\_HOST\_ADDR\_RANGE

## Syntax

LSF\_HOST\_ADDR\_RANGE=*IP\_address ...*

## Description

Identifies the range of IP addresses that are allowed to be LSF hosts that can be dynamically added to or removed from the cluster.

---

### Caution:

To enable dynamically added hosts after installation, you must define `LSF_HOST_ADDR_RANGE` in `lsf.cluster.cluster_name`, and `LSF_DYNAMIC_HOST_WAIT_TIME` in `lsf.conf`. If you enable dynamic hosts during installation, you must define an IP address range after installation to enable security.

If a value is defined, security for dynamically adding and removing hosts is enabled, and only hosts with IP addresses within the specified range can be added to or removed from a cluster dynamically.

Specify an IP address or range of addresses, using either a dotted quad notation (IPv4) or IP Next Generation (IPv6) format. LSF supports both formats; you do not have to map IPv4 addresses to an IPv6 format. Multiple ranges can be defined, separated by spaces.

---

### Note:

To use IPv6 addresses, you must define the parameter `LSF_ENABLE_SUPPORT_IPV6` in `lsf.conf`.

If there is an error in the configuration of `LSF_HOST_ADDR_RANGE` (for example, an address range is not in the correct format), no host will be allowed to join the cluster dynamically and an error message will be logged in the LIM log. Address ranges are validated at startup, reconfiguration, or restart, so they must conform to the required format.

If a requesting host belongs to an IP address that falls in the specified range, the host will be accepted to become a dynamic LSF host.

IP addresses are separated by spaces, and considered "OR" alternatives.

If you define the parameter `LSF_HOST_ADDR_RANGE` with:

- No range specified, all IPv4 and IPv6 clients are allowed.
- Only an IPv4 range specified, only IPv4 clients within the range are allowed.
- Only an IPv6 range specified, only IPv6 clients within the range are allowed.
- Both an IPv6 and IPv4 range specified, IPv6 and IPv4 clients within the ranges are allowed.

The asterisk (\*) character indicates any value is allowed.

The dash (-) character indicates an explicit range of values. For example 1-4 indicates 1,2,3,4 are allowed.

Open ranges such as \*-30, or 10-\*, are allowed.

For IPv6 addresses, the double colon symbol (::) indicates multiple groups of 16-bits of zeros. You can also use (::) to compress leading and trailing zeros in an address filter, as shown in the following example:

**LSF\_HOST\_ADDR\_RANGE=1080::8:800:20fc:\***

This definition allows hosts with addresses 1080:0:0:0:8:800:20fc:\* (three leading zeros).

You cannot use the double colon (::) more than once within an IP address. You cannot use a zero before or after (::). For example, 1080:0::8:800:20fc:\* is not a valid address.

If a range is specified with fewer fields than an IP address such as 10.161, it is considered as 10.161.\*.\*.

This parameter is limited to 2048 characters.

## Notes

After you configure `LSF_HOST_ADDR_RANGE`, check the `lim.log` *host\_name* file to make sure this parameter is correctly set. If this parameter is not set or is wrong, this will be indicated in the log file.

## Examples

```
LSF_HOST_ADDR_RANGE=100
```

All IPv4 and IPv6 hosts with a domain address starting with 100 will be allowed access.

- To specify only IPv4 hosts, set the value to **100.\***
- To specify only IPv6 hosts, set the value to **100:\***

```
LSF_HOST_ADDR_RANGE=100- 110. 34. 1- 10. 4- 56
```

All hosts belonging to a domain with an address having the first number between 100 and 110, then 34, then a number between 1 and 10, then, a number between 4 and 56 will be allowed access. No IPv6 hosts are allowed. Example: 100.34.9.45, 100.34.1.4, 102.34.3.20, etc.

```
LSF_HOST_ADDR_RANGE=100. 172. 1. 13 100. *. 30- 54 124. 24- *. 1. *- 34
```

The host with the address 100.172.1.13 will be allowed access. All hosts belonging to domains starting with 100, then any number, then a range of 30 to 54 will be allowed access. All hosts belonging to domains starting with 124, then from 24 onward, then 1, then from 0 to 34 will be allowed access. No IPv6 hosts are allowed.

```
LSF_HOST_ADDR_RANGE=12. 23. 45. *
```

All hosts belonging to domains starting with 12.23.45 are allowed. No IPv6 hosts are allowed.

```
LSF_HOST_ADDR_RANGE=100. *43
```

The \* character can only be used to indicate any value. The format of this example is not correct, and an error will be inserted in the LIM log and no hosts will be able to join the cluster dynamically. No IPv6 hosts are allowed.

```
LSF_HOST_ADDR_RANGE=100. *43 100. 172. 1. 13
```

Although one correct address range is specified, because \*43 is not correct format, the entire line is considered not valid. An error will be inserted in the LIM log and no hosts will be able to join the cluster dynamically. No IPv6 hosts are allowed.

```
LSF_HOST_ADDR_RANGE = 3ffe
```

All client IPv6 hosts with a domain address starting with 3ffe will be allowed access. No IPv4 hosts are allowed.

```
LSF_HOST_ADDR_RANGE = 3ffe: fffe: : 88bb: *
```

Expands to 3ffe:ffe:0:0:0:88bb:\*. All IPv6 client hosts belonging to domains starting with 3ffe:ffe::88bb:\* are allowed. No IPv4 hosts are allowed.

```
LSF_HOST_ADDR_RANGE = 3ffe- 4fff: fffe: : 88bb: aa- ff 12. 23. 45. *
```

All IPv6 client hosts belonging to domains starting with 3ffe up to 4fff, then fffe::88bb, and ending with aa up to ff are allowed. IPv4 client hosts belonging to domains starting with 12.23.45 are allowed.

```
LSF_HOST_ADDR_RANGE = 3ffe- *: fffe: : 88bb: *- ff
```

All IPv6 client hosts belonging to domains starting with 3ffe up to ffff and ending with 0 up to ff are allowed. No IPv4 hosts are allowed.

## Default

Undefined (dynamic host feature disabled). If you enable dynamic hosts during installation, no security is enabled and all hosts can join the cluster.

## See also

LSF\_ENABLE\_SUPPORT\_IPV6

# MASTER\_INACTIVITY\_LIMIT

## Syntax

MASTER\_INACTIVITY\_LIMIT=*integer*

## Description

An integer reflecting a multiple of EXINTERVAL. A slave will attempt to become master if it does not hear from the previous master after (HOST\_INACTIVITY\_LIMIT + *host\_number*\*MASTER\_INACTIVITY\_LIMIT)\*EXINTERVAL seconds, where *host\_number* is the position of the host in `lsf.cluster.cluster_name`.

The master host is *host\_number* 0.

## Default

2

# PROBE\_TIMEOUT

## Syntax

PROBE\_TIMEOUT=*time\_in\_seconds*

## Description

Specifies the timeout in seconds to be used for the `connect(2)` system call

Before taking over as the master, a slave LIM will try to connect to the last known master via TCP.

## Default

2 seconds

# PRODUCTS

## Syntax

PRODUCTS=*product\_keyword* ...

## Description

Specifies the LSF products that the cluster will run (you must also have a license for each product). The list of items is separated by space.

The PRODUCTS parameter is set automatically during LSF installation to include LSF\_Base and LSF\_Manager, which are both required to run Platform LSF. Specify additional product keywords if your cluster is fully licensed for the corresponding products.

For partially licensed products, do not include the product keyword in this parameter, configure the RESOURCES parameter in the Hosts section of this file instead.

## Valid Values

LSF\_Base, LSF\_Manager, LSF\_MultiCluster, LSF\_Make, LSF\_Session\_Scheduler

## Default

LSF\_Base LSF\_Manager

# RETRY\_LIMIT

## Syntax

RETRY\_LIMIT=*integer*

## Description

Integer reflecting a multiple of EXINTERVAL that controls the number of retries a master or slave LIM makes before assuming that the slave or master is unavailable.

If the master does not hear from a slave for HOST\_INACTIVITY\_LIMIT exchange intervals, it will actively poll the slave for RETRY\_LIMIT exchange intervals before it will declare the slave as unavailable. If a slave does not hear from the master for HOST\_INACTIVITY\_LIMIT exchange intervals, it will actively poll the master for RETRY\_LIMIT intervals before assuming that the master is down.

## Default

2

## ClusterAdmins section

(Optional) The ClusterAdmins section defines the LSF administrators for the cluster. The only keyword is ADMINISTRATORS.

If the ClusterAdmins section is not present, the default LSF administrator is root. Using root as the primary LSF administrator is not recommended.

## ADMINISTRATORS

### Syntax

ADMINISTRATORS=*administrator\_name ...*

### Description

Specify UNIX user names.

You can also specify UNIX user group names, Windows user names, and Windows user group names. To specify a Windows user account or user group, include the domain name in uppercase letters (*DOMAIN\_NAME\user\_name* or *DOMAIN\_NAME\user\_group*).

The first administrator of the expanded list is considered the primary LSF administrator. The primary administrator is the owner of the LSF configuration files, as well as the working files under *LSB\_SHARED1R/cluster\_name*. If the primary administrator is changed, make sure the owner of the configuration files and the files under *LSB\_SHARED1R/cluster\_name* are changed as well.

Administrators other than the primary LSF administrator have the same privileges as the primary LSF administrator except that they do not have permission to change LSF configuration files. They can perform clusterwide operations on jobs, queues, or hosts in the system.

For flexibility, each cluster may have its own LSF administrators, identified by a user name, although the same administrators can be responsible for several clusters.

Use the `-l` option of the `lsc clusters` command to display all of the administrators within a cluster.

Windows domain:

- If the specified user or user group is a domain administrator, member of the `Power Users` group or a group with domain administrative privileges, the specified user or user group must belong to the LSF user domain.
- If the specified user or user group is a user or user group with a lower degree of privileges than outlined in the previous point, the user or user group must belong to the LSF user domain and be part of the `Global Admins` group.

Windows workgroup

- If the specified user or user group is not a workgroup administrator, member of the `Power Users` group, or a group with administrative privileges on each host, the specified user or user group must belong to the `Local Admins` group on each host.

## Compatibility

For backwards compatibility, `ClusterManager` and `Manager` are synonyms for `ClusterAdmins` and `ADMINISTRATORS` respectively. It is possible to have both sections present in the same `lsf.cluster.cluster_name` file to allow daemons from different LSF versions to share the same file.

## Example

The following gives an example of a cluster with two LSF administrators. The user listed first, `user2`, is the primary administrator.

```
Begin ClusterAdmins
ADMINISTRATORS = user2 user7
End ClusterAdmins
```

## Default

lsfadmin

## Host section

The Host section is the last section in `lsf.cluster.cluster_name` and is the only required section. It lists all the hosts in the cluster and gives configuration information for each host.

The order in which the hosts are listed in this section is important, because the first host listed becomes the LSF master host. Since the master LIM makes all placement decisions for the cluster, it should be on a fast machine.

The LIM on the first host listed becomes the master LIM if this host is up; otherwise, that on the second becomes the master if its host is up, and so on. Also, to avoid the delays involved in switching masters if the first machine goes down, the master should be on a reliable machine. It is desirable to arrange the list such that the first few hosts in the list are always in the same subnet. This avoids a situation where the second host takes over as master when there are communication problems between subnets.

Configuration information is of two types:

- Some fields in a host entry simply describe the machine and its configuration.
- Other fields set thresholds for various resources.

## Example Host section

This example Host section contains descriptive and threshold information for three hosts:

```

Begin Host
HOSTNAME  model    type  server  r1m  pg  tmp  RESOURCES          RUNWI NDOW
hostA     SparcIPC Sparc  1      3.5 15  0  (sunos frame)     ()
hostD     Sparc10  Sparc  1      3.5 15  0  (sunos)           (5: 18: 30- 1: 8: 30)
hostD     !         !      1      2.0 10  0  ()                ()
hostE     !         !      1      2.0 10  0  (linux !bigmem)  ()
End Host

```

## Descriptive fields

The following fields are required in the Host section:

- HOSTNAME
- RESOURCES
- type
- model

The following fields are optional:

- server
- nd
- RUNWINDOW
- REXPRI

## HOSTNAME

### Description

Official name of the host as returned by `hostname(1)`

The name must be listed in `lsf.shared` as belonging to this cluster.

## model

### Description

Host model

The name must be defined in the HostModel section of `lsf.shared`. This determines the CPU speed scaling factor applied in load and placement calculations.

Optionally, the `!` keyword for the model or type column, indicates that the host model or type is to be automatically detected by the LIM running on the host.

## nd

### Description

Number of local disks

This corresponds to the `ndisks` static resource. On most host types, LSF automatically determines the number of disks, and the `nd` parameter is ignored.

`nd` should only count local disks with file systems on them. Do not count either disks used only for swapping or disks mounted with NFS.

## Default

The number of disks determined by the LIM, or 1 if the LIM cannot determine this

# RESOURCES

## Description

The static Boolean resources and static or dynamic numeric and string resources available on this host. The keywords `LSF_Make` and `LSF_SessionScheduler` are also allowed, to support partial licensing.

The resource names are strings defined in the Resource section of `lsf.shared`. You may list any number of resources, enclosed in parentheses and separated by blanks or tabs. For example:

```
(fs frame hpux)
```

Optionally, you can specify an exclusive resource by prefixing the resource with an exclamation mark (!). For example, resource `bigmem` is defined in `lsf.shared`, and is defined as an exclusive resource for hostE:

```
Begin Host
HOSTNAME  model  type  server rlm pg tmp RESOURCES          RUNWINDOW
...
hostE     !      !      1      2.0 10   0 (linux !bigmem)  ()
...
End Host
```

Square brackets are not valid and the resource name must be alphanumeric.

You must explicitly specify the exclusive resources in the resource requirements for the job to select a host with an exclusive resource for a job. For example:

```
bsub -R "bigmem" myjob
```

or

```
bsub -R "defined(bigmem)" myjob
```

You can specify static and dynamic numeric and string resources in the resource column of the Host clause. For example:

```
Begin Host
HOSTNAME  model  type  server rlm  mem  swp RESOURCES  #Keywords
hostA     !      !      1      3.5  ()   ()  (mg elimres patchrev=3 owner=user1)
hostB     !      !      1      3.5  ()   ()  (specman=5 switch=1 owner=test)
hostC     !      !      1      3.5  ()   ()  (switch=2 rack=rack2_2_3 owner=test)
hostD     !      !      1      3.5  ()   ()  (switch=1 rack=rack2_2_3 owner=test)
End Host
```

Static resource information is displayed by `lshosts`, with exclusive resources prefixed by '!'.

## REXPRI

### Description

UNIX only

Default execution priority for interactive remote jobs run under the RES

The range is from -20 to 20. REXPRI corresponds to the BSD-style nice value used for remote jobs. For hosts with System V-style nice values with the range 0 - 39, a REXPRI of -20 corresponds to a nice value of 0, and +20 corresponds to 39. Higher values of REXPRI correspond to lower execution priority; -20 gives the highest priority, 0 is the default priority for login sessions, and +20 is the lowest priority.

### Default

0

## RUNWINDOW

### Description

Dispatch window for interactive tasks.

When the host is not available for remote execution, the host status is `lockW` (locked by run window). LIM does not schedule interactive tasks on hosts locked by dispatch windows. Run windows only apply to interactive tasks placed by LIM. The LSF batch system uses its own (optional) host dispatch windows to control batch job processing on batch server hosts.

### Format

A dispatch window consists of one or more time windows in the format *begin\_time-end\_time*. No blanks can separate *begin\_time* and *end\_time*. Time is specified in the form *[day:]hour[:minute]*. If only one field is specified, LSF assumes it is an *hour*. Two fields are assumed to be *hour.minute*. Use blanks to separate time windows.

### Default

Always accept remote jobs

## server

### Description

Indicates whether the host can receive jobs from other hosts

Specify 1 if the host can receive jobs from other hosts; specify 0 otherwise. Servers that are set to 0 are LSF clients. Client hosts do not run the LSF daemons. Client hosts can submit interactive and batch jobs to the cluster, but they cannot execute jobs sent from other hosts.

### Default

1

## type

### Description

Host type as defined in the HostType section of `lsf.shared`

The strings used for host types are determined by the system administrator: for example, SUNSOL, DEC, or HPPA. The host type is used to identify binary-compatible hosts.

The host type is used as the default resource requirement. That is, if no resource requirement is specified in a placement request, the task is run on a host of the same type as the sending host.

Often one host type can be used for many machine models. For example, the host type name SUNSOL6 might be used for any computer with a SPARC processor running SunOS 6. This would include many Sun models and quite a few from other vendors as well.

Optionally, the `!` keyword for the model or type column, indicates that the host model or type is to be automatically detected by the LIM running on the host.

### Threshold fields

The LIM uses these thresholds in determining whether to place remote jobs on a host. If one or more LSF load indices exceeds the corresponding threshold (too many users, not enough swap space, etc.), then the host is regarded as busy, and LIM will not recommend jobs to that host.

The CPU run queue length threshold values (`r15s`, `r1m`, and `r15m`) are taken as effective queue lengths as reported by `lsload -E`.

All of these fields are optional; you only need to configure thresholds for load indices that you wish to use for determining whether hosts are busy. Fields that are not configured are not considered when determining host status. The keywords for the threshold fields are not case sensitive.

Thresholds can be set for any of the following:

- The built-in LSF load indexes (`r15s`, `r1m`, `r15m`, `ut`, `pg`, `it`, `io`, `ls`, `swp`, `mem`, `tmp`)
- External load indexes defined in the Resource section of `lsf.shared`

## ResourceMap section

The ResourceMap section defines shared resources in your cluster. This section specifies the mapping between shared resources and their sharing hosts. When you define resources in the Resources section of `lsf.shared`, there is no distinction between a shared and non-shared resource. By default, all resources are not shared and are local to each host. By defining the ResourceMap section, you can define resources that are shared by all hosts in the cluster or define resources that are shared by only some of the hosts in the cluster.

This section must appear after the Host section of `lsf.cluster`. *cluster\_name*, because it has a dependency on host names defined in the Host section.

### ResourceMap section structure

The first line consists of the keywords RESOURCENAME and LOCATION. Subsequent lines describe the hosts that are associated with each configured resource.

## Example ResourceMap section

```
Begin ResourceMap
RESOURCENAME LOCATION
verilog          (5@[all])
local            ([host1 host2] [others])
End ResourceMap
```

The resource `verilog` must already be defined in the `RESOURCE` section of the `lsf.shared` file. It is a static numeric resource shared by all hosts. The value for `verilog` is 5. The resource `local` is a numeric shared resource that contains two instances in the cluster. The first instance is shared by two machines, `host1` and `host2`. The second instance is shared by all other hosts.

Resources defined in the `ResourceMap` section can be viewed by using the `-s` option of the `lshosts` (for static resource) and `lslload` (for dynamic resource) commands.

## LOCATION

### Description

Defines the hosts that share the resource

For a static resource, you must define an initial value here as well. Do not define a value for a dynamic resource.

*instance* is a list of host names that share an instance of the resource. The reserved words `all`, `others`, and `default` can be specified for the instance:

`all` — Indicates that there is only one instance of the resource in the whole cluster and that this resource is shared by all of the hosts

Use the not operator (`~`) to exclude hosts from the `all` specification. For example:

```
(2@[all ~host3 ~host4])
```

means that 2 units of the resource are shared by all server hosts in the cluster made up of `host1`, `host2`, . . . , `hostn`, except for `host3` and `host4`. This is useful if you have a large cluster but only want to exclude a few hosts.

The parentheses are required in the specification. The not operator can only be used with the `all` keyword. It is not valid with the keywords `others` and `default`.

`others` — Indicates that the rest of the server hosts not explicitly listed in the `LOCATION` field comprise one instance of the resource

For example:

```
2@[host1] 4@[others]
```

indicates that there are 2 units of the resource on `host1` and 4 units of the resource shared by all other hosts.

`default` — Indicates an instance of a resource on each host in the cluster

This specifies a special case where the resource is in effect not shared and is local to every host.

`default` means at each host. Normally, you should not need to use `default`, because by default all resources are local to each host. You might want to use `ResourceMap` for a non-shared static resource if you need to specify different values for the resource on different hosts.

## RESOURCENAME

### Description

Name of the resource

This resource name must be defined in the Resource section of `lsf.shared`. You must specify at least a name and description for the resource, using the keywords `RESOURCENAME` and `DESCRIPTION`.

- A resource name cannot begin with a number.
- A resource name cannot contain any of the following characters:  
: . ( ) [ + - \* / ! & | < > @ =
- A resource name cannot be any of the following reserved names:  
cpu cpuf io logins ls idle maxmem maxswp maxtmp type model status it  
mem ncpus define\_ncpus\_cores define\_ncpus\_procs  
define\_ncpus\_threads ndisks pg r15m r15s r1m swap swp tmp ut
- To avoid conflict with `inf` and `nan` keywords in 3rd-party libraries, resource names should not begin with `inf` or `nan` (upper case or lower case). Resource requirement strings, such as `-R "infra"` or `-R "nano"` will cause an error. Use `-R "defined(infxx)"` or `-R "defined(nanxx)"`, to specify these resource names.
- Resource names are case sensitive
- Resource names can be up to 39 characters in length

## RemoteClusters section

Optional. This section is used only in a MultiCluster environment. By default, the local cluster can obtain information about all other clusters specified in `lsf.shared`. The `RemoteClusters` section limits the clusters that the local cluster can obtain information about.

The `RemoteClusters` section is required if you want to configure cluster equivalency, cache interval, daemon authentication across clusters, or if you want to run parallel jobs across clusters. To maintain compatibility in this case, make sure the list includes all clusters specified in `lsf.shared`, even if you only configure the default behavior for some of the clusters.

The first line consists of keywords. `CLUSTERNAME` is mandatory and the other parameters are optional.

Subsequent lines configure the remote cluster.

### Example RemoteClusters section

```
Begin RemoteClusters
CLUSTERNAME  EQUIV  CACHE_INTERVAL  RECV_FROM  AUTH
cluster1     Y         60              Y          KRB
cluster2     N         60              Y          -
cluster4     N         60              N          PKI
End RemoteClusters
```

## CLUSTERNAME

### Description

Remote cluster name

Defines the Remote Cluster list. Specify the clusters you want the local cluster will recognize. Recognized clusters must also be defined in `lsf.shared`. Additional clusters listed in `lsf.shared` but not listed here will be ignored by this cluster.

## EQUIV

### Description

Specify 'Y' to make the remote cluster equivalent to the local cluster. Otherwise, specify 'N'. The master LIM considers all equivalent clusters when servicing requests from clients for load, host, or placement information.

EQUIV changes the default behavior of LSF commands and utilities and causes them to automatically return load (`lsload(1)`), host (`lshosts(1)`), or placement (`lsplace(1)`) information about the remote cluster as well as the local cluster, even when you don't specify a cluster name.

## CACHE\_INTERVAL

### Description

Specify the load information cache threshold, in seconds. The host information threshold is twice the value of the load information threshold.

To reduce overhead and avoid updating information from remote clusters unnecessarily, LSF displays information in the cache, unless the information in the cache is older than the threshold value.

### Default

60 (seconds)

## RECV\_FROM

### Description

Specifies whether the local cluster accepts parallel jobs that originate in a remote cluster

RECV\_FROM does not affect regular or interactive batch jobs.

Specify 'Y' if you want to run parallel jobs across clusters. Otherwise, specify 'N'.

### Default

Y

## AUTH

### Description

Defines the preferred authentication method for LSF daemons communicating across clusters. Specify the same method name that is used to identify the corresponding `eauth` program (`eauth.method_name`). If the remote cluster does not prefer the same method, LSF uses default security between the two clusters.

### Default

- (only privileged port (setuid) authentication is used between clusters)

## lsf.*cluster\_name*.license.acct

This is the license accounting file. There is one for each cluster, called `lsf.cluster_name.license.acct`. The *cluster\_name* variable is the name of the cluster defined in the Cluster section of `lsf.shared`.

The `lsf.cluster_name.license.acct` file contains three types of configuration information:

- LSF license information
- MultiCluster license information

## lsf.*cluster\_name*.license.acct structure

The license audit log file is an ASCII file with one record per line. The fields of a record are separated by blanks.

## File properties

### Location

The default location of this file is defined by `LSF_LOGDIR` in `lsf.conf`, but you can override this by defining `LSF_LICENSE_ACCT_PATH` in `lsf.conf`.

### Owner

The primary LSF admin is the owner of this file.

### Permissions

```
-rw-r--r--
```

## Records and fields

The fields of a record are separated by blanks. The fields in order of occurrence are as follows:

### timestamp (%d)

Time stamp of the logged event (in seconds since the epoch).

### type (%s)

The LSF product type. The valid values are as follows:

- `LSF_MANAGER`
- `LSF_MULTICLUSTER`

### version (%s)

The version of the LSF product.

### value (%s)

The actual tracked value. The format of this field depends on the product type as specified by the `type` field:

#### LSF\_MANAGER

```
e e_peak e e_max_avail S s_peak s s_max_avail B b_peak b b_max_avail
```

Where

*e\_peak*, *s\_peak*, and *b\_peak* are the peak usage values (in number of CPUs) of the E, S, and B class licenses, respectively.

*e\_max\_avail*, *s\_max\_avail*, and *b\_max\_avail* are the maximum availability and usage values (in number of CPUs) of the E, S, and B class licenses, respectively. This is determined by the license that you purchased.

#### LSF\_MULTICLUSTER

*mc\_peak mc\_max\_avail*

Where

*mc\_peak* is the peak usage value (in number of CPUs) of the Platform MultiCluster license

*mc\_max\_avail* is the maximum availability and usage (in number of CPUs) of the Platform MultiCluster license. This is determined by the license that you purchased.

#### status (%s)

The results of the license usage check. The valid values are as follows:

#### OK

Peak usage is less than the maximum license availability

#### OVERUSE

Peak usage is more than the maximum license availability

#### hash (%s)

Line encryption used to authenticate the record.

## Example record Format

```
1128372131 LSF_MANAGER 8.0 E hostA OVERUSE 7c7998a6861ea119cd48414a820be18cd641
1128372131 LSF_MULTICLUSTER 8.0 8 10 OK 281288c606a50065ea0e2f3e7161972c56491dc
1128372185 LSF_MANAGER 8.0 E 8 0 S 0 2 B 0 10 OVERUSE fb439ee293821761af9ed0785 1128372185
LSF_MANAGER 8.0 E hostA OVERUSE 2d22a06d6c5cf d5aba40875c2cb8544444a5
```

# lsf.conf

The `lsf.conf` file controls the operation of LSF.

## About lsf.conf

`lsf.conf` is created during installation and records all the settings chosen when LSF was installed. The `lsf.conf` file dictates the location of the specific configuration files and operation of individual servers and applications.

The `lsf.conf` file is used by LSF and applications built on top of it. For example, information in `lsf.conf` is used by LSF daemons and commands to locate other configuration files, executables, and network services. `lsf.conf` is updated, if necessary, when you upgrade to a new version.

This file can also be expanded to include application-specific parameters.

Parameters in this file can also be set as environment variables, except for the parameters related to job packs.

## Corresponding parameters in ego.conf

When Platform EGO is enabled in LSF Version 7, you can configure some LSF parameters in `lsf.conf` that have corresponding Platform EGO parameter names in `EGO_CONFDIR/ego.conf` (`LSF_CONFDIR/lsf.conf` is a separate file from `EGO_CONFDIR/ego.conf`). If both the LSF and the EGO parameters are set in their respective files, the definition in `ego.conf` is used. You must continue to set LSF parameters only in `lsf.conf`.

When EGO is enabled in the LSF cluster (`LSF_ENABLE_EGO=Y`), you also can set the following EGO parameters related to LIM, PIM, and ELIM in either `lsf.conf` or `ego.conf`:

- `EGO_DISABLE_UNRESOLVABLE_HOST` (dynamically added hosts only)
- `EGO_ENABLE_AUTO_DAEMON_SHUTDOWN`
- `EGO_DAEMONS_CPUS`
- `EGO_DEFINE_NCPUS`
- `EGO_SLAVE_CTRL_REMOTE_HOST`
- `EGO_WORKDIR`
- `EGO_PIM_SWAP_REPORT`
- `EGO_ESLIM_TIMEOUT`

If EGO is not enabled, you do not need to set these parameters.

See *Administering Platform LSF* for more information about configuring LSF for EGO. See the *Platform EGO Reference* for information about `ego.conf` parameters.

## Change lsf.conf configuration

Depending on the parameters you change in `lsf.conf`, you may need to run the following commands:

- `lsadm in reconfi g` to reconfigure LIM
- `badmi n mbdrestart` to restart `mbatchd`
- `badmi n hrestart` to restart `sbatchd`

If you have installed LSF in a mixed cluster, you must make sure that `lsf.conf` parameters set on UNIX and Linux match any corresponding parameters in the local `lsf.conf` files on your Windows hosts.

## Location

The default location of `lsf.conf` is in `$LSF_TOP/conf`. This default location can be overridden when necessary by either the environment variable `LSF_ENVDIR` or the command line option `-d` available to some of the applications.

## Format

Each entry in `lsf.conf` has one of the following forms:

```
NAME=VALUE
```

```
NAME=
```

```
NAME="STRING1 STRING2 ..."
```

The equal sign `=` must follow each `NAME` even if no value follows and there should be no space beside the equal sign.

A value that contains multiple strings separated by spaces must be enclosed in quotation marks.

Lines starting with a pound sign (`#`) are comments and are ignored. Do not use `#if` as this is reserved syntax for time-based configuration.

## DAEMON\_SHUTDOWN\_DELAY

### Syntax

```
DAEMON_SHUTDOWN_DELAY=time_in_seconds
```

### Description

Applies when `EGO_ENABLE_AUTO_DAEMON_SHUTDOWN=Y`. Controls amount of time the slave LIM waits to communicate with other (RES and SBD) local daemons before exiting. Used to shorten or lengthen the time interval between a host attempting to join the cluster and, if it was unsuccessful, all of the local daemons shutting down.

The value should not be less than the minimum interval of RES and SBD housekeeping. Most administrators should set this value to somewhere between 3 minutes and 60 minutes.

### Default

1800 seconds (30 minutes)

## EGO\_DEFINE\_NCPUS

### Syntax

```
EGO_DEFINE_NCPUS=procs | cores | threads
```

### Description

If defined, enables an administrator to define a value other than the number of cores available. Follow one of the three equations below for an accurate value.

- `EGO_DEFINE_NCPUS=procs-ncpus=number of processors`
- `EGO_DEFINE_NCPUS=cores-ncpus=number of processors x number of cores`
- `EGO_DEFINE_NCPUS=threads-ncpus=number of processors x number of cores x number of threads.`

**Note:**


---

When `PARALLEL_SCHED_BY_SLOT=Y` in `lsb.params`, the resource requirement string keyword `n_cpus` refers to the number of slots instead of the number of CPUs, however `lshosts` output will continue to show `n_cpus` as defined by `EGO_DEFINE_NCPUS` in `lsf.conf`.

---

## Default

`EGO_DEFINE_NCPUS=cores`

# EGO\_ENABLE\_AUTO\_DAEMON\_SHUTDOWN

## Syntax

`EGO_ENABLE_AUTO_DAEMON_SHUTDOWN="Y" | "N"`

## Description

For hosts that attempted to join the cluster but failed to communicate within the `LSF_DYNAMIC_HOST_WAIT_TIME` period, automatically shuts down any running daemons.

This parameter can be useful if an administrator remove machines from the cluster regularly (by editing `lsf.clusterfile`) or when a host belonging to the cluster is imaged, but the new host should not be part of the cluster. An administrator no longer has to go to each host that is not a part of the cluster to shut down any running daemons.

## Default

N (daemons continue to run on hosts that were not successfully added to the cluster)

# EGO\_PARAMETER

`EGO_ENABLE_AUTO_DAEMON_SHUTDOWN`

# EGO\_ESLIM\_TIMEOUT

## Syntax

`EGO_ESLIM_TIMEOUT=time_seconds`

## Description

Controls how long the LIM waits for any external static LIM scripts to run. After the timeout period expires, the LIM stops the scripts.

Use the external static LIM to automatically detect the operating system type and version of hosts.

LSF automatically detects the operating systems types and versions and displays them when running `lshosts -l` or `lshosts -s`. You can then specify those types in any `-R` resource requirement string. For example, `bsub -R "select[ostype=RHEL4.6]"`.

## Default

10 seconds

## EGO\_PARAMETER

EGO\_ESLIM\_TIMEOUT

## JOB\_STARTER\_EXTEND

### Syntax

`JOB_STARTER_EXTEND="preservestarter" | "preservestarter userstarter"`

### Description

Applies to Windows execution hosts only.

Allows you to use a job starter that includes symbols (for example: &&, |, ||). The job starter configured in `JOB_STARTER_EXTEND` can handle these special characters. The file `$LSF_TOP/8.0/mi sc/examples/preservestarter.c` is the only extended job starter created by default. Users can also develop their own extended job starters based on `preservestarter.c`.

You must also set `JOB_STARTER=preservestarter` in `lsb.queues`.

### Default

Not defined.

## LSB\_API\_CONNTIMEOUT

### Syntax

`LSB_API_CONNTIMEOUT=time_seconds`

### Description

The timeout in seconds when connecting to LSF.

### Valid values

Any positive integer or zero

### Default

10

### See also

LSB\_API\_RECVTIMEOUT

## LSB\_API\_RECVTIMEOUT

### Syntax

`LSB_API_RECVTIMEOUT=time_seconds`

### Description

Timeout in seconds when waiting for a reply from LSF.

## Valid values

Any positive integer or zero

## Default

10

## See also

LSB\_API\_CONNTIMEOUT

# LSB\_API\_VERBOSE

## Syntax

**LSB\_API\_VERBOSE=Y | N**

## Description

When `LSB_API_VERBOSE=Y`, LSF batch commands will display a retry error message to `stderr` when LIM is not available:

```
LSF daemon (LIM) not responding ... still trying
```

When `LSB_API_VERBOSE=N`, LSF batch commands will not display a retry error message when LIM is not available.

## Default

Y. Retry message is displayed to `stderr`.

# LSB\_BJOBS\_CONSISTENT\_EXIT\_CODE

## Syntax

**LSB\_BJOBS\_CONSISTENT\_EXIT\_CODE=Y | N**

## Description

When `LSB_BJOBS_CONSISTENT_EXIT_CODE=Y`, the `bjobs` command exits with 0 only when unfinished jobs are found, and 255 when no jobs are found, or a non-existent job ID is entered.

No jobs are running:

```
bjobs
```

```
No unfinished job found
```

```
echo $?
```

```
255
```

Job 123 does not exist:

```
bjobs 123
```

```
Job <123> is not found
```

```
echo $?
```

```
255
```

Job 111 is running:

**bjobs 111**

JOBID	USER	STAT	QUEUE	FROM_HOST	EXEC_HOST	JOB_NAME	SUBMIT_TIME
111	user1	RUN	normal	hostA	hostB	myjob	Oct 22 09:22

**echo \$?**

0

Job 111 is running, and job 123 does not exist:

**bjobs 111 123**

JOBID	USER	STAT	QUEUE	FROM_HOST	EXEC_HOST	JOB_NAME	SUBMIT_TIME
111	user1	RUN	normal	hostA	hostB	myjob	Oct 22 09:22

Job <123> is not found

**echo \$?**

255

Job 111 is finished:

**bjobs 111**

No unfinished job found

**echo \$?**

255

When LSB\_BJOBS\_CONSISTENT\_EXIT\_CODE=N, the bjobs command exits with 255 only when a non-existent job ID is entered. bjobs returns 0 when no jobs are found, all jobs are finished, or if at least one job ID is valid.

No jobs are running:

**bjobs**

No unfinished job found

**echo \$?**

0

Job 123 does not exist:

**bjobs 123**

Job <123> is not found

**echo \$?**

0

Job 111 is running:

**bjobs 111**

JOBID	USER	STAT	QUEUE	FROM_HOST	EXEC_HOST	JOB_NAME	SUBMIT_TIME
111	user1	RUN	normal	hostA	hostB	myjob	Oct 22 09:22

**echo \$?**

0

Job 111 is running, and job 123 does not exist:

#### **bjobs 111 123**

JOBID	USER	STAT	QUEUE	FROM_HOST	EXEC_HOST	JOB_NAME	SUBMIT_TIME
111	user1	RUN	normal	hostA	hostB	myjob	Oct 22 09:22

Job <123> is not found

**echo \$?**

255

Job 111 is finished:

**bjobs 111**

No unfinished job found

**echo \$?**

0

## Default

N.

# LSB\_BLOCK\_JOBINFO\_TIMEOUT

## Syntax

**LSB\_BLOCK\_JOBINFO\_TIMEOUT**=*time\_minutes*

## Description

Timeout in minutes for job information query commands (e.g., bjobs).

## Valid values

Any positive integer

## Default

Not defined (no timeout)

## See also

MAX\_JOBINFO\_QUERY\_PERIOD in lsb.params

# LSB\_BPEEK\_REMOTE\_OUTPUT

## Syntax

**LSB\_BPEEK\_REMOTE\_OUTPUT**=y|Y|n|N

## Description

If disabled (set to N), the bpeek command attempts to retrieve the job output from the local host first. If that fails, bpeek attempts to retrieve the job output from the remote host instead.

If enabled (set to Y), it is the opposite. The bpeek command attempts to retrieve the job output from the remote host first, then the local host.

When attempting to retrieve the job output from the remote host, `bpeek` attempts to use `RES` first, then `rsh`. If neither is running on the remote host, the `bpeek` command cannot retrieve job output.

## Best Practices

Three directories are related to the `bpeek` command:

- the user's home directory
- `JOB_SPOOL_DIR`
- the checkpoint directory

If these directories are on a shared file system, this parameter can be disabled.

If any of these directories are not on a shared file system, this parameter should be enabled, and either `RES` or `rsh` should be started on the remote job execution host.

## Default

N

# LSB\_CHUNK\_RUSAGE

## Syntax

**LSB\_CHUNK\_RUSAGE=y**

## Description

Applies only to chunk jobs. When set, `sbat chd` contacts PIM to retrieve resource usage information to enforce resource usage limits on chunk jobs.

By default, resource usage limits are not enforced for chunk jobs because chunk jobs are typically too short to allow LSF to collect resource usage.

If `LSB_CHUNK_RUSAGE=Y` is defined, limits may not be enforced for chunk jobs that take less than a minute to run.

## Default

Not defined. No resource usage is collected for chunk jobs.

# LSB\_CMD\_LOG\_MASK

## Syntax

**LSB\_CMD\_LOG\_MASK=*log\_level***

## Description

Specifies the logging level of error messages from LSF batch commands.

To specify the logging level of error messages for LSF commands, use `LSB_CMD_LOG_MASK`. To specify the logging level of error messages for LSF daemons, use `LSB_LOG_MASK`.

`LSB_CMD_LOG_MASK` sets the log level and is used in combination with `LSB_DEBUG_CMD`, which sets the log class for LSF batch commands. For example:

```
LSB_CMD_LOG_MASK=LOG_DEBUG LSB_DEBUG_CMD="LC_TRACE LC_EXEC"
```

LSF commands log error messages in different levels so that you can choose to log all messages, or only log messages that are deemed critical. The level specified by `LSB_CMD_LOG_MASK` determines which messages are recorded and which are discarded. All messages logged at the specified level or higher are recorded, while lower level messages are discarded.

For debugging purposes, the level `LOG_DEBUG` contains the fewest number of debugging messages and is used for basic debugging. The level `LOG_DEBUG3` records all debugging messages, and can cause log files to grow very large; it is not often used. Most debugging is done at the level `LOG_DEBUG2`.

The commands log to the `syslog` facility unless `LSB_CMD_LOGDIR` is set.

## Valid values

The log levels from highest to lowest are:

- `LOG_EMERG`
- `LOG_ALERT`
- `LOG_CRIT`
- `LOG_ERR`
- `LOG_WARNING`
- `LOG_NOTICE`
- `LOG_INFO`
- `LOG_DEBUG`
- `LOG_DEBUG1`
- `LOG_DEBUG2`
- `LOG_DEBUG3`

## Default

`LOG_WARNING`

## See also

`LSB_CMD_LOGDIR`, `LSB_DEBUG`, `LSB_DEBUG_CMD`, `LSB_TIME_CMD`, `LSF_CMD_LOGDIR`, `LSF_CMD_LOG_MASK`, `LSF_LOG_MASK`, `LSF_LOGDIR`, `LSF_TIME_CMD`

# LSB\_CMD\_LOGDIR

## Syntax

**`LSB_CMD_LOGDIR=`*path***

## Description

Specifies the path to the LSF command log files.

## Default

`/tmp`

## See also

`LSB_CMD_LOGDIR`, `LSB_DEBUG`, `LSB_DEBUG_CMD`, `LSB_TIME_CMD`, `LSF_CMD_LOGDIR`, `LSF_CMD_LOG_MASK`, `LSF_LOG_MASK`, `LSF_LOGDIR`, `LSF_TIME_CMD`

## LSB\_CPUSET\_BESTCPUS

### Syntax

**LSB\_CPUSET\_BESTCPUS=y | Y**

### Description

If set, enables the best-fit algorithm for SGI cpusets

### Default

Y (best-fit)

## LSB\_CONFDIR

### Syntax

**LSB\_CONFDIR=*path***

### Description

Specifies the path to the directory containing the LSF configuration files.

The configuration directories are installed under LSB\_CONFDIR.

Configuration files for each cluster are stored in a subdirectory of LSB\_CONFDIR. This subdirectory contains several files that define user and host lists, operation parameters, and queues.

All files and directories under LSB\_CONFDIR must be readable from all hosts in the cluster.

LSB\_CONFDIR/*cluster\_name/confdir* must be owned by the LSF administrator.

If live reconfiguration through the `bconf` command is enabled by the parameter `LSF_LIVE_CONFDIR`, configuration files are written to and read from the directory set by `LSF_LIVE_CONFDIR`.

Do not change this parameter after LSF has been installed.

### Default

LSF\_CONFDIR/*lsbat ch*

### See also

LSF\_CONFDIR, LSF\_LIVE\_CONFDIR

## LSB\_CRDIR

### Syntax

**LSB\_CRDIR=*path***

### Description

Specifies the path and directory to the checkpointing executables on systems that support kernel-level checkpointing. LSB\_CRDIR specifies the directory containing the `chkpnt` and `restart` utility programs that `lsbat chd` uses to checkpoint or restart a job.

For example:

```
LSB_CRDIR=/usr/bin
```

If your platform supports kernel-level checkpointing, and if you want to use the utility programs provided for kernel-level checkpointing, set `LSB_CRDIR` to the location of the utility programs.

## Default

Not defined. The system uses `/bin`.

# LSB\_DEBUG

## Syntax

```
LSB_DEBUG=1 | 2
```

## Description

Sets the LSF batch system to debug.

If defined, LSF runs in single user mode:

- No security checking is performed
- Daemons do not run as root

When `LSB_DEBUG` is defined, LSF does not look in the system services database for port numbers. Instead, it uses the port numbers defined by the parameters `LSB_MBD_PORT/LSB_SBD_PORT` in `lsf.conf`. If these parameters are not defined, it uses port number 40000 for `mbatchd` and port number 40001 for `sbatchd`.

You should always specify 1 for this parameter unless you are testing LSF.

Can also be defined from the command line.

## Valid values

```
LSB_DEBUG=1
```

The LSF system runs in the background with no associated control terminal.

```
LSB_DEBUG=2
```

The LSF system runs in the foreground and prints error messages to `tty`.

## Default

Not defined

## See also

`LSB_DEBUG`, `LSB_DEBUG_CMD`, `LSB_DEBUG_MBD`, `LSB_DEBUG_NQS`, `LSB_DEBUG_SBD`, `LSB_DEBUG_SCH`, `LSF_DEBUG_LIM`, `LSF_DEBUG_RES`, `LSF_LIM_PORT`, `LSF_RES_PORT`, `LSB_MBD_PORT`, `LSB_SBD_PORT`, `LSF_LOGDIR`, `LSF_LIM_DEBUG`, `LSF_RES_DEBUG`

# LSB\_DEBUG\_CMD

## Syntax

```
LSB_DEBUG_CMD=log_class
```

## Description

Sets the debugging log class for commands and APIs.

Specifies the log class filtering to be applied to LSF batch commands or the API. Only messages belonging to the specified log class are recorded.

LSB\_DEBUG\_CMD sets the log class and is used in combination with LSB\_CMD\_LOG\_MASK, which sets the log level. For example:

```
LSB_CMD_LOG_MASK=LOG_DEBUG LSB_DEBUG_CMD="LC_TRACE LC_EXEC"
```

Debugging is turned on when you define both parameters.

The daemons log to the `syslog` facility unless `LSB_CMD_LOGDIR` is defined.

To specify multiple log classes, use a space-separated list enclosed by quotation marks. For example:

```
LSB_DEBUG_CMD="LC_TRACE LC_EXEC"
```

Can also be defined from the command line.

## Valid values

Valid log classes are:

- LC\_ADVRSV and LC2\_ADVRSV: Log advance reservation modifications
- LC\_AFS and LC2\_AFS: Log AFS messages
- LC\_AUTH and LC2\_AUTH: Log authentication messages
- LC\_CHKPNT and LC2\_CHKPNT: Log checkpointing messages
- LC\_COMM and LC2\_COMM: Log communication messages
- LC\_DCE and LC2\_DCE: Log messages pertaining to DCE support
- LC\_EEVENTD and LC2\_EEVENTD: Log eventd messages
- LC\_ELIM and LC2\_ELIM: Log ELIM messages
- LC\_EXEC and LC2\_EXEC: Log significant steps for job execution
- LC\_FAIR and LC2\_FAIR: Log fairshare policy messages
- LC\_FILE and LC2\_FILE: Log file transfer messages
- LC\_FLEX and LC2\_FLEX: Log messages related to FlexNet
- LC2\_GUARANTEE: Log messages related to guarantee SLAs
- LC\_HANG and LC2\_HANG: Mark where a program might hang
- LC\_JARRAY and LC2\_JARRAY: Log job array messages
- LC\_JLIMIT and LC2\_JLIMIT: Log job slot limit messages
- LC\_LICENSE and LC2\_LICENSE : Log license management messages (LC\_LICENSE is also supported for backward compatibility)
- LC2\_LIVECONF: Log live reconfiguration messages
- LC\_LOADINDX and LC2\_LOADINDX: Log load index messages
- LC\_M\_LOG and LC2\_M\_LOG: Log multievent logging messages
- LC\_MEMORY and LC2\_MEMORY: Log messages related to MEMORY allocation
- LC\_MPI and LC2\_MPI: Log MPI messages
- LC\_MULTI and LC2\_MULTI: Log messages pertaining to MultiCluster
- LC\_PEND and LC2\_PEND: Log messages related to job pending reasons
- LC\_PERFM and LC2\_PERFM: Log performance messages
- LC\_PIM and LC2\_PIM: Log PIM messages
- LC\_PREEMPT and LC2\_PREEMPT: Log preemption policy messages

- LC\_RESOURCE and LC2\_RESOURCE: Log messages related to resource broker
- LC\_RESREQ and LC2\_RESREQ: Log resource requirement messages
- LC\_SCHED and LC2\_SCHED: Log messages pertaining to the mbatchd scheduler.
- LC\_SIGNAL and LC2\_SIGNAL: Log messages pertaining to signals
- LC\_SYS and LC2\_SYS: Log system call messages
- LC\_TRACE and LC2\_TRACE: Log significant program walk steps
- LC\_XDR and LC2\_XDR: Log everything transferred by XDR
- LC\_XDRVERSION and LC2\_XDRVERSION: Log messages for XDR version

## Default

Not defined

## See also

LSB\_CMD\_LOG\_MASK, LSB\_CMD\_LOGDIR, LSB\_DEBUG, LSB\_DEBUG\_MBD, LSB\_DEBUG\_NQS, LSB\_DEBUG\_SBD, LSB\_DEBUG\_SCH, LSF\_DEBUG\_LIM, LSF\_DEBUG\_RES, LSF\_LIM\_PORT, LSF\_RES\_PORT, LSB\_MBD\_PORT, LSB\_SBD\_PORT, LSF\_LOGDIR, LSF\_LIM\_DEBUG, LSF\_RES\_DEBUG

# LSB\_DEBUG\_MBD

## Syntax

**LSB\_DEBUG\_MBD**=*log\_class*

## Description

Sets the debugging log class for mbat chd.

Specifies the log class filtering to be applied to mbat chd. Only messages belonging to the specified log class are recorded.

LSB\_DEBUG\_MBD sets the log class and is used in combination with LSF\_LOG\_MASK, which sets the log level. For example:

```
LSF_LOG_MASK=LOG_DEBUG LSB_DEBUG_MBD="LC_TRACE LC_EXEC"
```

To specify multiple log classes, use a space-separated list enclosed in quotation marks. For example:

```
LSB_DEBUG_MBD="LC_TRACE LC_EXEC"
```

You need to restart the daemons after setting LSB\_DEBUG\_MBD for your changes to take effect.

If you use the command `badmi n mbddebug` to temporarily change this parameter without changing `lsf.conf`, you do not need to restart the daemons.

## Valid values

Valid log classes are the same as for LSB\_DEBUG\_CMD except for the log class LC\_ELIM, which cannot be used with LSB\_DEBUG\_MBD. See LSB\_DEBUG\_CMD.

## Default

Not defined

## See also

LSB\_CMD\_LOG\_MASK, LSB\_CMD\_LOGDIR, LSB\_DEBUG, LSB\_DEBUG\_MBD, LSB\_DEBUG\_NQS, LSB\_DEBUG\_SBD, LSB\_DEBUG\_SCH, LSF\_DEBUG\_LIM, LSF\_DEBUG\_RES, LSF\_LIM\_PORT, LSF\_RES\_PORT, LSB\_MBD\_PORT, LSB\_SBD\_PORT, LSF\_LOGDIR, LSF\_LIM\_DEBUG, LSF\_RES\_DEBUG

# LSB\_DEBUG\_NQS

## Syntax

**LSB\_DEBUG\_NQS**=*log\_class*

## Description

Sets the log class for debugging the NQS interface.

Specifies the log class filtering to be applied to NQS. Only messages belonging to the specified log class are recorded.

LSB\_DEBUG\_NQS sets the log class and is used in combination with LSF\_LOG\_MASK, which sets the log level. For example:

```
LSF_LOG_MASK=LOG_DEBUG LSB_DEBUG_NQS="LC_TRACE LC_EXEC"
```

Debugging is turned on when you define both parameters.

To specify multiple log classes, use a space-separated list enclosed in quotation marks. For example:

```
LSB_DEBUG_NQS="LC_TRACE LC_EXEC"
```

This parameter can also be defined from the command line.

## Valid values

For a list of valid log classes, see LSB\_DEBUG\_CMD.

## Default

Not defined

## See also

LSB\_DEBUG\_CMD, LSF\_CMD\_LOGDIR, LSF\_CMD\_LOG\_MASK, LSF\_LOG\_MASK, LSF\_LOGDIR

# LSB\_DEBUG\_SBD

## Syntax

**LSB\_DEBUG\_SBD**=*log\_class*

## Description

Sets the debugging log class for sbat chd.

Specifies the log class filtering to be applied to sbat chd. Only messages belonging to the specified log class are recorded.

LSB\_DEBUG\_SBD sets the log class and is used in combination with LSF\_LOG\_MASK, which sets the log level. For example:

```
LSF_LOG_MASK=LOG_DEBUG LSB_DEBUG_SBD="LC_TRACE LC_EXEC"
```

To specify multiple log classes, use a space-separated list enclosed in quotation marks. For example:

```
LSB_DEBUG_SBD="LC_TRACE LC_EXEC"
```

You need to restart the daemons after setting LSB\_DEBUG\_SBD for your changes to take effect.

If you use the command `badmi n sbddebug` to temporarily change this parameter without changing `lsf.conf`, you do not need to restart the daemons.

## Valid values

Valid log classes are the same as for LSB\_DEBUG\_CMD except for the log class LC\_ELIM, which cannot be used with LSB\_DEBUG\_SBD. See LSB\_DEBUG\_CMD.

## Default

Not defined

## See also

LSB\_DEBUG\_MBD, LSF\_CMD\_LOGDIR, LSF\_CMD\_LOG\_MASK, LSF\_LOG\_MASK, LSF\_LOGDIR, badmi n

# LSB\_DEBUG\_SCH

## Syntax

```
LSB_DEBUG_SCH=log_class
```

## Description

Sets the debugging log class for `mbschd`.

Specifies the log class filtering to be applied to `mbschd`. Only messages belonging to the specified log class are recorded.

LSB\_DEBUG\_SCH sets the log class and is used in combination with LSF\_LOG\_MASK, which sets the log level. For example:

```
LSF_LOG_MASK=LOG_DEBUG LSB_DEBUG_SCH="LC_SCHED"
```

To specify multiple log classes, use a space-separated list enclosed in quotation marks. For example:

```
LSB_DEBUG_SCH="LC_SCHED LC_TRACE LC_EXEC"
```

You need to restart the daemons after setting LSB\_DEBUG\_SCH for your changes to take effect.

## Valid values

Valid log classes are the same as for LSB\_DEBUG\_CMD except for the log class LC\_ELIM, which cannot be used with LSB\_DEBUG\_SCH, and LC\_HPC and LC\_SCHED, which are only valid for LSB\_DEBUG\_SCH. See LSB\_DEBUG\_CMD.

## Default

Not defined

## See also

LSB\_DEBUG\_MBD, LSB\_DEBUG\_SBD, LSF\_CMD\_LOGDIR, LSF\_CMD\_LOG\_MASK, LSF\_LOG\_MASK, LSF\_LOGDIR, badmi n

# LSB\_DISABLE\_LIMLOCK\_EXCL

## Syntax

**LSB\_DISABLE\_LIMLOCK\_EXCL=y | n**

## Description

If preemptive scheduling is enabled, this parameter enables preemption of and preemption by exclusive jobs when `PREEMPT_JOBTYPE=EXCLUSIVE` in `lsb.params`. Changing this parameter requires a restart of all `sbatchds` in the cluster (`badmi n hrestart`). Do not change this parameter while exclusive jobs are running.

When `LSB_DISABLE_LIMLOCK_EXCL=y`, for a host running an exclusive job:

- LIM is not locked on a host running an exclusive job
- `lsl oad` displays the host status `ok`.
- `bhosts` displays the host status `closed`.
- Users can run tasks on the host using `l srun` or `l sgrun`. To prevent users from running tasks during execution of an exclusive job, the parameter `LSF_DISABLE_LSRUN=y` must be defined in `lsf.conf`.

## Default

n. LSF locks the LIM on a host running an exclusive job and unlocks the LIM when the exclusive job finishes.

# LSB\_DISABLE\_RERUN\_POST\_EXEC

## Syntax

**LSB\_DISABLE\_RERUN\_POST\_EXEC=y | Y**

## Description

If set, and the job is rerunnable, the `POST_EXEC` configured at the job level or the queue level is not executed if the job is rerun.

Running of post-execution commands upon restart of a rerunnable job may not always be desirable. For example, if the post-exec removes certain files, or does other cleanup that should only happen if the job finishes successfully, use `LSB_DISABLE_RERUN_POST_EXEC` to prevent the post-exec from running and allow the successful continuation of the job when it reruns.

The `POST_EXEC` may still run for a job rerun when the execution host loses contact with the master host due to network problems. In this case `mbatchd` assumes the job has failed and restarts the job on another host. The original execution host, out of contact with the master host, completes the job and runs the `POST_EXEC`.

## Default

Not defined

## LSB\_DISPLAY\_YEAR

### Syntax

**LSB\_DISPLAY\_YEAR=y|Y|n|N**

### Description

Toggles on and off inclusion of the year in the time string displayed by the commands `bj obs -l`, `bacct -l`, and `bhist -l|-b|-t`.

### Default

N

## LSB\_ECHKPNT\_KEEP\_OUTPUT

### Syntax

**LSB\_ECHKPNT\_KEEP\_OUTPUT=y | Y**

### Description

Saves the standard output and standard error of custom `echknt` and `erestart` methods to:

- `checkpoint_dir/$LSB_JOBID/echknt.out`
- `checkpoint_dir/$LSB_JOBID/echknt.err`
- `checkpoint_dir/$LSB_JOBID/erestart.out`
- `checkpoint_dir/$LSB_JOBID/erestart.err`

Can also be defined as an environment variable.

### Default

Not defined. Standard error and standard output messages from custom `echknt` and `erestart` programs is directed to `/dev/null` and discarded by LSF.

### See also

LSB\_ECHKPNT\_METHOD, LSB\_ECHKPNT\_METHOD\_DIR

## LSB\_ECHKPNT\_METHOD

### Syntax

**LSB\_ECHKPNT\_METHOD="*method\_name* [*method\_name*] ..."**

### Description

Name of custom `echknt` and `erestart` methods.

Can also be defined as an environment variable, or specified through the `bsub -k` option.

The name you specify here is used for both your custom `echknt` and `erestart` programs. You must assign your custom `echknt` and `erestart` programs the name `echknt.method_name` and `erestart.method_name`. The programs `echknt.method_name` and `erestart.method_name` must be in `LSF_SERVERDIR` or in the directory specified by `LSB_ECHKPNT_METHOD_DIR`.

Do not define `LSB_ECHKPNT_METHOD=default` as `default` is a reserved keyword to indicate to use the default `echknt` and `erestart` methods of LSF. You can however, specify `bsub -k "my_dir method=default" my_job` to indicate that you want to use the default checkpoint and restart methods.

When this parameter is not defined in `lsf.conf` or as an environment variable and no custom method is specified at job submission through `bsub -k`, LSF uses `echknt.default` and `erestart.default` to checkpoint and restart jobs.

When this parameter is defined, LSF uses the custom checkpoint and restart methods specified.

## Limitations

The method name and directory (`LSB_ECHKPNT_METHOD_DIR`) combination must be unique in the cluster.

For example, you may have two `echknt` applications with the same name such as `echknt.mymethod` but what differentiates them is the different directories defined with `LSB_ECHKPNT_METHOD_DIR`. It is the cluster administrator's responsibility to ensure that method name and method directory combinations are unique in the cluster.

## Default

Not defined. LSF uses `echknt.default` and `erestart.default` to checkpoint and restart jobs

## See also

`LSB_ECHKPNT_METHOD_DIR`, `LSB_ECHKPNT_KEEP_OUTPUT`

# LSB\_ECHKPNT\_METHOD\_DIR

## Syntax

`LSB_ECHKPNT_METHOD_DIR=path`

## Description

Absolute path name of the directory in which custom `echknt` and `erestart` programs are located.

The checkpoint method directory should be accessible by all users who need to run the custom `echknt` and `erestart` programs.

Can also be defined as an environment variable.

## Default

Not defined. LSF searches in `LSF_SERVERDIR` for custom `echknt` and `erestart` programs.

## See also

`LSB_ESUB_METHOD`, `LSB_ECHKPNT_KEEP_OUTPUT`

# LSB\_ESUB\_METHOD

## Syntax

`LSB_ESUB_METHOD="esub_application [esub_application] ..."`

## Description

Specifies a mandatory esub that applies to all job submissions. `LSB_ESUB_METHOD` lists the names of the application-specific esub executables used in addition to any executables specified by the `bsub -a` option.

For example, `LSB_ESUB_METHOD="dce fluent"` runs `LSF_SERVERDIR/esub.dce` and `LSF_SERVERDIR/esub.fluent` for all jobs submitted to the cluster. These esubs define, respectively, DCE as the mandatory security system and FLUENT as the mandatory application for all jobs.

`LSB_ESUB_METHOD` can also be defined as an environment variable.

The value of `LSB_ESUB_METHOD` must correspond to an actual esub file. For example, to use `LSB_ESUB_METHOD=fluent`, the file `esub.fluent` must exist in `LSF_SERVERDIR`.

The name of the esub program must be a valid file name. Valid file names contain only alphanumeric characters, underscore (`_`) and hyphen (`-`).

---

### Restriction:

The name `esub.user` is reserved. Do not use the name `esub.user` for an application-specific esub.

---

The master esub (`mesub`) uses the name you specify to invoke the appropriate esub program. The esub and `esub.esub_application` programs must be located in `LSF_SERVERDIR`.

LSF does not detect conflicts based on esub names. For example, if `LSB_ESUB_METHOD="openmpi"` and `bsub -a pvm` is specified at job submission, the job could fail because these esubs define two different types of parallel job handling.

## Default

Not defined. LSF does not apply a mandatory esub to jobs submitted to the cluster.

## LSB\_EVENTS\_FILE\_KEEP\_OPEN

### Syntax

`LSB_EVENTS_FILE_KEEP_OPEN=Y|N`

### Description

Windows only.

Specify `Y` to open the events file once, and keep it open always.

Specify `N` to open and close the events file each time a record is written.

## Default

`Y`

## LSB\_HJOB\_PER\_SESSION

### Syntax

`LSB_HJOB_PER_SESSION=max_num`

## Description

Specifies the maximum number of jobs that can be dispatched in each scheduling cycle to each host

## Valid values

Any positive integer

## Default

Not defined

## Notes

LSB\_HJOB\_PER\_SESSION is activated only if the JOB\_ACCEPT\_INTERVAL parameter is set to 0.

## See also

JOB\_ACCEPT\_INTERVAL parameter in lsb.params

## LSB\_INDEX\_BY\_JOB

### Syntax

**LSB\_INDEX\_BY\_JOB="JOBNAME"**

### Description

When set to JOBNAME, creates a job index of job names. Define when using job dependency conditions (bsub -w) with job names to optimize job name searches.

### Valid values

JOBNAME

### Default

Not defined. Job index is not created.

## LSB\_INTERACT\_MSG\_ENH

### Syntax

**LSB\_INTERACT\_MSG\_ENH=y | Y**

### Description

If set, enables enhanced messaging for interactive batch jobs. To disable interactive batch job messages, set LSB\_INTERACT\_MSG\_ENH to any value other than y or Y; for example, LSB\_INTERACT\_MSG\_ENH=N.

### Default

Not defined

## See also

LSB\_INTERACT\_MSG\_INTVAL

# LSB\_INTERACT\_MSG\_INTVAL

## Syntax

**LSB\_INTERACT\_MSG\_INTVAL**=*time\_seconds*

## Description

Specifies the update interval in seconds for interactive batch job messages.

LSB\_INTERACT\_MSG\_INTVAL is ignored if LSB\_INTERACT\_MSG\_ENH is not set.

Job information that LSF uses to get the pending or suspension reason is updated according to the value of PEND\_REASON\_UPDATE\_INTERVAL in l sb. params.

## Default

Not defined. If LSB\_INTERACT\_MSG\_INTVAL is set to an incorrect value, the default update interval is 60 seconds.

## See also

LSB\_INTERACT\_MSG\_ENH

# LSB\_JOB\_CPULIMIT

## Syntax

**LSB\_JOB\_CPULIMIT**=y | n

## Description

Determines whether the CPU limit is a per-process limit enforced by the OS or whether it is a per-job limit enforced by LSF:

- The per-process limit is enforced by the OS when the CPU time of one process of the job exceeds the CPU limit.
- The per-job limit is enforced by LSF when the total CPU time of all processes of the job exceed the CPU limit.

This parameter applies to CPU limits set when a job is submitted with bsub -c, and to CPU limits set for queues by CPULIMIT in l sb. queues.

- LSF-enforced per-job limit: When the sum of the CPU time of all processes of a job exceed the CPU limit, LSF sends a SIGXCPU signal (where supported by the operating system) from the operating system to all processes belonging to the job, then SIGINT, SIGTERM and SIGKILL. The interval between signals is 10 seconds by default. The time interval between SIGXCPU, SIGINT, SIGKILL, SIGTERM can be configured with the parameter JOB\_TERMINATE\_INTERVAL in l sb. params.

---

### Restriction:

SIGXCPU is not supported by Windows.

---

- OS-enforced per process limit: When one process in the job exceeds the CPU limit, the limit is enforced by the operating system. For more details, refer to your operating system documentation for `setrlimit(0)`.

The setting of `LSB_JOB_CPULIMIT` has the following effect on how the limit is enforced:

`LSB_JOB_CPULIMIT` LSF per-job limit OS per-process limit

y Enabled Disabled

n Disabled Enabled

Not defined Enabled Enabled

## Default

Not defined

## Notes

To make `LSB_JOB_CPULIMIT` take effect, use the command `badmi n hrestart all` to restart all `lsb_queues` in the cluster.

Changing the default Terminate job control action: You can define a different terminate action in `lsb_queues` with the parameter `JOB_CONTROLS` if you do not want the job to be killed. For more details on job controls, see *Administering Platform LSF*.

## Limitations

If a job is running and the parameter is changed, LSF is not able to reset the type of limit enforcement for running jobs.

- If the parameter is changed from per-process limit enforced by the OS to per-job limit enforced by LSF (`LSB_JOB_CPULIMIT=n` changed to `LSB_JOB_CPULIMIT=y`), both per-process limit and per-job limit affect the running job. This means that signals may be sent to the job either when an individual process exceeds the CPU limit or the sum of the CPU time of all processes of the job exceed the limit. A job that is running may be killed by the OS or by LSF.
- If the parameter is changed from per-job limit enforced by LSF to per-process limit enforced by the OS (`LSB_JOB_CPULIMIT=y` changed to `LSB_JOB_CPULIMIT=n`), the job is allowed to run without limits because the per-process limit was previously disabled.

## See also

`lsb_queues`, `bsub`, `JOB_TERMINATE_INTERVAL` in `lsb.params`, `LSB_MOD_ALL_JOBS`

# LSB\_JOB\_MEMLIMIT

## Syntax

`LSB_JOB_MEMLIMIT=y | n`

## Description

Determines whether the memory limit is a per-process limit enforced by the OS or whether it is a per-job limit enforced by LSF.

- The per-process limit is enforced by the OS when the memory allocated to one process of the job exceeds the memory limit.

- The per-job limit is enforced by LSF when the sum of the memory allocated to all processes of the job exceeds the memory limit.

This parameter applies to memory limits set when a job is submitted with `bsub -M mem_limit`, and to memory limits set for queues with `MEMLIMIT` in `l sb. queues`.

The setting of `LSB_JOB_MEMLIMIT` has the following effect on how the limit is enforced:

When <code>LSB_JOB_MEMLIMIT</code> is	LSF-enforced per-job limit	OS-enforced per-process limit
y	Enabled	Disabled
n or not defined	Disabled	Enabled

When `LSB_JOB_MEMLIMIT` is Y, the LSF-enforced per-job limit is enabled, and the OS-enforced per-process limit is disabled.

When `LSB_JOB_MEMLIMIT` is N or not defined, the LSF-enforced per-job limit is disabled, and the OS-enforced per-process limit is enabled.

*LSF-enforced per-job limit:* When the total memory allocated to all processes in the job exceeds the memory limit, LSF sends the following signals to kill the job: `SIGINT`, `SIGTERM`, then `SIGKILL`. The interval between signals is 10 seconds by default.

On UNIX, the time interval between `SIGINT`, `SIGKILL`, `SIGTERM` can be configured with the parameter `JOB_TERMINATE_INTERVAL` in `l sb. params`.

*OS-enforced per process limit:* When the memory allocated to one process of the job exceeds the memory limit, the operating system enforces the limit. LSF passes the memory limit to the operating system. Some operating systems apply the memory limit to each process, and some do not enforce the memory limit at all.

OS memory limit enforcement is only available on systems that support `RLIMIT_RSS` for `setrlimit()`.

The following operating systems do not support the memory limit at the OS level and the job is allowed to run without a memory limit:

- Windows
- Sun Solaris 2.x

## Default

Not defined. Per-process memory limit enforced by the OS; per-job memory limit enforced by LSF disabled

## Notes

To make `LSB_JOB_MEMLIMIT` take effect, use the command `badmi n hrestart all` to restart all `sbatchds` in the cluster.

If `LSB_JOB_MEMLIMIT` is set, it overrides the setting of the parameter `LSB_MEMLIMIT_ENFORCE`. The parameter `LSB_MEMLIMIT_ENFORCE` is ignored.

The difference between `LSB_JOB_MEMLIMIT` set to y and `LSB_MEMLIMIT_ENFORCE` set to y is that with `LSB_JOB_MEMLIMIT`, only the per-job memory limit enforced by LSF is enabled. The per-process memory limit enforced by the OS is disabled. With `LSB_MEMLIMIT_ENFORCE` set to y, both the per-job memory limit enforced by LSF and the per-process memory limit enforced by the OS are enabled.

Changing the default Terminate job control action: You can define a different Terminate action in l sb. queues with the parameter JOB\_CONTROLS if you do not want the job to be killed. For more details on job controls, see *Administering Platform LSF*.

## Limitations

If a job is running and the parameter is changed, LSF is not able to reset the type of limit enforcement for running jobs.

- If the parameter is changed from per-process limit enforced by the OS to per-job limit enforced by LSF (LSB\_JOB\_MEMLIMIT=*n* or not defined changed to LSB\_JOB\_MEMLIMIT=*y*), both per-process limit and per-job limit affect the running job. This means that signals may be sent to the job either when the memory allocated to an individual process exceeds the memory limit or the sum of memory allocated to all processes of the job exceed the limit. A job that is running may be killed by LSF.
- If the parameter is changed from per-job limit enforced by LSF to per-process limit enforced by the OS (LSB\_JOB\_MEMLIMIT=*y* changed to LSB\_JOB\_MEMLIMIT=*n* or not defined), the job is allowed to run without limits because the per-process limit was previously disabled.

## See also

LSB\_MEMLIMIT\_ENFORCE, LSB\_MOD\_ALL\_JOBS, l sb. queues, bsub, JOB\_TERMINATE\_INTERVAL in l sb. params

## LSB\_JOB\_OUTPUT\_LOGGING

### Syntax

**LSB\_JOB\_OUTPUT\_LOGGING=Y | N**

### Description

Determines whether jobs write job notification messages to the logfile.

### Default

Not defined (jobs do not write job notification messages to the logfile).

## LSB\_JOBID\_DISP\_LENGTH

### Syntax

**LSB\_JOBID\_DISP\_LENGTH=*integer***

### Description

By default, LSF commands bjobs and bhist display job IDs with a maximum length of 7 characters. Job IDs greater than 9999999 are truncated on the left.

When LSB\_JOBID\_DISP\_LENGTH=10, the width of the JOBID column in bjobs and bhist increases to 10 characters.

### Valid values

Specify an integer between 7 and 10.

## Default

Not defined. LSF uses the default 7-character length for job ID display.

## LSB\_KEEP\_SYSDEF\_RLIMIT

### Syntax

**LSB\_KEEP\_SYSDEF\_RLIMIT=y | n**

### Description

If resource limits are configured for a user in the SGI IRIX User Limits Database (ULDB) domain specified in `LSF_ULDB_DOMAIN`, and there is no domain default, the system default is honored.

If `LSB_KEEP_SYSDEF_RLIMIT=n`, and no resource limits are configured in the domain for the user and there is no domain default, LSF overrides the system default and sets system limits to unlimited.

## Default

Not defined. No resource limits are configured in the domain for the user and there is no domain default.

## LSB\_LOAD\_TO\_SERVER\_HOSTS (OBSOLETE)

### Syntax

**LSB\_LOAD\_TO\_SERVER\_HOSTS=Y | y**

### Description

---

#### Note:

This parameter is obsolete in LSF 7 Update 2. By default, client `sbat chd` contacts the local LIM for host status and load information.

---

Highly recommended for large clusters to decrease the load on the master LIM. Forces the client `sbat chd` to contact the local LIM for host status and load information. The client `sbat chd` only contacts the master LIM or a LIM on one of the `LSF_SERVER_HOSTS` if `sbat chd` cannot find the information locally.

## Default

Y. Client `sbat chd` contacts the local LIM for host status and load information.

## See also

`LSF_SERVER_HOSTS` in `slave.config`

## LSB\_LOCALDIR

### Syntax

**LSB\_LOCALDIR=*path***

## Description

Enables duplicate logging.

Specify the path to a local directory that exists only on the first LSF master host. LSF puts the primary copies of the event and accounting log files in this directory. LSF puts the duplicates in LSB\_SHAREDIR.

---

### Important:

Always restart both the mbacthd and sbatchd when modifying LSB\_LOCALDIR.

---

## Example

```
LSB_LOCALDIR=/usr/share/l sbatchd/lo g i n f o
```

## Default

Not defined

## See also

LSB\_SHAREDIR, EVENT\_UPDATE\_INTERVAL in l sb. params

# LSB\_MAILPROG

## Syntax

**LSB\_MAILPROG**=*file\_name*

## Description

Path and file name of the mail program used by LSF to send email. This is the electronic mail program that LSF uses to send system messages to the user. When LSF needs to send email to users it invokes the program defined by LSB\_MAILPROG in l sf. conf. You can write your own custom mail program and set LSB\_MAILPROG to the path where this program is stored.

LSF administrators can set the parameter as part of cluster reconfiguration. Provide the name of any mail program. For your convenience, LSF provides the sendmail mail program, which supports the sendmail protocol on UNIX.

In a mixed cluster, you can specify different programs for Windows and UNIX. You can set this parameter during installation on Windows. For your convenience, LSF provides the l smail . exe mail program, which supports SMTP and Microsoft Exchange Server protocols on Windows. If l smail is specified, the parameter LSB\_MAILSERVER must also be specified.

If you change your mail program, the LSF administrator must restart sbatchd on all hosts to retrieve the new value.

## UNIX

By default, LSF uses /usr/lib/sendmail to send email to users. LSF calls LSB\_MAILPROG with two arguments; one argument gives the full name of the sender, and the other argument gives the return address for mail.

LSB\_MAILPROG must read the body of the mail message from the standard input. The end of the message is marked by end-of-file. Any program or shell script that accepts the arguments and input, and delivers the mail correctly, can be used.

LSB\_MAILPROG must be executable by any user.

## Windows

If LSB\_MAILPROG is not defined, no email is sent.

## Examples

```
LSB_MAILPROG=lsmail.exe
```

```
LSB_MAILPROG=/serverA/tools/lsf/bin/unixhost.exe
```

## Default

/usr/lib/sendmail (UNIX)

blank (Windows)

## See also

LSB\_MAILSERVER, LSB\_MAILTO

# LSB\_MAILSERVER

## Syntax

```
LSB_MAILSERVER=mail_protocol:mail_server
```

## Description

Part of mail configuration on Windows.

This parameter only applies when `lsmail` is used as the mail program (LSB\_MAILPROG=lsmail.exe). Otherwise, it is ignored.

Both *mail\_protocol* and *mail\_server* must be indicated.

Set this parameter to either SMTP or Microsoft Exchange protocol (SMTP or EXCHANGE) and specify the name of the host that is the mail server.

This parameter is set during installation of LSF on Windows or is set or modified by the LSF administrator.

If this parameter is modified, the LSF administrator must restart `sbatchd` on all hosts to retrieve the new value.

## Examples

```
LSB_MAILSERVER=EXCHANGE: Host2@company.com
```

```
LSB_MAILSERVER=SMTP: MailHost
```

## Default

Not defined

## See also

LSB\_LOCALDIR

# LSB\_MAILSIZE\_LIMIT

## Syntax

**LSB\_MAILSIZE\_LIMIT**=*email\_size\_KB*

## Description

Limits the size in KB of the email containing job output information.

The system sends job information such as CPU, process and memory usage, job output, and errors in email to the submitting user account. Some batch jobs can create large amounts of output. To prevent large job output files from interfering with your mail system, use LSB\_MAILSIZE\_LIMIT to set the maximum size in KB of the email containing the job information. Specify a positive integer.

If the size of the job output email exceeds LSB\_MAILSIZE\_LIMIT, the output is saved to a file under JOB\_SPOOL\_DIR or to the default job output directory if JOB\_SPOOL\_DIR is not defined. The email informs users of where the job output is located.

If the -o option of bsub is used, the size of the job output is not checked against LSB\_MAILSIZE\_LIMIT.

If you use a custom mail program specified by the LSB\_MAILPROG parameter that can use the LSB\_MAILSIZE environment variable, it is not necessary to configure LSB\_MAILSIZE\_LIMIT.

## Default

By default, LSB\_MAILSIZE\_LIMIT is not enabled. No limit is set on size of batch job output email.

## See also

LSB\_MAILPROG, LSB\_MAILTO

# LSB\_MAIL\_FROM\_DOMAIN

## Syntax

**LSB\_MAIL\_FROM\_DOMAIN**=*domain\_name*

## Description

Windows only.

LSF uses the username as the from address to send mail. In some environments the from address requires domain information. If LSB\_MAIL\_FROM\_DOMAIN is set, the domain name specified in this parameter will be added to the from address.

For example, if LSB\_MAIL\_FROM\_DOMAIN is not set the, from address is SYSTEM; if **LSB\_MAIL\_FROM\_DOMAIN=platform.com**, the from address is SYSTEM@platform.com.

## Default

Not defined.

# LSB\_MAILTO

## Syntax

**LSB\_MAILTO**=*mail\_account*

## Description

LSF sends electronic mail to users when their jobs complete or have errors, and to the LSF administrator in the case of critical errors in the LSF system. The default is to send mail to the user who submitted the job, on the host on which the daemon is running; this assumes that your electronic mail system forwards messages to a central mailbox.

The LSB\_MAILTO parameter changes the mailing address used by LSF. LSB\_MAILTO is a format string that is used to build the mailing address.

Common formats are:

- **!U**: Mail is sent to the submitting user's account name on the local host. The substring **!U**, if found, is replaced with the user's account name.
- **!U@company\_name.com**: Mail is sent to `user@company_name.com` on the mail server. The mail server is specified by LSB\_MAILSERVER.
- **!U!H**: Mail is sent to `user@submission_hostname`. The substring **!H** is replaced with the name of the submission host. This format is valid on UNIX only. It is not supported on Windows.

All other characters (including any other '!') are copied exactly.

If this parameter is modified, the LSF administrator must restart `sbat chd` on all hosts to retrieve the new value.

Windows only: When a job exception occurs (for example, a job is overrun or underrun), an email is sent to the primary administrator set in the `lsf.cluster_name` file to the domain set in LSB\_MAILTO. For example, if the primary administrator is `lsfadmin` and `LSB_MAILTO=fred@company.com`, an email is sent to `lsfadmin@company.com`. The email must be a valid Windows email account.

## Default

**!U**

## See also

LSB\_MAILPROG, LSB\_MAILSIZE\_LIMIT

# LSB\_MAX\_ASKED\_HOSTS\_NUMBER

## Syntax

**LSB\_MAX\_ASKED\_HOSTS\_NUMBER**=integer

## Description

Limits the number of hosts a user can specify with the `-m` (host preference) option of the following commands:

- `bsub`
- `brun`

- bmod
- brestart
- brsvadd
- brsvmod
- brsvs

The job is rejected if more hosts are specified than the value of `LSB_MAX_ASKED_HOSTS_NUMBER`.

---

### Caution:

If this value is set high, there will be a performance effect if users submit or modify jobs using the `-m` option and specify a large number of hosts. 512 hosts is the suggested upper limit.

---

## Valid values

Any whole, positive integer.

## Default

512

# LSB\_MAX\_JOB\_DISPATCH\_PER\_SESSION

## Syntax

`LSB_MAX_JOB_DISPATCH_PER_SESSION=integer`

## Description

Defines the maximum number of jobs that `mbatchd` can dispatch during one job scheduling session.

Both `mbatchd` and `sbatchd` must be restarted when you change the value of this parameter.

If set to a value greater than 300, the file descriptor limit is increased on operating systems that support a file descriptor limit greater than 1024.

Use together with `MAX_SBD_CONNS` in `lsb.params`. Set `LSB_MAX_JOB_DISPATCH_PER_SESSION` to a value no greater than one-half the value of `MAX_SBD_CONNS`. This setting configures `mbatchd` to dispatch jobs at a high rate while maintaining the processing speed of other `mbatchd` tasks.

## Examples

```
LSB_MAX_JOB_DISPATCH_PER_SESSION=300
```

The file descriptor limit is 1024.

```
LSB_MAX_JOB_DISPATCH_PER_SESSION=1000
```

The file descriptor limit is greater than 1024 on operating systems that support a greater limit.

## Default

300

## See also

MAX\_SBD\_CONNS in `l sb. params`

# LSB\_MAX\_PACK\_JOBS

## Syntax

**LSB\_MAX\_PACK\_JOBS**=*integer*

## Description

Applies to job packs only. Enables the job packs feature and specifies the maximum number of job submission requests in one job pack.

If the value is 0, job packs are disabled.

If the value is 1, jobs from the file are submitted individually, as if submitted directly using the `bsub` command.

We recommend 100 as the initial pack size. Tune this parameter based on cluster performance. The larger the pack size, the faster the job submission rate is for all the job requests the job submission file. However, while `mbat chd` is processing a pack, `mbat chd` is blocked from processing other requests, so increasing pack size can affect `mbat chd` response time for other job submissions.

If you change the configuration of this parameter, you must restart `mbat chd`.

Parameters related to job packs are not supported as environment variables.

## Valid Values

Any positive integer or 0.

## Default

0 (disabled)

# LSB\_MAX\_PROBE\_SBD

## Syntax

**LSB\_MAX\_PROBE\_SBD**=*integer*

## Description

Specifies the maximum number of `sbat chd` instances can be polled by `mbat chd` in the interval `MBD_SLEEP_TIME/10` (6 seconds by default). Use this parameter in large clusters to reduce the time it takes for `mbat chd` to probe all `sbat chds`.

The value of `LSB_MAX_PROBE_SBD` cannot be greater than the number of hosts in the cluster. If it is, `mbatchd` adjusts the value of `LSB_MAX_PROBE_SBD` to be same as the number of hosts.

After modifying `LSB_MAX_PROBE_SBD`, use `badmi n mbdrestart` to restart `mbat chd` and let the modified value take effect.

If `LSB_MAX_PROBE_SBD` is defined, the value of `MAX_SBD_FAIL` in `l sb. params` can be less than 3.

## Valid values

Any positive integer between 0 and 64

## Default

20

## See also

MAX\_SBD\_FAIL in `l sb. params`

# LSB\_MAX\_NQS\_QUEUES

## Syntax

**LSB\_MAX\_NQS\_QUEUES**=*nqs\_queues*

## Description

The maximum number of NQS queues allowed in the LSF cluster. Required for LSF to work with NQS. You must restart `mbat chd` if you change the value of `LSB_MAX_NQS_QUEUES`.

The total number of NQS queues configured by `NQS_QUEUES` in `l sb. queues` cannot exceed the value of `LSB_MAX_NQS_QUEUES`. NQS queues in excess of the maximum queues are ignored.

If you do not define `LSB_MAX_NQS_QUEUES` or define an incorrect value, LSF-NQS interoperation is disabled.

## Valid values

Any positive integer

## Default

None

# LSB\_MBD\_BUSY\_MSG

## Syntax

**LSB\_MBD\_BUSY\_MSG**=*"message\_string"*

## Description

Specifies the message displayed when `mbat chd` is too busy to accept new connections or respond to client requests.

Define this parameter if you want to customize the message.

## Valid values

String, either non-empty or empty.

## Default

Not defined. By default, LSF displays the message "LSF is processing your request. Please wait..."

Batch commands retry the connection to `mbat chd` at the intervals specified by the parameters `LSB_API_CONNTIMEOUT` and `LSB_API_RECVTIMEOUT`.

# LSB\_MBD\_CONNECT\_FAIL\_MSG

## Syntax

```
LSB_MBD_CONNECT_FAIL_MSG="message_string"
```

## Description

Specifies the message displayed when internal system connections to `mbat chd` fail.

Define this parameter if you want to customize the message.

## Valid values

String, either non-empty or empty.

## Default

Not defined. By default, LSF displays the message "Cannot connect to LSF. Please wait..."

Batch commands retry the connection to `mbat chd` at the intervals specified by the parameters `LSB_API_CONNTIMEOUT` and `LSB_API_RECVTIMEOUT`.

# LSB\_MBD\_DOWN\_MSG

## Syntax

```
LSB_MBD_DOWN_MSG="message_string"
```

## Description

Specifies the message displayed by the `bhost s` command when `mbat chd` is down or there is no process listening at either the `LSB_MBD_PORT` or the `LSB_QUERY_PORT`.

Define this parameter if you want to customize the message.

## Valid values

String, either non-empty or empty.

## Default

Not defined. By default, LSF displays the message "LSF is down. Please wait..."

Batch commands retry the connection to `mbat chd` at the intervals specified by the parameters `LSB_API_CONNTIMEOUT` and `LSB_API_RECVTIMEOUT`.

## LSB\_MBD\_MAX\_SIG\_COUNT

### Syntax

**LSB\_MBD\_MAX\_SIG\_COUNT=integer**

### Description

When a host enters an unknown state, the mbatchd attempts to retry any pending jobs. This parameter specifies the maximum number of pending signals that the mbatchd deals with concurrently in order not to overload it. A high value for LSB\_MBD\_MAX\_SIG\_COUNT can negatively impact the performance of your cluster.

### Valid values

Integers between 5-100, inclusive.

### Default

5

## LSB\_MBD\_PORT

See LSF\_LIM\_PORT, LSF\_RES\_PORT, LSB\_MBD\_PORT, LSB\_SBD\_PORT.

## LSB\_MC\_CHKPNT\_RERUN

### Syntax

**LSB\_MC\_CHKPNT\_RERUN=y | n**

### Description

For checkpointable MultiCluster jobs, if a restart attempt fails, the job is rerun from the beginning (instead of from the last checkpoint) without administrator or user intervention.

The submission cluster does not need to forward the job again. The execution cluster reports the job's new pending status back to the submission cluster, and the job is dispatched to the same host to restart from the beginning.

### Default

n

## LSB\_MC\_INITFAIL\_MAIL

### Syntax

**LSB\_MC\_INITFAIL\_MAIL=Y | All | Administrator**

### Description

MultiCluster job forwarding model only.

Specify Y to make LSF email the job owner when a job is suspended after reaching the retry threshold.

Specify **Administrator** to make LSF email the primary administrator when a job is suspended after reaching the retry threshold.

Specify **All** to make LSF email both the job owner and the primary administrator when a job is suspended after reaching the retry threshold.

## Default

not defined

# LSB\_MC\_INITFAIL\_RETRY

## Syntax

**LSB\_MC\_INITFAIL\_RETRY**=*integer*

## Description

MultiCluster job forwarding model only. Defines the retry threshold and causes LSF to suspend a job that repeatedly fails to start. For example, specify 2 retry attempts to make LSF attempt to start a job 3 times before suspending it.

## Default

5

# LSB\_MEMLIMIT\_ENFORCE

## Syntax

**LSB\_MEMLIMIT\_ENFORCE**=*y | n*

## Description

Specify *y* to enable LSF memory limit enforcement.

If enabled, LSF sends a signal to kill all processes that exceed queue-level memory limits set by **MEMLIMIT** in `lsb.queues` or job-level memory limits specified by `bsub -M mem_limit`.

Otherwise, LSF passes memory limit enforcement to the OS. UNIX operating systems that support `RLIMIT_RSS` for `setrlimit()` can apply the memory limit to each process.

The following operating systems do not support memory limit at the OS level:

- Windows
- Sun Solaris 2.x

## Default

Not defined. LSF passes memory limit enforcement to the OS.

## See also

`lsb.queues`

## LSB\_MIG2PEND

### Syntax

**LSB\_MIG2PEND=0 | 1**

### Description

Applies only to migrating checkpointable or rerunnable jobs.

When defined with a value of 1, requeues migrating jobs instead of restarting or rerunning them on the first available host. Requeues the jobs in the PEND state in order of the original submission time and with the original job priority.

If you want to place the migrated jobs at the bottom of the queue without considering submission time, define both `LSB_MIG2PEND=1` and `LSB_REQUEUE_TO_BOTTOM=1` in `lsf.conf`.

Ignored in a MultiCluster environment.

### Default

Not defined. LSF restarts or reruns migrating jobs on the first available host.

### See also

`LSB_REQUEUE_TO_BOTTOM`

## LSB\_MIXED\_PATH\_DELIMITER

### Syntax

**LSB\_MIXED\_PATH\_DELIMITER="|"**

### Description

Defines the delimiter between UNIX and Windows paths if `LSB_MIXED_PATH_ENABLE=y`. For example, `/home/tmp/J.out | c:\tmp\J.out`.

### Default

A pipe "|" is the default delimiter.

### See also

`LSB_MIXED_PATH_ENABLE`

## LSB\_MIXED\_PATH\_ENABLE

### Syntax

**LSB\_MIXED\_PATH\_ENABLE=y | n**

### Description

Allows you to specify both a UNIX and Windows path when submitting a job in a mixed cluster (both Windows and UNIX hosts).

The format is always **unix\_path\_cmd|windows\_path\_cmd**.

Applies to the following options of bsub:

- -o, -oo
- -e, -eo
- -i, -is
- -cwd
- -E, -Ep
- CMD
- queue level PRE\_EXEC, POST\_EXEC
- application level PRE\_EXEC, POST\_EXEC

For example:

```
bsub -o "/home/tmp/job%J.out|c:\tmp\job%J.out" -e "/home/tmp/err%J.out|c:\tmp\err%J.out" -E "sleep 9|sleep 8" -Ep "sleep 7|sleep 6" -cwd "/home/tmp|c:\tmp" "sleep 121|sleep 122"
```

The delimiter is configurable: LSB\_MIXED\_PATH\_DELIMITER.

---

#### Note:

LSB\_MIXED\_PATH\_ENABLE doesn't support interactive mode (bsub -I).

---

## Default

Not defined. LSF jobs submitted .

## See also

LSB\_MIXED\_PATH\_DELIMITER

# LSB\_MOD\_ALL\_JOBS

## Syntax

**LSB\_MOD\_ALL\_JOBS=y | Y**

## Description

If set, enables bmod to modify resource limits and location of job output files for running jobs.

After a job has been dispatched, the following modifications can be made:

- CPU limit (-c [*hour*:]*minute*[/*host\_name* | /*host\_model*] | -cn)
- Memory limit (-M *mem\_limit* | -Mn)
- Rerunnable jobs (-r | -rn)
- Resource requirements (-R "*res\_req*" except -R "cu[*cu\_string*]")
- Run limit (-W *run\_limit*[/*host\_name* | /*host\_model*] | -Wn)
- Standard output file name (-o *output\_file* | -on)
- Standard error file name (-e *error\_file* | -en)
- Overwrite standard output (st dout) file name up to 4094 characters for UNIX or 255 characters for Windows (-oo *output\_file*)

- Overwrite standard error (`stderr`) file name up to 4094 characters for UNIX or 255 characters for Windows (`-eo error_file`)

To modify the CPU limit or the memory limit of running jobs, the parameters `LSB_JOB_CPULIMIT=Y` and `LSB_JOB_MEMLIMIT=Y` must be defined in `lsf.conf`.

---

### Important:

Always run `badmi n mbdrestart` after modifying `LSB_MOD_ALL_JOBS`.

---

## Default

Not defined

## See also

`LSB_JOB_CPULIMIT`, `LSB_JOB_MEMLIMIT`

# LSB\_NCPU\_ENFORCE

## Description

When set to 1, enables parallel fairshare and considers the number of CPUs when calculating dynamic priority for queue-level user-based fairshare. `LSB_NCPU_ENFORCE` does not apply to host-partition user-based fairshare. For host-partition user-based fairshare, the number of CPUs is automatically considered.

## Default

Not defined

# LSB\_NQS\_PORT

## Syntax

`LSB_NQS_PORT=port_number`

## Description

Required for LSF to work with NQS.

TCP service port to use for communication with NQS.

## Where defined

This parameter can alternatively be set as an environment variable or in the services database such as `/etc/services`.

## Example

`LSB_NQS_PORT=607`

## Default

Not defined

## LSB\_NUM\_NIOS\_CALLBACK\_THREADS

### Syntax

**LSB\_NUM\_NIOS\_CALLBACK\_THREADS=*integer***

### Description

Specifies the number of callback threads to use for batch queries.

If your cluster runs a large amount of blocking mode (bsub -K) and interactive jobs (bsub -I), response to batch queries can become very slow. If you run large number of bsub -I or bsub -K jobs, you can define the threads to the number of processors on the master host.

### Default

Not defined

## LSB\_PACK\_MESUB

### Syntax

**LSB\_PACK\_MESUB=Y|y|N|n**

### Description

Applies to job packs only.

If LSB\_PACK\_MESUB=N, mesub will not be executed for any jobs in the job submission file, even if there are esubs configured at the application level (-a option of bsub), or using LSB\_ESUB\_METHOD in `lsf.conf`, or through a named esub executable under LSF\_SERVERDIR.

If LSB\_PACK\_MESUB=Y, mesub is executed for every job in the job submission file.

Parameters related to job packs are not supported as environment variables.

### Default

Y

## LSB\_PACK\_SKIP\_ERROR

### Syntax

**LSB\_PACK\_SKIP\_ERROR=Y|y|N|n**

### Description

Applies to job packs only.

If LSB\_PACK\_SKIP\_ERROR=Y, all requests in the job submission file are submitted, even if some of the job submissions fail. The job submission process always continues to the end of the file.

If LSB\_PACK\_SKIP\_ERROR=N, job submission stops if one job submission fails. The remaining requests in the job submission file are not submitted.

If you change the configuration of this parameter, you must restart `mbat chd`.

Parameters related to job packs are not supported as environment variables.

## Default

N

# LSB\_PSET\_BIND\_DEFAULT

## Syntax

**LSB\_PSET\_BIND\_DEFAULT=y | Y**

## Description

If set, LSF binds a job that is not explicitly associated with an HP-UX pset to the default pset 0. If `LSB_PSET_BIND_DEFAULT` is not set, LSF must still attach the job to a pset, and so binds the job to the same pset used by the LSF HPC daemons.

Use `LSB_PSET_BIND_DEFAULT` to improve LSF daemon performance by automatically unbinding a job with no pset options from the pset used by the LSF daemons, and binding it to the default pset.

## Default

Not defined

# LSB\_QUERY\_PORT

## Syntax

**LSB\_QUERY\_PORT=*port\_number***

## Description

Optional. Applies only to UNIX platforms that support thread programming.

When using MultiCluster, `LSB_QUERY_PORT` must be defined on all clusters.

This parameter is recommended for busy clusters with many jobs and frequent query requests to increase `mbat chd` performance when you use the `bj obs` command.

This may indirectly increase overall `mbat chd` performance.

The `port_number` is the TCP/IP port number to be used by `mbat chd` to only service query requests from the LSF system. `mbat chd` checks the query port during initialization.

If `LSB_QUERY_PORT` *is not* defined:

- `mbat chd` uses the port specified by `LSB_MBD_PORT` in `lsf.conf`, or, if `LSB_MBD_PORT` is not defined, looks into the system services database for port numbers to communicate with other hosts in the cluster.
- For each query request it receives, `mbat chd` forks one child `mbat chd` to service the request. Each child `mbat chd` processes one request and then exits.

If `LSB_QUERY_PORT` is defined:

- `mbat chd` prepares this port for connection. The default behavior of `mbat chd` changes, a child `mbat chd` is forked, and the child `mbat chd` creates threads to process requests.

- `mbat chd` responds to requests by forking one child `mbat chd`. As soon as `mbat chd` has forked a child `mbat chd`, the child `mbat chd` takes over and listens on the port to process more query requests. For each request, the child `mbat chd` creates a thread to process it.

The interval used by `mbat chd` for forking new child `mbat chds` is specified by the parameter `MBD_REFRESH_TIME` in `l sb. params`.

The child `mbat chd` continues to listen to the port number specified by `LSB_QUERY_PORT` and creates threads to service requests until the job changes status, a new job is submitted, or the time specified in `MBD_REFRESH_TIME` in `l sb. params` has passed (see `MBD_REFRESH_TIME` in `l sb. params` for more details). When any of these happens, the parent `mbat chd` sends a message to the child `mbat chd` to exit.

`LSB_QUERY_PORT` must be defined when `NEWJOB_REFRESH=Y` in `l sb. params` to enable a child `mbat chd` to get up to date information about new jobs from the parent `mbat chd`.

## Operating system support

### Tip:

See the Online Support area of the Platform Computing Web site at [www.platform.com](http://www.platform.com) for the latest information about operating systems that support multithreaded `mbat chd`.

## Default

Not defined

## See also

`MBD_REFRESH_TIME` and `NEWJOB_REFRESH` in `l sb. params`

# LSB\_REQUEUE\_TO\_BOTTOM

## Syntax

`LSB_REQUEUE_TO_BOTTOM=0 | 1`

## Description

Specify 1 to put automatically requeued jobs at the bottom of the queue instead of at the top. Also requeues migrating jobs to the bottom of the queue if `LSB_MIG2PEND` is also defined with a value of 1.

Specify 0 to requeue jobs to the top of the queue.

Ignored in a MultiCluster environment.

## Default

0 (LSF requeues jobs to the top of the queue).

## See also

`LSB_MIG2PEND`, `REQUEUE_EXIT_VALUES` in `l sb. queues`

## LSB\_RLA\_PORT

### Syntax

**LSB\_RLA\_PORT**=*port\_number*

### Description

TCP port used for communication between the LSF topology adapter (RLA) and the HPC scheduler plugin.

### Default

6883

## LSB\_RLA\_UPDATE

### Syntax

**LSB\_RLA\_UPDATE**=*time\_seconds*

### Description

Specifies how often the HPC scheduler refreshes free node information from the LSF topology adapter (RLA).

### Default

600 seconds

## LSB\_RLA\_WORKDIR

### Syntax

**LSB\_RLA\_WORKDIR**=*directory*

### Description

Directory to store the LSF topology adapter (RLA) status file. Allows RLA to recover its original state when it restarts. When RLA first starts, it creates the directory defined by LSB\_RLA\_WORKDIR if it does not exist, then creates subdirectories for each host.

You should avoid using `/tmp` or any other directory that is automatically cleaned up by the system. Unless your installation has restrictions on the LSB\_SHAREDIR directory, you should use the default for LSB\_RLA\_WORKDIR.

### Default

LSB\_SHAREDIR/*cluster\_name*/rla\_workdir

## LSB\_SACCT\_ONE\_UG

### Syntax

**LSB\_SACCT\_ONE\_UG**=y | Y | n | N

## Description

When set to Y, minimizes overall memory usage of `mbat chd` during fairshare accounting at job submission by limiting the number of share account nodes created on `mbat chd` startup. Most useful when there are a lot of user groups with a 1 member in the fairshare policy.

When a default user group is defined, inactive user share accounts are still defined for the default user group.

When setting this parameter, you must restart the `mbat chd`.

## Default

N

## LSB\_SBD\_PORT

See LSF\_LIM\_PORT, LSF\_RES\_PORT, LSB\_MBD\_PORT, LSB\_SBD\_PORT.

## LSB\_SET\_TMPDIR

### Syntax

**LSB\_SET\_TMPDIR=y | n**

If y, LSF sets the TMPDIR environment variable, overwriting the current value with `/tmp/job_ID.tmpdir`.

### Default

n

## LSB\_SHAREDIR

### Syntax

**LSB\_SHAREDIR=directory**

## Description

Directory in which the job history and accounting logs are kept for each cluster. These files are necessary for correct operation of the system. Like the organization under LSF\_CONFDIR, there is one subdirectory for each cluster.

The LSB\_SHAREDIR directory must be owned by the LSF administrator. It must be accessible from all hosts that can potentially become the master host, and must allow read and write access from the master host.

The LSB\_SHAREDIR directory typically resides on a reliable file server.

## Default

LSF\_INDEP/work

## See also

LSB\_LOCALDIR

# LSB\_SHORT\_HOSTLIST

## Syntax

```
LSB_SHORT_HOSTLIST=1
```

## Description

Displays an abbreviated list of hosts in `bj obs` and `bhi st` for a parallel job where multiple processes of a job are running on a host. Multiple processes are displayed in the following format:

```
processes*hostA
```

For example, if a parallel job is running 5 processes on host A, the information is displayed in the following manner:

```
5*hostA
```

Setting this parameter may improve `mbat chd` restart performance and accelerate event replay.

## Default

Not defined

# LSB\_SIGSTOP

## Syntax

```
LSB_SIGSTOP=signal_name | signal_value
```

## Description

Specifies the signal sent by the SUSPEND action in LSF. You can specify a signal name or a number.

If this parameter is not defined, by default the SUSPEND action in LSF sends the following signals to a job:

- Parallel or interactive jobs: SIGTSTP is sent to allow user programs to catch the signal and clean up. The parallel job launcher also catches the signal and stops the entire job (task by task for parallel jobs). Once LSF sends SIGTSTP, LSF assumes the job is stopped.
- Other jobs: SIGSTOP is sent. SIGSTOP cannot be caught by user programs. The same set of signals is not supported on all UNIX systems. To display a list of the symbolic names of the signals (without the SIG prefix) supported on your system, use the `kill -l` command.

## Example

```
LSB_SIGSTOP=SIGKILL
```

In this example, the SUSPEND action sends the three default signals sent by the TERMINATE action (SIGINT, SIGTERM, and SIGKILL) 10 seconds apart.

## Default

Not defined. Default SUSPEND action in LSF is sent.

## LSB\_SSH\_XFORWARD\_CMD

### Syntax

**LSB\_SSH\_XFORWARD\_CMD**=[/path[/path]]ssh command [ssh options]

### Description

Optional when submitting jobs with SSH X11 forwarding. Allows you to specify an SSH command and options when a job is submitted with -XF.

Replace the default value with an SSH command (full PATH and options allowed).

When running a job with the -XF option, runs the SSH command specified here.

### Default

ssh -X -n

## LSB\_STDOUT\_DIRECT

### Syntax

**LSB\_STDOUT\_DIRECT**=y | Y

### Description

When set, and used with the -o or -e options of bsub, redirects standard output or standard error from the job directly to a file as the job runs.

If LSB\_STDOUT\_DIRECT is not set and you use the bsub -o option, the standard output of a job is written to a temporary file and copied to the file you specify *after* the job finishes.

LSB\_STDOUT\_DIRECT is not supported on Windows.

### Default

Not defined

## LSB\_STOP\_IGNORE\_IT

### Usage

**LSB\_STOP\_IGNORE\_IT**= Y | y

### Description

Allows a solitary job to be stopped regardless of the idle time (IT) of the host that the job is running on. By default, if only one job is running on a host, the host idle time must be zero in order to stop the job.

### Default

Not defined

# LSB\_SUB\_COMMANDNAME

## Syntax

`LSB_SUB_COMMANDNAME=y | Y`

## Description

If set, enables `esub` to use the variable `LSB_SUB_COMMAND_LINE` in the `esub` job parameter file specified by the `$LSB_SUB_PARM_FILE` environment variable.

The `LSB_SUB_COMMAND_LINE` variable carries the value of the `bsub` command argument, and is used when `esub` runs.

## Example

`esub` contains:

```
#!/bin/sh . $LSB_SUB_PARM_FILE exec 1>&2 if [ $LSB_SUB_COMMAND_LINE="netscape" ]; then
echo "netscape is not allowed to run in batch mode" exit $LSB_SUB_ABORT_VALUE fi
```

`LSB_SUB_COMMAND_LINE` is defined in `$LSB_SUB_PARM_FILE` as:

```
LSB_SUB_COMMAND_LINE=netscape
```

A job submitted with:

```
bsub netscape ...
```

Causes `esub` to echo the message:

```
netscape is not allowed to run in batch mode
```

## Default

Not defined

## See also

`LSB_SUB_COMMAND_LINE` and `LSB_SUB_PARM_FILE` environment variables

# LSB\_SUBK\_SHOW\_EXEC\_HOST

## Syntax

`LSB_SUBK_SHOW_EXEC_HOST=Y | N`

## Description

When enabled, displays the execution host in the output of the command `bsub -K`. If the job runs on multiple hosts, only the first execution host is shown.

In a MultiCluster environment, this parameter must be set in both clusters.

### Tip:

Restart `sbat chd` on the execution host to make changes take effect.

## Default

N

## LSB\_TIME\_CMD

### Syntax

**LSB\_TIME\_CMD**=*timing\_level*

### Description

The timing level for checking how long batch commands run.  
Time usage is logged in milliseconds; specify a positive integer.  
Example: LSB\_TIME\_CMD=1

### Default

Not defined

### See also

LSB\_TIME\_MBD, LSB\_TIME\_SBD, LSF\_TIME\_LIM, LSF\_TIME\_RES

## LSB\_TIME\_MBD

### Syntax

**LSB\_TIME\_MBD**=*timing\_level*

### Description

The timing level for checking how long mbat chd routines run.  
Time usage is logged in milliseconds; specify a positive integer.  
Example: LSB\_TIME\_MBD=1

### Default

Not defined

### See also

LSB\_TIME\_CMD, LSB\_TIME\_SBD, LSF\_TIME\_LIM, LSF\_TIME\_RES

## LSB\_TIME\_RESERVE\_NUMJOBS

### Syntax

**LSB\_TIME\_RESERVE\_NUMJOBS**=*maximum\_reservation\_jobs*

### Description

Enables time-based slot reservation. The value must be positive integer.  
LSB\_TIME\_RESERVE\_NUMJOBS controls maximum number of jobs using time-based slot reservation.  
For example, if LSB\_TIME\_RESERVE\_NUMJOBS=4, only the top 4 jobs get their future allocation information.

Use `LSB_TIME_RESERVE_NUMJOBS=1` to allow only the highest priority job to get accurate start time prediction.

## Recommended value

3 or 4 is the recommended setting. Larger values are not as useful because after the first pending job starts, the estimated start time of remaining jobs may be changed.

## Default

Not defined

# LSB\_TIME\_SBD

## Syntax

`LSB_TIME_SBD=timing_level`

## Description

The timing level for checking how long `sbat chd` routines run.

Time usage is logged in milliseconds; specify a positive integer.

Example: `LSB_TIME_SBD=1`

## Default

Not defined

## See also

`LSB_TIME_CMD`, `LSB_TIME_MBD`, `LSF_TIME_LIM`, `LSF_TIME_RES`

# LSB\_TIME\_SCH

## Syntax

`LSB_TIME_SCH=timing_level`

## Description

The timing level for checking how long `mbschd` routines run.

Time usage is logged in milliseconds; specify a positive integer.

Example: `LSB_TIME_SCH=1`

## Default

Not defined

# LSB\_UTMP

## Syntax

`LSB_UTMP=y | Y`

## Description

If set, enables registration of user and account information for interactive batch jobs submitted with `bsub -I p` or `bsub -I s`. To disable `utmp` file registration, set `LSB_UTMP` to any value other than `y` or `Y`; for example, `LSB_UTMP=N`.

LSF registers interactive batch jobs the job by adding a entries to the `utmp` file on the execution host when the job starts. After the job finishes, LSF removes the entries for the job from the `utmp` file.

## Limitations

Registration of `utmp` file entries is supported on the following platforms:

- SGI IRIX (6.4 and later)
- Solaris (all versions)
- HP-UX (all versions)
- Linux (all versions)

`utmp` file registration is not supported in a MultiCluster environment.

Because interactive batch jobs submitted with `bsub -I` are not associated with a pseudo-terminal, `utmp` file registration is not supported for these jobs.

## Default

Not defined

# LSF\_AFS\_CELLNAME

## Syntax

**LSF\_AFS\_CELLNAME**=*AFS\_cell\_name*

## Description

Must be defined to AFS cell name if the AFS file system is in use.

Example:

```
LSF_AFS_CELLNAME=xxx.ch
```

## Default

Not defined

# LSF\_AM\_OPTIONS

## Syntax

**LSF\_AM\_OPTIONS**=**AMFIRST** | **AMNEVER**

## Description

Determines the order of file path resolution when setting the user's home directory.

This variable is rarely used but sometimes LSF does not properly change the directory to the user's home directory when the user's home directory is automounted. Setting `LSF_AM_OPTIONS` forces LSF to change directory to `$HOME` before attempting to automount the user's home.

When this parameter is not defined or set to AMFIRST, LSF, sets the user's \$HOME directory from the automount path. If it cannot do so, LSF sets the user's \$HOME directory from the passwd file.

When this parameter is set to AMNEVER, LSF, never uses automount to set the path to the user's home. LSF sets the user's \$HOME directory directly from the passwd file.

## Valid values

The two values are AMFIRST and AMNEVER

## Default

Same as AMFIRST

# LSF\_API\_CONNTIMEOUT

## Syntax

**LSF\_API\_CONNTIMEOUT**=*time\_seconds*

## Description

Timeout when connecting to LIM.

## EGO parameter

EGO\_LIM\_CONNTIMEOUT

## Default

5

## See also

LSF\_API\_RECVTIMEOUT

# LSF\_API\_RECVTIMEOUT

## Syntax

**LSF\_API\_RECVTIMEOUT**=*time\_seconds*

## Description

Timeout when receiving a reply from LIM.

## EGO parameter

EGO\_LIM\_RECVTIMEOUT

## Default

20

## See also

LSF\_API\_CONNTIMEOUT

# LSF\_AUTH

## Syntax

**LSF\_AUTH=eauth | ident**

## Description

Enables either external authentication or authentication by means of identification daemons. This parameter is required for any cluster that contains Windows hosts, and is optional for UNIX-only clusters. After defining or changing the value of LSF\_AUTH, you must shut down and restart the LSF daemons on all server hosts to apply the new authentication method.

### eauth

For site-specific customized external authentication. Provides the highest level of security of all LSF authentication methods.

### ident

For authentication using the RFC 931/1413/1414 protocol to verify the identity of the remote client. If you want to use `ident` authentication, you must download and install the `ident` protocol, available from the public domain, and register `ident` as required by your operating system.

For UNIX-only clusters, privileged ports authentication (`setuid`) can be configured by commenting out or deleting the LSF\_AUTH parameter. If you choose privileged ports authentication, LSF commands must be installed as `setuid` programs owned by `root`. If the commands are installed in an NFS-mounted shared file system, the file system must be mounted with `setuid` execution allowed, that is, without the `nosuid` option.

---

### Restriction:

To enable privileged ports authentication, LSF\_AUTH must not be defined; `setuid` is not a valid value for LSF\_AUTH.

---

## Default

`eauth`

During LSF installation, a default `eauth` executable is installed in the directory specified by the parameter LSF\_SERVERDIR in the `lsf.conf` file. The default executable provides an example of how the `eauth` protocol works. You should write your own `eauth` executable to meet the security requirements of your cluster.

# LSF\_ASPLUGIN

## Syntax

**LSF\_ASPLUGIN=*path***

## Description

Points to the SGI Array Services library `libarray.so`. The parameter only takes effect on 64-bit x-86 Linux 2.6, glibc 2.3.

## Default

`/usr/lib64/libarray.so`

# LSF\_AUTH\_DAEMONS

## Syntax

`LSF_AUTH_DAEMONS=y | Y`

## Description

Enables LSF daemon authentication when external authentication is enabled (`LSF_AUTH=eauth` in the file `lsf.conf`). Daemons invoke `eauth` to authenticate each other as specified by the `eauth` executable.

## Default

Not defined.

# LSF\_BINDIR

## Syntax

`LSF_BINDIR=directory`

## Description

Directory in which all LSF user commands are installed.

## Default

`LSF_MACHDEP/bin`

# LSF\_BIND\_JOB

## Syntax

`LSF_BIND_JOB=NONE | BALANCE | PACK | ANY | USER | USER_CPU_LIST`

## Description

Specifies the processor binding policy for sequential and parallel job processes that run on a single host.

On Linux execution hosts that support this feature, job processes are hard bound to selected processors.

If processor binding feature is not configured with the `BIND_JOB` parameter in an application profile in `lsb.applications`, the `lsf.conf` configuration setting takes effect. The application profile configuration for processor binding overrides the `lsf.conf` configuration.

For backwards compatibility:

- `LSF_BIND_JOB=Y` is interpreted as `LSF_BIND_JOB=BALANCE`
- `LSF_BIND_JOB=N` is interpreted as `LSF_BIND_JOB=NONE`

## Supported platforms

Linux with kernel version 2.6 or higher

## Default

Not defined. Processor binding is disabled.

# LSF\_BMPLUGIN

## Syntax

**LSF\_BMPLUGIN**=*path*

## Description

Points to the bitmask library `libbitmask.so`. The parameter only takes effect on 64-bit x-86 Linux 2.6, glibc 2.3.

## Default

`/usr/lib64/libbitmask.so`

# LSF\_CMD\_LOGDIR

## Syntax

**LSF\_CMD\_LOGDIR**=*path*

## Description

The path to the log files used for debugging LSF commands.

This parameter can also be set from the command line.

## Default

`/tmp`

## See also

LSB\_CMD\_LOG\_MASK, LSB\_CMD\_LOGDIR, LSB\_DEBUG, LSB\_DEBUG\_CMD,  
LSB\_TIME\_CMD, LSF\_CMD\_LOG\_MASK, LSF\_LOG\_MASK, LSF\_LOGDIR, LSF\_TIME\_CMD

# LSF\_CMD\_LOG\_MASK

## Syntax

**LSF\_CMD\_LOG\_MASK**=*log\_level*

## Description

Specifies the logging level of error messages from LSF commands.

For example:

```
LSF_CMD_LOG_MASK=LOG_DEBUG
```

To specify the logging level of error messages, use `LSB_CMD_LOG_MASK`. To specify the logging level of error messages for LSF daemons, use `LSF_LOG_MASK`.

LSF commands log error messages in different levels so that you can choose to log all messages, or only log messages that are deemed critical. The level specified by `LSF_CMD_LOG_MASK` determines which

messages are recorded and which are discarded. All messages logged at the specified level or higher are recorded, while lower level messages are discarded.

For debugging purposes, the level LOG\_DEBUG contains the fewest number of debugging messages and is used for basic debugging. The level LOG\_DEBUG3 records all debugging messages, and can cause log files to grow very large; it is not often used. Most debugging is done at the level LOG\_DEBUG2.

The commands log to the syslog facility unless LSF\_CMD\_LOGDIR is set.

## Valid values

The log levels from highest to lowest are:

- LOG\_EMERG
- LOG\_ALERT
- LOG\_CRIT
- LOG\_ERR
- LOG\_WARNING
- LOG\_NOTICE
- LOG\_INFO
- LOG\_DEBUG
- LOG\_DEBUG1
- LOG\_DEBUG2
- LOG\_DEBUG3

## Default

LOG\_WARNING

## See also

LSB\_CMD\_LOG\_MASK, LSB\_CMD\_LOGDIR, LSB\_DEBUG, LSB\_DEBUG\_CMD,  
LSB\_TIME\_CMD, LSB\_CMD\_LOGDIR, LSF\_LOG\_MASK, LSF\_LOGDIR, LSF\_TIME\_CMD

# LSF\_CONF\_RETRY\_INT

## Syntax

**LSF\_CONF\_RETRY\_INT**=*time\_seconds*

## Description

The number of seconds to wait between unsuccessful attempts at opening a configuration file (only valid for LIM). This allows LIM to tolerate temporary access failures.

## EGO parameter

EGO\_CONF\_RETRY\_INT

## Default

30

## See also

LSF\_CONF\_RETRY\_MAX

# LSF\_CONF\_RETRY\_MAX

## Syntax

**LSF\_CONF\_RETRY\_MAX**=*integer*

## Description

The maximum number of retry attempts by LIM to open a configuration file. This allows LIM to tolerate temporary access failures. For example, to allow one more attempt after the first attempt has failed, specify a value of 1.

## EGO parameter

EGO\_CONF\_RETRY\_MAX

## Default

0

## See also

LSF\_CONF\_RETRY\_INT

# LSF\_CONFDIR

## Syntax

**LSF\_CONFDIR**=*directory*

## Description

Directory in which all LSF configuration files are installed. These files are shared throughout the system and should be readable from any host. This directory can contain configuration files for more than one cluster.

The files in the LSF\_CONFDIR directory must be owned by the primary LSF administrator, and readable by all LSF server hosts.

If live reconfiguration through the `bconf` command is enabled by the parameter `LSF_LIVE_CONFDIR`, configuration files are written to and read from the directory set by `LSF_LIVE_CONFDIR`.

## Default

LSF\_INDEP/conf

## See also

LSB\_CONFDIR, LSF\_LIVE\_CONFDIR

# LSF\_CPUSETLIB

## Syntax

**LSF\_CPUSETLIB**=*path*

## Description

Points to the SGI cpuset library `libcpuset.so`. The parameter only takes effect on 64-bit x-86 Linux 2.6, glibc 2.3.

## Default

`/usr/lib64/libcpuset.so`

## LSF\_CRASH\_LOG

## Syntax

**LSF\_CRASH\_LOG=Y | N**

## Description

On Linux hosts only, enables logging when or if a daemon crashes. Relies on the Linux debugger (gdb). Two log files are created, one for the root daemons (res, lim, sbd, and mbatchd) in `/tmp/lsf_root_daemons_crash.log` and one for administrative daemons (mbschd) in `/tmp/lsf_admin_daemons_crash.log`.

File permissions for both files are 600.

If enabling, you must restart the daemons for the change to take effect.

## Default

N (no log files are created for daemon crashes)

## LSF\_DAEMONS\_CPUS

## Syntax

**LSF\_DAEMONS\_CPUS="*mbatchd\_cpu\_list:mbschd\_cpu\_list*"**

### *mbatchd\_cpu\_list*

Defines the list of master host CPUs where the `mbatchd` daemon processes can run (hard CPU affinity). Format the list as a white-space delimited list of CPU numbers.

### *mbschd\_cpu\_list*

Defines the list of master host CPUs where the `mbschd` daemon processes can run. Format the list as a white-space delimited list of CPU numbers.

## Description

By default, `mbatchd` and `mbschd` can run on any CPUs. If `LSF_DAEMONS_CPUS` is set, they only run on a specified list of CPUs. An empty list means LSF daemons can run on any CPUs. Use spaces to separate multiple CPUs.

The operating system can assign other processes to run on the same CPU; however, if utilization of the bound CPU is lower than utilization of the unbound CPUs.

## Related parameters

To improve scheduling and dispatch performance of all LSF daemons, you should use `LSF_DAEMONS_CPUS` together with `EGO_DAEMONS_CPUS` (in `ego.conf` or `lsf.conf`), which controls LIM CPU allocation, and `MBD_QUERY_CPUS`, which binds `mbact chd` query processes to specific CPUs so that higher priority daemon processes can run more efficiently. To get best performance, CPU allocation for all four daemons should be assigned their own CPUs. For example, on a 4 CPU SMP host, the following configuration gives the best performance:

```
EGO_DAEMONS_CPUS=0 LSF_DAEMONS_CPUS=1: 2 MBD_QUERY_CPUS=3
```

## Examples

If you specify

```
LSF_DAEMONS_CPUS="1:2"
```

the `mbat chd` processes run only on CPU number 1 on the master host, and `mbschd` run only on CPU number 2.

If you specify

```
LSF_DAEMONS_CPUS="1 2:1 2"
```

both `mbat chd` and `mbschd` run CPU 1 and CPU 2.

## Important

You can specify CPU affinity only for master hosts that use one of the following operating systems:

- Linux 2.6 or higher
- Solaris 8 or higher

## EGO parameter

`LSF_DAEMONS_CPUS=lim_cpu_list`: run the EGO LIM daemon on the specified CPUs.

## Default

Not defined

## See also

`MBD_QUERY_CPUS` in `lsb.params`

# LSF\_DAEMON\_WRAP

## Syntax

```
LSF_DAEMON_WRAP=y | Y
```

## Description

Applies to Kerberos, DCE/DFS and AFS environments; if you are using LSF with DCE, AFS, or Kerberos, set this parameter to `y` or `Y`.

When this parameter is set to `y` or `Y`, `mbat chd`, `sbat chd`, and `RES` run the executable daemons. `wrap` located in `LSF_SERVERDIR`.

## Default

Not defined. LSF does not run the daemons. wrap executable.

# LSF\_DEBUG\_CMD

## Syntax

**LSF\_DEBUG\_CMD**=*log\_class*

## Description

Sets the debugging log class for LSF commands and APIs.

Specifies the log class filtering to be applied to LSF commands or the API. Only messages belonging to the specified log class are recorded.

LSF\_DEBUG\_CMD sets the log class and is used in combination with LSF\_CMD\_LOG\_MASK, which sets the log level. For example:

```
LSF_CMD_LOG_MASK=LOG_DEBUG LSF_DEBUG_CMD="LC_TRACE LC_EXEC"
```

Debugging is turned on when you define both parameters.

The daemons log to the `syslog` facility unless LSF\_CMD\_LOGDIR is defined.

To specify multiple log classes, use a space-separated list enclosed by quotation marks. For example:

```
LSF_DEBUG_CMD="LC_TRACE LC_EXEC"
```

Can also be defined from the command line.

## Valid values

Valid log classes are:

- LC\_AFS and LC2\_AFS: Log AFS messages
- LC\_AUTH and LC2\_AUTH: Log authentication messages
- LC\_CHKPNT and LC2\_CHKPNT: Log checkpointing messages
- LC\_COMM and LC2\_COMM: Log communication messages
- LC\_DCE and LC2\_DCE: Log messages pertaining to DCE support
- LC\_EEVENTD and LC2\_EEVENTD: Log eeventd messages
- LC\_ELIM and LC2\_ELIM: Log ELIM messages
- LC\_EXEC and LC2\_EXEC: Log significant steps for job execution
- LC\_FAIR - Log fairshare policy messages
- LC\_FILE and LC2\_FILE: Log file transfer messages
- LC\_HANG and LC2\_HANG: Mark where a program might hang
- LC\_JARRAY and LC2\_JARRAY: Log job array messages
- LC\_JLIMIT and LC2\_JLIMIT: Log job slot limit messages
- LC\_LICENSE and LC2\_LICENSE : Log license management messages (LC\_LICENSE is also supported for backward compatibility)
- LC\_LOADINDX and LC2\_LOADINDX: Log load index messages
- LC\_M\_LOG and LC2\_M\_LOG: Log multievent logging messages
- LC\_MPI and LC2\_MPI: Log MPI messages
- LC\_MULTI and LC2\_MULTI: Log messages pertaining to MultiCluster
- LC\_PEND and LC2\_PEND: Log messages related to job pending reasons

- LC\_PERFM and LC2\_PERFM: Log performance messages
- LC\_PIM and LC2\_PIM: Log PIM messages
- LC\_PREEMPT and LC2\_PREEMPT: Log preemption policy messages
- LC\_RESREQ and LC2\_RESREQ: Log resource requirement messages
- LC\_SIGNAL and LC2\_SIGNAL: Log messages pertaining to signals
- LC\_SYS and LC2\_SYS: Log system call messages
- LC\_TRACE and LC2\_TRACE: Log significant program walk steps
- LC\_XDR and LC2\_XDR: Log everything transferred by XDR

## Default

Not defined

## See also

LSF\_CMD\_LOG\_MASK, LSF\_CMD\_LOGDIR, LSF\_DEBUG\_LIM, LSF\_DEBUG\_RES, LSF\_LIM\_PORT, LSF\_RES\_PORT, LSB\_MBD\_PORT, LSB\_SBD\_PORT, LSF\_LOGDIR, LSF\_LIM\_DEBUG, LSF\_RES\_DEBUG

# LSF\_DEBUG\_LIM

## Syntax

**LSF\_DEBUG\_LIM**=*log\_class*

## Description

Sets the log class for debugging LIM.

Specifies the log class filtering to be applied to LIM. Only messages belonging to the specified log class are recorded.

The LSF\_DEBUG\_LIM sets the log class and is used in combination with EGO\_LOG\_MASK in `ego.conf`, which sets the log level.

For example, in `ego.conf`:

```
EGO_LOG_MASK=LOG_DEBUG
```

and in `lsf.conf`:

```
LSF_DEBUG_LIM=LC_TRACE
```

---

### Important:

If EGO is enabled, LSF\_LOG\_MASK no longer specifies LIM logging level. Use EGO\_LOG\_MASK in `ego.conf` to control message logging for LIM. The default value for EGO\_LOG\_MASK is LOG\_WARNING.

---

You need to restart the daemons after setting LSF\_DEBUG\_LIM for your changes to take effect.

If you use the command `lsadmin limdebug` to temporarily change this parameter without changing `lsf.conf`, you do not need to restart the daemons.

To specify multiple log classes, use a space-separated list enclosed in quotation marks. For example:

```
LSF_DEBUG_LIM="LC_TRACE LC_EXEC"
```

This parameter can also be defined from the command line.

## Valid values

Valid log classes are:

- LC\_AFS and LC2\_AFS: Log AFS messages
- LC\_AUTH and LC2\_AUTH: Log authentication messages
- LC\_CHKPNT - log checkpointing messages
- LC\_COMM and LC2\_COMM: Log communication messages
- LC\_DCE and LC2\_DCE: Log messages pertaining to DCE support
- LC\_EXEC and LC2\_EXEC: Log significant steps for job execution
- LC\_FILE and LC2\_FILE: Log file transfer messages
- LC\_HANG and LC2\_HANG: Mark where a program might hang
- LC\_JGRP - Log job group messages
- LC\_LICENSE and LC2\_LICENSE : Log license management messages (LC\_LICENSE is also supported for backward compatibility)
- LC\_LICSCHEM - Log License Scheduler messages
- LC\_MEMORY - Log memory limit messages
- LC\_MULTI and LC2\_MULTI: Log messages pertaining to MultiCluster
- LC\_PIM and LC2\_PIM: Log PIM messages
- LC\_RESOURCE - Log resource broker messages
- LC\_SIGNAL and LC2\_SIGNAL: Log messages pertaining to signals
- LC\_TRACE and LC2\_TRACE: Log significant program walk steps
- LC\_XDR and LC2\_XDR: Log everything transferred by XDR

## EGO parameter

EGO\_DEBUG\_LIM

## Default

Not defined

## See also

LSF\_DEBUG\_RES, LSF\_CMD\_LOGDIR, LSF\_CMD\_LOG\_MASK, LSF\_LOG\_MASK, LSF\_LOGDIR

# LSF\_DEBUG\_RES

## Syntax

**LSF\_DEBUG\_RES**=*log\_class*

## Description

Sets the log class for debugging RES.

Specifies the log class filtering to be applied to RES. Only messages belonging to the specified log class are recorded.

LSF\_DEBUG\_RES sets the log class and is used in combination with LSF\_LOG\_MASK, which sets the log level. For example:

```
LSF_LOG_MASK=LOG_DEBUG LSF_DEBUG_RES=LC_TRACE
```

To specify multiple log classes, use a space-separated list enclosed in quotation marks. For example:

```
LSF_DEBUG_RES="LC_TRACE LC_EXEC"
```

You need to restart the daemons after setting `LSF_DEBUG_RES` for your changes to take effect.

If you use the command `lsadmin resdebug` to temporarily change this parameter without changing `lsf.conf`, you do not need to restart the daemons.

## Valid values

For a list of valid log classes see `LSF_DEBUG_LIM`

## Default

Not defined

## See also

`LSF_DEBUG_LIM`, `LSF_CMD_LOGDIR`, `LSF_CMD_LOG_MASK`, `LSF_LOG_MASK`, `LSF_LOGDIR`

# LSF\_DHCP\_ENV

## Syntax

```
LSF_DHCP_ENV=y
```

## Description

If defined, enables dynamic IP addressing for all LSF client hosts in the cluster.

Dynamic IP addressing is not supported across clusters in a MultiCluster environment.

If you set `LSF_DHCP_ENV`, you must also specify `LSF_DYNAMIC_HOST_WAIT_TIME` in order for hosts to rejoin a cluster after their IP address changes.

---

### Tip:

After defining or changing this parameter, you must run `lsadmin reconfig` and `lsadmin mbdrestart` to restart all LSF daemons.

---

## EGO parameter

```
EGO_DHCP_ENV
```

## Default

Not defined

## See also

`LSF_DYNAMIC_HOST_WAIT_TIME`

# LSF\_DISABLE\_LSRUN

## Syntax

```
LSF_DISABLE_LSRUN=y | Y
```

## Description

When defined, RES refuses remote connections from `l srun` and `l sgrun` unless the user is either an LSF administrator or root. For remote execution by root, `LSF_ROOT_REX` must be defined.

Other remote execution commands, such as `ch` and `l smake` are not affected.

## Default

Not defined

# LSF\_DISPATCHER\_LOGDIR

## Syntax

`LSF_DISPATCHER_LOGDIR=path`

## Description

Specifies the path to the log files for slot allocation decisions for queue-based fairshare.

If defined, LSF writes the results of its queue-based fairshare slot calculation to the specified directory. Each line in the file consists of a timestamp for the slot allocation and the number of slots allocated to each queue under its control. LSF logs in this file every minute. The format of this file is suitable for plotting with `gnuplot`.

## Example

```
# clients managed by LSF
# Roma # Verona # Genova # Pi sa # Venezi a # Bol o gna
15/3      19: 4: 50    0 0 0 0 0 0
15/3      19: 5: 51    8 5 2 5 2 0
15/3      19: 6: 51    8 5 2 5 5 1
15/3      19: 7: 53    8 5 2 5 5 5
15/3      19: 8: 54    8 5 2 5 5 0
15/3      19: 9: 55    8 5 0 5 4 2
```

The queue names are in the header line of the file. The columns correspond to the allocations per each queue.

## Default

Not defined

# LSF\_DUALSTACK\_PREFER\_IPV6

## Syntax

`LSF_DUALSTACK_PREFER_IPV6=Y | y`

## Description

Define this parameter when you want to ensure that clients and servers on dual-stack hosts use IPv6 addresses only. Setting this parameter configures LSF to sort the dynamically created address lookup list in order of AF\_INET6 (IPv6) elements first, followed by AF\_INET (IPv4) elements, and then others.

---

### Restriction:

IPv4-only and IPv6-only hosts cannot belong to the same cluster. In a MultiCluster environment, you cannot mix IPv4-only and IPv6-only clusters.

---

Follow these guidelines for using IPv6 addresses within your cluster:

- Define this parameter only if your cluster
    - Includes only dual-stack hosts, or a mix of dual-stack and IPv6-only hosts, *and*
    - Does not include IPv4-only hosts or IPv4 servers running on dual-stack hosts (servers prior to LSF version 7)
- 

### Important:

Do not define this parameter for any other cluster configuration.

- Within a MultiCluster environment, do not define this parameter if any cluster contains IPv4-only hosts or IPv4 servers (prior to LSF version 7) running on dual-stack hosts.
- Applications must be engineered to work with the cluster IP configuration.
- If you use IPv6 addresses within your cluster, ensure that you have configured the dual-stack hosts correctly. For more detailed information, see *Administering Platform LSF*.
- Define the parameter LSF\_ENABLE\_SUPPORT\_IPV6 in `lsf.conf`.

## Default

Not defined. LSF sorts the dynamically created address lookup list in order of AF\_INET (IPv4) elements first, followed by AF\_INET6 (IPv6) elements, and then others. Clients and servers on dual-stack hosts use the first address lookup structure in the list (IPv4).

## See also

LSF\_ENABLE\_SUPPORT\_IPV6

# LSF\_DYNAMIC\_HOST\_TIMEOUT

## Syntax

`LSF_DYNAMIC_HOST_TIMEOUT=time_hours`

`LSF_DYNAMIC_HOST_TIMEOUT=time_minutesm|M`

## Description

Enables automatic removal of dynamic hosts from the cluster and specifies the timeout value (minimum 10 minutes). To improve performance in very large clusters, you should disable this feature and remove unwanted hosts from the `hostcache` file manually.

Specifies the length of time a dynamic host is unavailable before the master host removes it from the cluster. Each time LSF removes a dynamic host, `mbat chd` automatically reconfigures itself.

## Valid value

The timeout value must be greater than or equal to 10 minutes.

Values below 10 minutes are set to the minimum allowed value 10 minutes; values above 100 hours are set to the maximum allowed value 100 hours.

## Example

```
LSF_DYNAMIC_HOST_TIMEOUT=60
```

A dynamic host is removed from the cluster when it is unavailable for 60 hours.

```
LSF_DYNAMIC_HOST_TIMEOUT=60m
```

A dynamic host is removed from the cluster when it is unavailable for 60 minutes.

## EGO parameter

EGO\_DYNAMIC\_HOST\_TIMEOUT

## Default

-1 (Not defined.) Unavailable hosts are never removed from the cluster.

# LSF\_DYNAMIC\_HOST\_WAIT\_TIME

## Syntax

```
LSF_DYNAMIC_HOST_WAIT_TIME=time_seconds
```

## Description

Defines the length of time in seconds that a dynamic host waits communicating with the master LIM to either add the host to the cluster or to shut down any running daemons if the host is not added successfully.

---

### Note:

To enable dynamically added hosts, the following parameters must be defined:

- LSF\_MASTER\_LIST in `lsf.conf`
  - LSF\_DYNAMIC\_HOST\_WAIT\_TIME in `lsf.conf`, or  
EGO\_DYNAMIC\_HOST\_WAIT\_TIME in `ego.conf`
  - LSF\_HOST\_ADDR\_RANGE in `lsf.cluster.cluster_name`
- 

### Note:

To enable daemons to be shut down automatically for hosts that attempted to join the cluster but were rejected within the LSF\_DYNAMIC\_HOST\_WAIT\_TIME period:

- EGO\_ENABLE\_AUTO\_DAEMON\_SHUTDOWN in `lsf.conf` or in `ego.conf`.

## Recommended value

An integer greater than zero, up to 60 seconds for every 1000 hosts in the cluster, for a maximum of 15 minutes. Selecting a smaller value results in a quicker response time for hosts at the expense of an increased load on the master LIM.

## Example

```
LSF_DYNAMIC_HOST_WAIT_TIME=60
```

A host waits 60 seconds from startup to send a request for the master LIM to add it to the cluster or to shut down any daemons if it is not added to the cluster.

## EGO parameter

EGO\_DYNAMIC\_HOST\_WAIT\_TIME

## Default

Not defined. Dynamic hosts cannot join the cluster.

# LSF\_EGO\_DAEMON\_CONTROL

## Syntax

```
LSF_EGO_DAEMON_CONTROL="Y" | "N"
```

## Description

Enables EGO Service Controller to control LSF `res` and `sbatchd` startup. Set the value to "Y" if you want EGO Service Controller to start `res` and `sbatchd`, and restart them if they fail.

To configure this parameter at installation, set `EGO_DAEMON_CONTROL` in `install.config` so that `res` and `sbatchd` start automatically as EGO services.

If `LSF_ENABLE_EGO="N"`, this parameter is ignored and EGO Service Controller is not started.

If you manually set `EGO_DAEMON_CONTROL=Y` after installation, you *must* configure LSF `res` and `sbatchd` startup to `AUTOMATIC` in the EGO configuration files `res.xml` and `sbatchd.xml` under `EGO_ESRVDIR/esc/conf/services`.

To avoid conflicts with existing LSF startup scripts, do not set this parameter to "Y" if you use a script (for example in `/etc/rc` or `/etc/inettab`) to start LSF daemons. If this parameter is not defined in `install.config` file, it takes default value of "N".

---

### Important:

After installation, `LSF_EGO_DAEMON_CONTROL` alone *does not* change the start type for the `sbatchd` and `res` EGO services to `AUTOMATIC` in `res.xml` and `sbatchd.xml` under `EGO_ESRVDIR/`

`esc/conf/services`. You must edit these files and set the `<sc: StartType>` parameter to `AUTOMATIC`.

---

## Example

```
LSF_EGO_DAEMON_CONTROL="N"
```

## Default

N (res and sbatchd are started manually or through operating system rc facility)

# LSF\_EGO\_ENVDIR

## Syntax

```
LSF_EGO_ENVDIR=directory
```

## Description

Directory where all Platform EGO configuration files are installed. These files are shared throughout the system and should be readable from any host.

If `LSF_ENABLE_EGO="N"`, this parameter is ignored and `ego.conf` is not loaded.

## Default

`LSF_CONFDIR/ego/cluster_name/kernel`. If not defined, or commented out, `/etc` is assumed.

# LSF\_ENABLE\_CSA

## Syntax

```
LSF_ENABLE_CSA=y | Y
```

## Description

If set, enables LSF to write records for LSF jobs to SGI IRIX Comprehensive System Accounting facility (CSA).

CSA writes an accounting record for each process in the `pacct` file, which is usually located in the `/var/adm/acct/day` directory. IRIX system administrators then use the `csabuid` command to organize and present the records on a job by job basis.

When `LSF_ENABLE_CSA` is set, for each job run on the IRIX system, LSF writes an LSF-specific accounting record to CSA when the job starts, and when the job finishes. LSF daemon accounting in CSA starts and stops with the LSF daemon.

To disable IRIX CSA accounting, remove `LSF_ENABLE_CSA` from `lsf.conf`.

See the IRIX resource administration documentation for information about CSA.

## Set up IRIX CSA

1. Define the `LSF_ENABLE_CSA` parameter in `lsf.conf`:
 

```
... LSF_ENABLE_CSA=Y ...
```
2. Set the following parameters in `/etc/csa.conf` to on:

- CSA\_START
  - WKMG\_START
3. Run the `csaswi t ch` command to turn on the configuration changes in `/etc/csa.conf`.

---

**Note:**

See the IRIX resource administration documentation for information about the `csaswi t ch` command.

---

## Information written to the `pacct` file

LSF writes the following records to the `pacct` file when a job starts and when it exits:

- Job record type (job start or job exit)
- Current system clock time
- Service provider (LSF)
- Submission time of the job (at job start only)
- User ID of the job owner
- Array Session Handle (ASH) of the job
- IRIX job ID
- IRIX project ID
- LSF job name if it exists
- Submission host name
- LSF queue name
- LSF external job ID
- LSF job array index
- LSF job exit code (at job exit only)
- NCPUS: number of CPUs the LSF job has been using

## Default

Not defined

# LSF\_ENABLE\_DUALCORE

## Syntax

`LSF_ENABLE_DUALCORE=y | n`

## Description

Enables job scheduling based on dual-core information for a host. If yes (Y), LSF scheduling policies use the detected number of cores as the number of physical processors on the host instead of the number of dual-core chips for job scheduling. For a dual-core host, `lshosts` shows the number of cores under `ncpus` instead of the number of chips.

IF `LSF_ENABLE_DUALCORE=n`, then `lshosts` shows the number of processor chips under `ncpus`.

## EGO parameter

`EGO_ENABLE_DUALCORE`

## Default

N

# LSF\_ENABLE\_EGO

## Syntax

```
LSF_ENABLE_EGO="Y" | "N"
```

## Description

Enables Platform EGO functionality in the LSF cluster.

If you set `LSF_ENABLE_EGO="Y"`, you must set or uncomment `LSF_EGO_ENVDIR` in `lsf.conf`.

If you set `LSF_ENABLE_EGO="N"` you must remove or comment out `LSF_EGO_ENVDIR` in `lsf.conf`.

Set the value to "N" if you do not want to take advantage of the following LSF features that depend on EGO:

- LSF daemon control by EGO Service Controller
- EGO-enabled SLA scheduling

---

### Important:

After changing the value of `LSF_ENABLE_EGO`, you must shut down and restart the cluster.

---

## Default

Y (EGO is enabled in the LSF cluster)

# LSF\_ENABLE\_EXTSCHEULER

## Syntax

```
LSF_ENABLE_EXTSCHEULER=y | Y
```

## Description

If set, enables `mbatchd` external scheduling for LSF HPC features.

## Default

Not defined

# LSF\_ENABLE\_SUPPORT\_IPV6

## Syntax

```
LSF_ENABLE_SUPPORT_IPV6=y | Y
```

## Description

If set, enables the use of IPv6 addresses in addition to IPv4.

## Default

Not defined

lsf.conf

## See also

LSF\_DUALSTACK\_PREFER\_IPV6

# LSF\_ENVDIR

## Syntax

**LSF\_ENVDIR**=*directory*

## Description

Directory containing the `lsf.conf` file.

By default, `lsf.conf` is installed by creating a shared copy in `LSF_CONFDIR` and adding a symbolic link from `/etc/lsf.conf` to the shared copy. If `LSF_ENVDIR` is set, the symbolic link is installed in `LSF_ENVDIR/lsf.conf`.

The `lsf.conf` file is a global environment configuration file for all LSF services and applications. The LSF default installation places the file in `LSF_CONFDIR`.

## Default

`/etc`

# LSF\_EVENT\_PROGRAM

## Syntax

**LSF\_EVENT\_PROGRAM**=*event\_program\_name*

## Description

Specifies the name of the LSF event program to use.

If a full path name is not provided, the default location of this program is `LSF_SERVERDIR`.

If a program that does not exist is specified, event generation does not work.

If this parameter is not defined, the default name is `genevent` on UNIX, and `genevent.exe` on Windows.

## Default

Not defined

# LSF\_EVENT\_RECEIVER

## Syntax

**LSF\_EVENT\_RECEIVER**=*event\_receiver\_program\_name*

## Description

Specifies the LSF event receiver and enables event generation.

Any string may be used as the LSF event receiver; this information is not used by LSF to enable the feature but is only passed as an argument to the event program.

If `LSF_EVENT_PROGRAM` specifies a program that does not exist, event generation does not work.

## Default

Not defined. Event generation is disabled

# LSF\_GET\_CONF

## Syntax

`LSF_GET_CONF=lim`

## Description

Synchronizes a local host's cluster configuration with the master host's cluster configuration. Specifies that a slave host must request cluster configuration details from the LIM of a host on the `SERVER_HOST` list. Use when a slave host does not share a filesystem with master hosts, and therefore cannot access cluster configuration.

## Default

Not defined.

# LSF\_HOST\_CACHE\_NTTL

## Syntax

`LSF_HOST_CACHE_NTTL=time_seconds`

## Description

Negative-time-to-live value in seconds. Specifies the length of time the system caches a failed DNS lookup result. If you set this value to zero (0), LSF does not cache the result.

---

**Note:**

Setting this parameter does not affect the positive-time-to-live value set by the parameter `LSF_HOST_CACHE_PTTL`.

---

## Valid values

Positive integer. Recommended value less than or equal to 60 seconds (1 minute).

## Default

20 seconds

## See also

`LSF_HOST_CACHE_PTTL`

# LSF\_HOST\_CACHE\_PTTL

## Syntax

`LSF_HOST_CACHE_PTTL=time_seconds`

## Description

Positive-time-to-live value in seconds. Specifies the length of time the system caches a successful DNS lookup result. If you set this value to zero (0), LSF does not cache the result.

---

### Note:

Setting this parameter does not affect the negative-time-to-live value set by the parameter LSF\_HOST\_CACHE\_NTTL.

---

## Valid values

Positive integer. Recommended value equal to or greater than 3600 seconds (1 hour).

## Default

86400 seconds (24 hours)

## See also

LSF\_HOST\_CACHE\_NTTL

# LSF\_HPC\_EXTENSIONS

## Syntax

**LSF\_HPC\_EXTENSIONS**="*extension\_name ...*"

## Description

Enables Platform LSF HPC extensions.

After adding or changing LSF\_HPC\_EXTENSIONS, use `badmi n mbdrestart` and `badmi n hrestart` to reconfigure your cluster.

## Valid values

The following extension names are supported:

**CUMULATIVE\_RUSAGE**: When a parallel job script runs multiple commands, resource usage is collected for jobs in the job script, rather than being overwritten when each command is executed.

**DISP\_RES\_USAGE\_LIMITS**: `bjobs` displays resource usage limits configured in the queue as well as job-level limits.

**HOST\_RUSAGE**: For parallel jobs, reports the correct rusage based on each host's usage and the total rusage being charged to the execution host. This host rusage breakdown applies to the blaunch framework, the pam framework, and vendor MPI jobs (HP and SGI). For a running job, you will see run time, memory, swap, utime, stime, and pids and pgids on all hosts that a parallel job spans. For finished jobs, you will see memory, swap, utime, and stime on all hosts that a parallel job spans. The host-based rusage is reported in the JOB\_FINISH record of `lsb. acct` and `lsb. stream`, and the JOB\_STATUS record of `lsb. events` if the job status is done or exit. Also for finished jobs, `bjobs -l` shows CPU time, `bhist -l` shows CPU time, and `bacct -l` shows utime, stime, memory, and swap. In the MultiCluster lease model, the parallel job must run on hosts that are all in the same cluster. If you use the jobFinishLog API, all external tools must use jobFinishLog built with LSF 8.0, or host-based rusage will not work. If you add or remove this extension, you must restart `mbatchd`, `sbatchd`, and `reson` on all hosts.

**LSB\_HCLOSE\_BY\_RES**: If `res` is down, host is closed with a message

```
Host is closed because RES is not available.
```

The status of the closed host is `closed_Adm`. No new jobs are dispatched to this host, but currently running jobs are not suspended.

**RESERVE\_BY\_STARTTIME**: LSF selects the reservation that gives the job the earliest predicted start time.

By default, if multiple host groups are available for reservation, LSF chooses the largest possible reservation based on number of slots.

**SHORT\_EVENTFILE**: Compresses long host name lists when event records are written to `lsb.events` and `lsb.acct` for large parallel jobs. The short host string has the format:

```
number_of_hosts*real_host_name
```

---

### Tip:

When **SHORT\_EVENTFILE** is enabled, older daemons and commands (pre-LSF Version 7) cannot recognize the `lsb.acct` and `lsb.events` file format.

---

For example, if the original host list record is

```
6 "hostA" "hostA" "hostA" "hostA" "hostB" "hostC"
```

redundant host names are removed and the short host list record becomes

```
3 "4*hostA" "hostB" "hostC"
```

When **LSF\_HPC\_EXTENSIONS="SHORT\_EVENTFILE"** is set, and LSF reads the host list from `lsb.events` or `lsb.acct`, the compressed host list is expanded into a normal host list.

**SHORT\_EVENTFILE** affects the following events and fields:

- **JOB\_START** in `lsb.events` when a normal job is dispatched
  - `numExHosts` (%d)
  - `execHosts` (%s)
- **JOB\_CHUNK** in `lsb.events` when a job is inserted into a job chunk
  - `numExHosts` (%d)
  - `execHosts` (%s)
- **JOB\_FORWARD** in `lsb.events` when a job is forwarded to a MultiCluster leased host
  - `numReserHosts` (%d)
  - `reserHosts` (%s)
- **JOB\_FINISH** record in `lsb.acct`
  - `numExHosts` (%d)
  - `execHosts` (%s)

**SHORT\_PIDLIST**: Shortens the output from `bj obs` to omit all but the first process ID (PID) for a job. `bj obs` displays only the first ID and a count of the process group IDs (PGIDs) and process IDs for the job.

Without **SHORT\_PIDLIST**, `bj obs -l` displays all the PGIDs and PIDs for the job. With **SHORT\_PIDLIST** set, `bj obs -l` displays a count of the PGIDs and PIDs.

**TASK\_MEMLIMIT**: Enables enforcement of a memory limit (`bsub -M`, `bmod -M`, or `MEMLIMIT` in `l sb. queues`) for individual tasks in a parallel job. If any parallel task exceeds the memory limit, LSF terminates the entire job.

**TASK\_SWAPLIMIT**: Enables enforcement of a virtual memory (swap) limit (`bsub -v`, `bmod -v`, or `SWAPLIMIT` in `l sb. queues`) for individual tasks in a parallel job. If any parallel task exceeds the swap limit, LSF terminates the entire job.

## Example JOB\_START events in lsb.events:

For a job submitted with

```
bsub -n 64 -R "span[ptile=32]" sleep 100
```

*Without* `SHORT_EVENTFILE`, a `JOB_START` event like the following is logged in `l sb. events`:

```
"JOB_START" "8.0" 1058989891 710 4 0 0 10.3 64 "hostA" "hostA" "hostA" "hostA" "hostA"
"hostA" "hostA" "hostA" "hostA" "hostA" "hostA" "hostA" "hostA" "hostA" "hostA" "hostA"
"hostA" "hostA" "hostA" "hostA" "hostA" "u050" "hostA" "hostA" "hostA" "hostA" "hostA"
"hostA" "hostA" "hostA" "hostA" "hostA" "hostB" "hostB" "hostB" "hostB" "hostB" "hostB"
"hostB" "hostB" "hostB" "hostB" "hostB" "hostB" "hostB" "hostB" "hostB" "hostB" "hostB"
"hostB" "hostB" "hostB" "hostB" "hostB" "hostB" "hostB" "hostB" "hostB" "hostB" "hostB"
"hostB" "hostB" "hostB" "hostB" " " " " 0 " " 0
```

*With* `SHORT_EVENTFILE`, a `JOB_START` event would be logged in `l sb. events` with the number of execution hosts (`numExHosts` field) changed from 64 to 2 and the execution host list (`execHosts` field) shortened to `"32*hostA"` and `"32*hostB"`:

```
"JOB_START" "8.0" 1058998174 812 4 0 0 10.3 2 "32*hostA" "32*hostB" " " " " 0 " " 0 "
```

## Example JOB\_FINISH records in lsb.acct:

For a job submitted with

```
bsub -n 64 -R "span[ptile=32]" sleep 100
```

*Without* `SHORT_EVENTFILE`, a `JOB_FINISH` event like the following is logged in `l sb. acct`:

```
"JOB_FINISH" "8.0" 1058990001 710 33054 33816578 64 1058989880 0 0 1058989891 "user1"
"normal" "span[ptile=32]" " " " " "hostA" "/scratch/user1/work" " " " " "1058989880.710"
0 64 "hostA" "hostA"
"hostA" "hostA" "hostA" "hostA" "hostA" "hostA" "hostA" "hostA" "hostA" "hostA" "hostA"
"hostA" "hostA" "hostA" "hostA" "hostA" "hostA" "hostA" "hostA" "hostA" "hostA" "hostA"
"hostB" "hostB"
"hostB" "hostB" "hostB" "hostB" "hostB" "hostB" "hostB" "hostB" "hostB" "hostB" "hostB"
"hostB" "hostB" "hostB" "hostB" "hostB" "hostB" "hostB" "hostB" "hostB" "hostB" "hostB" 64 10.3
" " "sleep 100" 0.079999 0.270000 0 0 -1 0 0 0 0 0 0 -1 0 0 0 0 0 -1 " " "default" 0 64
" " " " 0 4304 6024 " " " " 0
```

With `SHORT_EVENTFILE`, a `JOB_FINISH` event like the following would be logged in `lsb.acct` with the number of execution hosts (`numExHosts` field) changed from 64 to 2 and the execution host list (`execHosts` field) shortened to `"32*hostA"` and `"32*hostB"`:

```
"JOB_FINISH" "8.0" 1058998282 812 33054 33816578 64 1058998163 0 0 1058998174 "user1"
"normal" "span[ptile=32]" "" "" "hostA" "/scratch/user1/work" "" "" "" "1058998163.812"
0 2 "32*hostA" "32*hostB" 64 10.3 "" "sleep 100" 0.039999 0.259999 0 0 -1 0 0 0 0 0 0
-1 0 0 0 0 0 -1 "" "default" 0 64 "" "" 0 4304 6024 "" "" "" "" 0 0
```

## Example `bjobs -l` output without `SHORT_PIDLIST`:

`bjobs -l` displays all the PGIDs and PIDs for the job:

### `bjobs -l`

```
Job <109>, User <user3>, Project <default>, Status <RUN>, Queue <normal>, Inte
ractive mode, Command <./myjob.sh>
Mon Jul 21 20:54:44 2009: Submitted from host <hostA>, CWD <$HOME/LSF/jobs>
```

#### RUNLIMIT

```
10.0 min of hostA
```

#### STACKLIMIT CORELIMIT MEMLIMIT

```
5256 K 10000 K 5000 K
```

```
Mon Jul 21 20:54:51 2009: Started on <hostA>;
```

```
Mon Jul 21 20:55:03 2009: Resource usage collected.
```

```
MEM: 2 Mbytes; SWAP: 15 Mbytes
```

```
PGID: 256871; PIDs: 256871
```

```
PGID: 257325; PIDs: 257325 257500 257482 257501 257523
```

```
257525 257531
```

#### SCHEDULING PARAMETERS:

	r15s	r1m	r15m	ut	pg	io	ls	it	tmp	swp	mem
loadSched	-	-	-	-	-	-	-	-	-	-	-
loadStop	-	-	-	-	-	-	-	-	-	-	-
	cpuspeed		bandwidth								
loadSched	-	-									
loadStop	-	-									

```
<< Job <109> is done successfully. >>
```

## Example bjobs -l output with SHORT\_PIDLIST:

bjobs -l displays a count of the PGIDS and PIDs:

### bjobs -l

```
Job <109>, User <user3>, Project <default>, Status <RUN>, Queue <normal>, Inte
ractive mode, Command <./myjob.sh>
```

```
Mon Jul 21 20:54:44 2009: Submitted from host <hostA>, CWD <$HOME/LSF/jobs>
```

#### RUNLIMIT

```
10.0 min of hostA
```

#### STACKLIMIT CORELIMIT MEMLIMIT

```
5256 K 10000 K 5000 K
```

```
Mon Jul 21 20:54:51 2009: Started on <hostA>;
```

```
Mon Jul 21 20:55:03 2009: Resource usage collected.
```

```
MEM: 2 Mbytes; SWAP: 15 Mbytes
```

```
PGID(s): 256871: 1 PID, 257325: 7 PIDs
```

#### SCHEDULING PARAMETERS:

	r15s	r1m	r15m	ut	pg	io	ls	it	tmp	swp	mem
loadSched	-	-	-	-	-	-	-	-	-	-	-
loadStop	-	-	-	-	-	-	-	-	-	-	-
	cpuspeed		bandwidth								
loadSched	-	-	-	-							
loadStop	-	-	-	-							

## Default

Not defined

## LSF\_HPC\_PJL\_LOADENV\_TIMEOUT

### Syntax

```
LSF_HPC_PJL_LOADENV_TIMEOUT=time_seconds
```

### Description

Timeout value in seconds for PJL to load or unload the environment. For example, set LSF\_HPC\_PJL\_LOADENV\_TIMEOUT to the number of seconds needed for IBM POE to load or unload adapter windows.

At job startup, the PJL times out if the first task fails to register with PAM within the specified timeout value. At job shutdown, the PJL times out if it fails to exit after the last Taskstarter termination report within the specified timeout value.

### Default

```
LSF_HPC_PJL_LOADENV_TIMEOUT=300
```

## LSF\_ID\_PORT

### Syntax

**LSF\_ID\_PORT**=*port\_number*

### Description

The network port number used to communicate with the authentication daemon when LSF\_AUTH is set to ident.

### Default

Not defined

## LSF\_INCLUDEDIR

### Syntax

**LSF\_INCLUDEDIR**=*directory*

### Description

Directory under which the LSF API header files `lsf.h` and `lsbatch.h` are installed.

### Default

LSF\_INDEP/include

### See also

LSF\_INDEP

## LSF\_INDEP

### Syntax

**LSF\_INDEP**=*directory*

### Description

Specifies the default top-level directory for all machine-independent LSF files.

This includes man pages, configuration files, working directories, and examples. For example, defining LSF\_INDEP as `/usr/share/lsf/mnt` places man pages in `/usr/share/lsf/mnt/man`, configuration files in `/usr/share/lsf/mnt/conf`, and so on.

The files in LSF\_INDEP can be shared by all machines in the cluster.

As shown in the following list, LSF\_INDEP is incorporated into other LSF environment variables.

- LSB\_SHAREDIR=\$LSF\_INDEP/work
- LSF\_CONFDIR=\$LSF\_INDEP/conf
- LSF\_INCLUDEDIR=\$LSF\_INDEP/include
- LSF\_MANDIR=\$LSF\_INDEP/man
- XLSF\_APPDIR=\$LSF\_INDEP/misc

## Default

`/usr/share/lsf/mnt`

## See also

LSF\_MACHDEP, LSB\_SHAREDDIR, LSF\_CONFDIR, LSF\_INCLUDEDIR, LSF\_MANDIR, XLSF\_APPDIR

# LSF\_INTERACTIVE\_STDERR

## Syntax

`LSF_INTERACTIVE_STDERR=y | n`

## Description

Separates `stderr` from `stdout` for interactive tasks and interactive batch jobs.

This is useful to redirect output to a file with regular operators instead of the `bsub -e err_file` and `-o out_file` options.

This parameter can also be enabled or disabled as an environment variable.

---

### Caution:

If you enable this parameter globally in `lsf.conf`, check any custom scripts that manipulate `stderr` and `stdout`.

---

When this parameter is not defined or set to `n`, the following are written to `stdout` on the submission host for interactive tasks and interactive batch jobs:

- Job standard output messages
- Job standard error messages

The following are written to `stderr` on the submission host for interactive tasks and interactive batch jobs:

- LSF messages
- NIOS standard messages
- NIOS debug messages (if `LSF_NIOS_DEBUG=1` in `lsf.conf`)

When this parameter is set to `y`, the following are written to `stdout` on the submission host for interactive tasks and interactive batch jobs:

- Job standard output messages

The following are written to `stderr` on the submission host:

- Job standard error messages
- LSF messages
- NIOS standard messages
- NIOS debug messages (if `LSF_NIOS_DEBUG=1` in `lsf.conf`)

## Default

Not defined

## Notes

When this parameter is set, the change affects interactive tasks and interactive batch jobs run with the following commands:

- `bsub -I`
- `bsub -I p`
- `bsub -I s`
- `l srun`
- `l sgrun`
- `l smake` (Platform Make)
- `bsub pam` (HPC features must be enabled)

## Limitations

- **Pseudo-terminal:** Do not use this parameter if your application depends on `stderr` as a terminal. This is because LSF must use a non-pseudo-terminal connection to separate `stderr` from `stdout`.
- **Synchronization:** Do not use this parameter if you depend on messages in `stderr` and `stdout` to be synchronized and jobs in your environment are continuously submitted. A continuous stream of messages causes `stderr` and `stdout` to not be synchronized. This can be emphasized with parallel jobs. This situation is similar to that of `rsh`.
- **NIOS standard and debug messages:** NIOS standard messages, and debug messages (when `LSF_NIOS_DEBUG=1` in `lsf.conf` or as an environment variable) are written to `stderr`. NIOS standard messages are in the format `<<message>>`, which makes it easier to remove them if you wish. To redirect NIOS debug messages to a file, define `LSF_CMD_LOGDIR` in `lsf.conf` or as an environment variable.

## See also

`LSF_NIOS_DEBUG`, `LSF_CMD_LOGDIR`

## LSF\_LD\_SECURITY

### Syntax

`LSF_LD_SECURITY=y | n`

### Description

**LSF\_LD\_SECURITY:** When set, jobs submitted using `bsub -I s` or `bsub -I p` cause the environment variables `LD_PRELOAD` and `LD_LIBRARY_PATH` to be removed from the job environment during job initialization to ensure enhanced security against users obtaining root privileges.

Two new environment variables are created (`LSF_LD_LIBRARY_PATH` and `LSF_LD_PRELOAD`) to allow `LD_PRELOAD` and `LD_LIBRARY_PATH` to be put back before the job runs.

### Default

N

## LSF\_LIBDIR

### Syntax

`LSF_LIBDIR=directory`

## Description

Specifies the directory in which the LSF libraries are installed. Library files are shared by all hosts of the same type.

## Default

LSF\_MACHDEP/lib

## LSF\_LIC\_SCHED\_HOSTS

### Syntax

**LSF\_LIC\_SCHED\_HOSTS**="*candidate\_host\_list*"

*candidate\_host\_list* is a space-separated list of hosts that are candidate License Scheduler hosts.

## Description

The candidate License Scheduler host list is read by LIM on each host to check if the host is a candidate License Scheduler master host. If the host is on the list, LIM starts the License Scheduler daemon (lsched) on the host.

## LSF\_LIC\_SCHED\_PREEMPT\_REQUEUE

### Syntax

**LSF\_LIC\_SCHED\_PREEMPT\_REQUEUE**=y | n

## Description

Set this parameter to requeue a job whose license is preempted by Platform License Scheduler. The job is killed and requeued instead of suspended.

If you set LSF\_LIC\_SCHED\_PREEMPT\_REQUEUE, do not set LSF\_LIC\_SCHED\_PREEMPT\_SLOT\_RELEASE. If both these parameters are set, LSF\_LIC\_SCHED\_PREEMPT\_SLOT\_RELEASE is ignored.

## Default

N

## See also

LSF\_LIC\_SCHED\_PREEMPT\_SLOT\_RELEASE, LSF\_LIC\_SCHED\_PREEMPT\_STOP

## LSF\_LIC\_SCHED\_PREEMPT\_SLOT\_RELEASE

### Syntax

**LSF\_LIC\_SCHED\_PREEMPT\_SLOT\_RELEASE**=y | n

## Description

Set this parameter to release the slot of a job that is suspended when its license is preempted by Platform License Scheduler.

If you set `LSF_LIC_SCHED_PREEMPT_SLOT_RELEASE`, do not set `LSF_LIC_SCHED_PREEMPT_REQUEUE`. If both these parameters are set, `LSF_LIC_SCHED_PREEMPT_SLOT_RELEASE` is ignored.

## Default

Y

## See also

`LSF_LIC_SCHED_PREEMPT_REQUEUE`, `LSF_LIC_SCHED_PREEMPT_STOP`

# LSF\_LIC\_SCHED\_PREEMPT\_STOP

## Syntax

`LSF_LIC_SCHED_PREEMPT_STOP=y | n`

## Description

Set this parameter to use job controls to stop a job that is preempted. When this parameter is set, a UNIX SIGSTOP signal is sent to suspend a job instead of a UNIX SIGTSTP.

To send a SIGSTOP signal instead of SIGTSTP, the following parameter in `l sb. queues` must also be set:

```
JOB_CONTROLS=SUSPEND[ SIGSTOP]
```

## Default

N

## See also

`LSF_LIC_SCHED_PREEMPT_SLOT_RELEASE`, `LSF_LIC_SCHED_PREEMPT_REQUEUE`

# LSF\_LIC\_SCHED\_STRICT\_PROJECT\_NAME

## Syntax

`LSF_LIC_SCHED_STRICT_PROJECT_NAME=y | n`

## Description

Enforces strict checking of the License Scheduler project name upon job submission or job modification (`bsub` or `bmod`). If the project named is misspelled (case sensitivity applies), the job is rejected.

If this parameter is not set or it is set to `n`, and if there is an error in the project name, the default project is used.

## Default

N

# LSF\_LICENSE\_ACCT\_PATH

## Syntax

**LSF\_LICENSE\_ACCT\_PATH**=*directory*

## Description

Specifies the location for the license accounting files. These include the license accounting files for LSF Family products.

Use this parameter to define the location of all the license accounting files. By defining this parameter, you can store the license accounting files for the LSF Family of products in the same directory for convenience.

## Default

Not defined. The license accounting files are stored in the default log directory for the particular product. For example, LSF stores its license audit file in the LSF system log file directory.

## See also

- LSF\_LOGDIR
- lsf.cluster\_name.license.acct
- bld.license.acct

# LSF\_LICENSE\_FILE

## Syntax

**LSF\_LICENSE\_FILE**="*file\_name ... | port\_number@host\_name[:port\_number@host\_name ...]*"

## Description

Specifies one or more demo or FlexNet permanent license files used by LSF.

The value for LSF\_LICENSE\_FILE can be either of the following:

- The full path name to the license file.
  - UNIX example:  

```
LSF_LICENSE_FILE=/usr/share/lsf/cluster1/conf/license.dat
```
  - Windows examples:  

```
LSF_LICENSE_FILE= C:\licenses\license.dat
```

```
LSF_LICENSE_FILE=\\Host\licenses\license.dat
```
- For a permanent license, the name of the license server host and TCP port number used by the lmgrd daemon, in the format *port@host\_name*. For example:  

```
LSF_LICENSE_FILE="1700@hostD"
```
- For a license with redundant servers, use a semi-colon or colon to separate the *port@host\_names*. The port number must be the same as that specified in the SERVER line of the license file. For example:  
 UNIX:  

```
LSF_LICENSE_FILE="port@hostA:port@hostB:port@hostC"
```

Windows:

```
LSF_LICENSE_FILE="port@hostA;port@hostB;port@hostC"
```

- For a license with distributed servers, use a pipe to separate the *port@host\_names*. The port number must be the same as that specified in the SERVER line of the license file. For example:

```
LSF_LICENSE_FILE="port@hostA|port@hostB|port@hostC"
```

Multiple license files should be quoted and must be separated by a pipe character (|).

Windows example:

```
LSF_LICENSE_FILE="C:\licenses\lic1|C:\licenses\lic2|D:\mydir\lic3"
```

Multiple files may be kept in the same directory, but each one must reference a different license server. When checking out a license, LSF searches the servers in the order in which they are listed, so it checks the second server when there are no more licenses available from the first server.

If this parameter is not defined, LSF assumes the default location.

## Default

The default licence installation directory is the value of the parameter LSF\_CONFDIR or LSF\_ENVDIR in `lsf.conf`.

Demo license: The default demo licence installation directory is `/usr/local/fluxm/`.

# LSF\_LICENSE\_MAINTENANCE\_INTERVAL

## Syntax

```
LSF_LICENSE_MAINTENANCE_INTERVAL=time_seconds
```

## Description

Specifies how often LSF checks the LSF licences when starting or restarting the cluster. A small number could delay LSF. Valid values are from 5 to 300.

When this parameter is not set, the default value is used.

## Recommended value

Set LSF\_LICENSE\_MAINTENANCE\_INTERVAL depending on your cluster size, system buffer size, license server, and cluster communication speed:

- If you have network delays or a small system buffer (less than 32 KB), set LSF\_LICENSE\_MAINTENANCE to the high end of the valid values (300).
- For a small cluster (fewer than 1000 hosts), specify LSF\_LICENSE\_MAINTENANCE\_INTERVAL with 5-60 second value.
- For a large cluster (greater than 4000 hosts) with limited licenses, use the maximum value: 300 seconds.
- If you have slow cluster communication (for example, if you use a Web-based intranet), use the maximum value: 300 seconds.

## Default

5 seconds

# LSF\_LICENSE\_NOTIFICATION\_INTERVAL

## Syntax

**LSF\_LICENSE\_MAINTENANCE\_INTERVAL**=*time\_hours*

## Description

Specifies how often notification email is sent to the primary cluster administrator about overuse of LSF Family product licenses and Platform License Scheduler tokens.

## Recommended value

To avoid getting the same audit information more than once, set **LSF\_LICENSE\_NOTIFICATION\_INTERVAL** greater than 24 hours.

## Example notification email

```
Subject: LSF license overuse LSF Administrator: Your cluster has
experienced license overuse. Platform Product License Name:
LSF_MANAGER CLASS E license usage: 0 in total; 8 in use (8 overused).
Overuse Hosts: hostA Use lim -t and lshosts -l or see
/usr/opt/lsf7.0/log/lsf.cluster_8.0.license.acct file for details.
Please contact Platform Support at support@platform.com for
information about getting additional licenses.
```

## Default

24 hours

## See also

- **LSF\_LICENSE\_ACCT\_PATH**
- **LSF\_LOGDIR**
- `lsf.cluster_name.license.acct`
- `bld.license.acct`

# LSF\_LIM\_API\_NTRIES

## Syntax

**LSF\_LIM\_API\_NTRIES**=*integer*

## Description

Defines the number of times LSF commands will try to communicate with the LIM API when LIM is not available. **LSF\_LIM\_API\_NTRIES** is ignored by LSF and EGO daemons and EGO commands. The **LSF\_LIM\_API\_NTRIES** environment variable overrides the value of **LSF\_LIM\_API\_NTRIES** in `lsf.conf`.

## Valid values

1 to 65535

## Default

1. LIM API exits without retrying.

# LSF\_LIM\_DEBUG

## Syntax

**LSF\_LIM\_DEBUG=1 | 2**

## Description

Sets LSF to debug mode.

If `LSF_LIM_DEBUG` is defined, LIM operates in single user mode. No security checking is performed, so LIM should not run as root.

LIM does not look in the services database for the LIM service port number. Instead, it uses port number 36000 unless `LSF_LIM_PORT` has been defined.

Specify 1 for this parameter unless you are testing LSF.

## Valid values

`LSF_LIM_DEBUG=1`

LIM runs in the background with no associated control terminal.

`LSF_LIM_DEBUG=2`

LIM runs in the foreground and prints error messages to `tty`.

## EGO parameter

`EGO_LIM_DEBUG`

## Default

Not defined

## See also

`LSF_RES_DEBUG`, `LSF_CMD_LOGDIR`, `LSF_CMD_LOG_MASK`, `LSF_LOG_MASK`, `LSF_LOGDIR`

# LSF\_LIM\_IGNORE\_CHECKSUM

## Syntax

**LSF\_LIM\_IGNORE\_CHECKSUM=y | Y**

## Description

Configure `LSF_LIM_IGNORE_CHECKSUM=Y` to ignore warning messages logged to `lim` log files on non-master hosts.

When `LSF_MASTER_LIST` is set, `lsadm in reconfi g` only restarts master candidate hosts (for example, after adding or removing hosts from the cluster). This can cause superfluous warning messages like the

following to be logged in the `limlog` files for non-master hosts because `limon` on these hosts are not restarted after configuration change:

```
Aug 26 13:47:35 2006 9746 4 8.0 xdr_loadvector: Sender <10.225.36.46:9999> has a
different configuration
```

## Default

Not defined.

## See also

`LSF_MASTER_LIST`

## LSF\_LIM\_PORT, LSF\_RES\_PORT, LSB\_MBD\_PORT, LSB\_SBD\_PORT

## Syntax

`LSF_LIM_PORT=port_number`

## Description

TCP service ports to use for communication with the LSF daemons.

If port parameters are not defined, LSF obtains the port numbers by looking up the LSF service names in the `/etc/services` file or the NIS (UNIX). If it is not possible to modify the services database, you can define these port parameters to set the port numbers.

## EGO parameter

`EGO_LIM_PORT`

## Default

On UNIX, the default is to get port numbers from the services database.

On Windows, these parameters are mandatory.

Default port number values are:

- `LSF_LIM_PORT=7869`
- `LSF_RES_PORT=6878`
- `LSB_MBD_PORT=6881`
- `LSB_SBD_PORT=6882`

## LSF\_LIVE\_CONFDIR

## Syntax

`LSF_LIVE_CONFDIR=directory`

## Description

Enables and disables live reconfiguration (`bconf` command) and sets the directory where configuration files changed by live reconfiguration are saved. `bconf` requests will be rejected if the directory does not exist and cannot be created, or is specified using a relative path.

When `LSF_LIVE_CONFDIR` is defined and contains configuration files, all LSF restart and reconfiguration reads these configuration files instead of the files in `LSF_CONFDIR`.

After adding or changing `LSF_LIVE_CONFDIR` in `lsf.conf`, use `badmi n mbdrestart` and `lsadmi n reconfi g` to reconfigure your cluster.

---

### Important:

Remove `LSF_LIVE_CONFDIR` configuration files or merge files into `LSF_CONFDIR` before upgrading LSF or applying patches to LSF.

---

See `bconf` in the *LSF Command Reference* or `bconf man` page for `bconf` (live reconfiguration) details.

## Default

Undefined. (`bconf` disabled.)

During installation, `LSF_LIVE_CONFDIR` is set to `LSB_SHAREDIR/cluster_name/live_confdir` where `cluster_name` is the name of the LSF cluster, as returned by `lsid`.

## See also

`LSF_CONFDIR`, `LSB_CONFDIR`

# LSF\_LOAD\_USER\_PROFILE

## Syntax

`LSF_LOAD_USER_PROFILE=local | roaming`

## Description

When running jobs on Windows hosts, you can specify whether a user profile should be loaded. Use this parameter if you have jobs that need to access user-specific resources associated with a user profile.

Local and roaming user profiles are Windows features. For more information about them, check Microsoft documentation.

- Local: LSF loads the Windows user profile from the local execution machine (the host on which the job runs).

---

### Note:

If the user has logged onto the machine before, the profile of that user is used. If not, the profile for the default user is used

---

- Roaming: LSF loads a roaming user profile if it has been set up. If not, the local user profile is loaded instead.

## Default

Not defined. No user profiles are loaded when jobs run on Windows hosts.

# LSF\_LOCAL\_RESOURCES

## Syntax

`LSF_LOCAL_RESOURCES="resource ..."`

## Description

Defines instances of local resources residing on the slave host.

- For numeric resources, defined name-value pairs:  
" [resourcemap *value*\**resource\_name*] "
- For Boolean resources, the value is the resource name in the form:  
" [resource *resource\_name*] "

When the slave host calls the master host to add itself, it also reports its local resources. The local resources to be added must be defined in `lsf.shared`.

If the same resource is already defined in `lsf.shared` as default or all, it cannot be added as a local resource. The shared resource overrides the local one.

---

### Tip:

LSF\_LOCAL\_RESOURCES is usually set in the `slave.config` file during installation. If LSF\_LOCAL\_RESOURCES are already defined in a local `lsf.conf` on the slave host, `lsfinstall` does not add resources you define in LSF\_LOCAL\_RESOURCES in `slave.config`. You should not have duplicate LSF\_LOCAL\_RESOURCES entries in `lsf.conf`. If local resources are defined more than once, only the last definition is valid.

---

### Important:

Resources must already be mapped to hosts in the ResourceMap section of `lsf.cluster.cluster_name`. If the ResourceMap section does not exist, local resources are not added.

---

## Example

```
LSF_LOCAL_RESOURCES=" [resourcemap 1*verilog] [resource linux] "
```

## EGO parameter

EGO\_LOCAL\_RESOURCES

## Default

Not defined

## LSF\_LOG\_MASK

## Syntax

**LSF\_LOG\_MASK**=*message\_log\_level*

## Description

Specifies the logging level of error messages for LSF daemons, except LIM, which is controlled by Platform EGO.

For example:

```
LSF_LOG_MASK=LOG_DEBUG
```

If EGO is enabled in the LSF cluster, and EGO\_LOG\_MASK is not defined, LSF uses the value of LSF\_LOG\_MASK for LIM, PIM, and MELIM. EGO `vemkd` and `pem` components continue to use the EGO default values. If EGO\_LOG\_MASK is defined, and EGO is enabled, then EGO value is taken.

To specify the logging level of error messages for LSF commands, use LSF\_CMD\_LOG\_MASK. To specify the logging level of error messages for LSF batch commands, use LSB\_CMD\_LOG\_MASK.

On UNIX, this is similar to `syslog`. All messages logged at the specified level or higher are recorded; lower level messages are discarded. The LSF\_LOG\_MASK value can be any log priority symbol that is defined in `syslog.h` (see `syslog`).

The log levels in order from highest to lowest are:

- LOG\_EMERG
- LOG\_ALERT
- LOG\_CRIT
- LOG\_ERR
- LOG\_WARNING
- LOG\_NOTICE
- LOG\_INFO
- LOG\_DEBUG
- LOG\_DEBUG1
- LOG\_DEBUG2
- LOG\_DEBUG3

The most important LSF log messages are at the LOG\_ERR or LOG\_WARNING level. Messages at the LOG\_INFO and LOG\_DEBUG level are only useful for debugging.

Although message log level implements similar functionality to UNIX `syslog`, there is no dependency on UNIX `syslog`. It works even if messages are being logged to files instead of `syslog`.

LSF logs error messages in different levels so that you can choose to log all messages, or only log messages that are deemed critical. The level specified by LSF\_LOG\_MASK determines which messages are recorded and which are discarded. All messages logged at the specified level or higher are recorded, while lower level messages are discarded.

For debugging purposes, the level LOG\_DEBUG contains the fewest number of debugging messages and is used for basic debugging. The level LOG\_DEBUG3 records all debugging messages, and can cause log files to grow very large; it is not often used. Most debugging is done at the level LOG\_DEBUG2.

In versions earlier than LSF 4.0, you needed to restart the daemons after setting LSF\_LOG\_MASK in order for your changes to take effect.

LSF 4.0 implements dynamic debugging, which means you do not need to restart the daemons after setting a debugging environment variable.

## EGO parameter

EGO\_LOG\_MASK

## Default

LOG\_WARNING

## See also

LSB\_CMD\_LOG\_MASK, LSB\_CMD\_LOGDIR, LSB\_DEBUG, LSB\_DEBUG\_CMD, LSB\_DEBUG\_NQS, LSB\_TIME\_CMD, LSF\_CMD\_LOGDIR, LSF\_CMD\_LOG\_MASK,

LSF\_DEBUG\_LIM, LSB\_DEBUG\_MBD, LSF\_DEBUG\_RES, LSB\_DEBUG\_SBD, LSB\_DEBUG\_SCH,  
LSF\_LOG\_MASK, LSF\_LOGDIR, LSF\_TIME\_CMD

## LSF\_LOG\_MASK\_WIN

### Syntax

**LSF\_LOG\_MASK\_WIN**=*message\_log\_level*

### Description

Allows you to reduce the information logged to the LSF Windows event log files. Messages of lower severity than the specified level are discarded.

For all LSF files, the types of messages saved depends on LSF\_LOG\_MASK, so the threshold for the Windows event logs is either LSF\_LOG\_MASK or LSF\_LOG\_MASK\_WIN, whichever is higher. LSF\_LOG\_MASK\_WIN is ignored if LSF\_LOG\_MASK is set to a higher level.

The LSF event log files for Windows are:

- `lim.log`. *host\_name*
- `res.log`. *host\_name*
- `sbatchd.log`. *host\_name*
- `mbatchd.log`. *host\_name*
- `piim.log`. *host\_name*

The log levels you can specify for this parameter, in order from highest to lowest, are:

- LOG\_ERR
- LOG\_WARNING
- LOG\_INFO
- LOG\_NONE (LSF does not log Windows events)

### Default

LOG\_ERR

### See also

LSF\_LOG\_MASK

## LSF\_LOGDIR

### Syntax

**LSF\_LOGDIR**=*directory*

### Description

Defines the LSF system log file directory. Error messages from all servers are logged into files in this directory. To effectively use debugging, set LSF\_LOGDIR to a directory such as `/tmp`. This can be done in your own environment from the shell or in `lsf.conf`.

### Windows

LSF\_LOGDIR is required on Windows if you wish to enable logging.

You must also define `LSF_LOGDIR_USE_WIN_REG=n`.

If you define `LSF_LOGDIR` without defining `LSF_LOGDIR_USE_WIN_REG=n`, LSF logs error messages into files in the default local directory specified in one of the following Windows registry keys:

- On Windows 2000, Windows XP, and Windows 2003:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Platform Computing Corporation\LSF\cluster_name
\LSF_LOGDIR
```

- On Windows XP x64 and Windows 2003 x64:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Platform Computing Corporation\LSF
\cluster_name\LSF_LOGDIR
```

If a server is unable to write in the LSF system log file directory, LSF attempts to write to the following directories in the following order:

- `LSF_TMPDIR` if defined
- `%TMP%` if defined
- `%TEMP%` if defined
- System directory, for example, `c: \winnt`

## UNIX

If a server is unable to write in this directory, the error logs are created in `/tmp` on UNIX.

If `LSF_LOGDIR` is not defined, `syslog` is used to log everything to the system log using the `LOG_DAEMON` facility. The `syslog` facility is available by default on most UNIX systems. The `/etc/syslog.conf` file controls the way messages are logged and the files they are logged to. See the man pages for the `syslog` daemon and the `syslog` function for more information.

## Default

Not defined. On UNIX, log messages go to `syslog`. On Windows, no logging is performed.

## See also

`LSB_CMD_LOG_MASK`, `LSB_CMD_LOGDIR`, `LSB_DEBUG`, `LSB_DEBUG_CMD`, `LSB_TIME_CMD`, `LSF_CMD_LOGDIR`, `LSF_CMD_LOG_MASK`, `LSF_LOG_MASK`, `LSF_LOGDIR_USE_WIN_REG`, `LSF_TIME_CMD`

## Files

- `lim.log`, *host\_name*
- `res.log`, *host\_name*
- `sbatchd.log`, *host\_name*
- `sbatchdc.log`, *host\_name* (when `LSF_DAEMON_WRAP=Y`)
- `mbatchd.log`, *host\_name*
- `eventd.log`, *host\_name*
- `piim.log`, *host\_name*

# LSF\_LOGDIR\_USE\_WIN\_REG

## Syntax

```
LSF_LOGDIR_USE_WIN_REG=n | N
```

## Description

Windows only.

If set, LSF logs error messages into files in the directory specified by `LSF_LOGDIR` in `lsf.conf`.

Use this parameter to enable LSF to save log files in a different location from the default local directory specified in the Windows registry.

If not set, or if set to any value other than `N` or `n`, LSF logs error messages into files in the default local directory specified in one of the following Windows registry keys:

- On Windows 2000, Windows XP, and Windows 2003:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Platform Computing Corporation\LSF\cluster_name\LSF_LOGDIR
```

- On Windows XP x64 and Windows 2003 x64:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Platform Computing Corporation\LSF\cluster_name\LSF_LOGDIR
```

## Default

Not set.

LSF uses the default local directory specified in the Windows registry.

## See also

`LSF_LOGDIR`

# LSF\_LOGFILE\_OWNER

## Syntax

```
LSF_LOGFILE_OWNER="user_name"
```

## Description

Specifies an owner for the LSF log files other than the default, the owner of `lsf.conf`. To specify a Windows user account, include the domain name in uppercase letters (`DOMAIN_NAME\user_name`).

## Default

Not set. The LSF Administrator with root privileges is the owner of LSF log files.

# LSF\_LSLOGIN\_SSH

## Syntax

```
LSF_LSLOGIN_SSH=YES | yes
```

## Description

Enables SSH to secure communication between hosts and during job submission.

SSH is used when running any of the following:

- Remote log on to a lightly loaded host (`lslogin`)
- An interactive job (`bsub -IS | -ISp | ISS`)

- An X-window job (bsub -IX)
- An externally submitted job that is interactive or X-window (esub)

## Default

Not set. LSF uses rlogin to authenticate users.

# LSF\_MACHDEP

## Syntax

**LSF\_MACHDEP**=*directory*

## Description

Specifies the directory in which machine-dependent files are installed. These files cannot be shared across different types of machines.

In clusters with a single host type, LSF\_MACHDEP is usually the same as LSF\_INDEP. The machine dependent files are the user commands, daemons, and libraries. You should not need to modify this parameter.

As shown in the following list, LSF\_MACHDEP is incorporated into other LSF variables.

- LSF\_BINDIR=\$LSF\_MACHDEP/bin
- LSF\_LIBDIR=\$LSF\_MACHDEP/lib
- LSF\_SERVERDIR=\$LSF\_MACHDEP/etc
- XLSF\_UIDDIR=\$LSF\_MACHDEP/lib/uid

## Default

/usr/share/lsf

## See also

LSF\_INDEP

# LSF\_MANDIR

## Syntax

**LSF\_MANDIR**=*directory*

## Description

Directory under which all man pages are installed.

The man pages are placed in the man1, man3, man5, and man8 subdirectories of the LSF\_MANDIR directory. This is created by the LSF installation process, and you should not need to modify this parameter.

Man pages are installed in a format suitable for BSD-style man commands.

For most versions of UNIX and Linux, you should add the directory LSF\_MANDIR to your MANPATH environment variable. If your system has a man command that does not understand MANPATH, you should either install the man pages in the /usr/man directory or get one of the freely available man programs.

## Default

LSF\_INDEP/man

# LSF\_MASTER\_LIST

## Syntax

**LSF\_MASTER\_LIST**="*host\_name ...*"

## Description

Required. Defines a list of hosts that are candidates to become the master host for the cluster.

Listed hosts must be defined in `lsf.cluster.cluster_name`.

Host names are separated by spaces.

---

### Tip:

On UNIX and Linux, master host candidates should share LSF configuration and binaries. On Windows, configuration files are shared, but not binaries.

---

Starting in LSF 7, `LSF_MASTER_LIST` *must* be defined in `lsf.conf`.

If EGO is enabled, `LSF_MASTER_LIST` can only be defined in `lsf.conf`. `EGO_MASTER_LIST` can only be defined in `ego.conf`. `EGO_MASTER_LIST` cannot be defined in `lsf.conf`. `LSF_MASTER_LIST` cannot be defined in `ego.conf`.

LIM reads `EGO_MASTER_LIST` wherever it is defined. If both `LSF_MASTER_LIST` and `EGO_MASTER_LIST` are defined, the value of `EGO_MASTER_LIST` in `ego.conf` is taken. To avoid errors, you should make sure that the value of `LSF_MASTER_LIST` matches the value of `EGO_MASTER_LIST`, or define only `EGO_MASTER_LIST`.

If EGO is disabled, `ego.conf` not loaded and the value of `LSF_MASTER_LIST` defined in `lsf.conf` is taken.

When you run `lsadm in_reconfi g` to reconfigure the cluster, only the master LIM candidates read `lsf.shared` and `lsf.cluster.cluster_name` to get updated information. The elected master LIM sends configuration information to slave LIMs.

If you have a large number of non-master hosts, you should configure `LSF_LIM_IGNORE_CHECKSUM=Y` to ignore warning messages like the following logged to lim log files on non-master hosts.

```
Aug 26 13:47:35 2006 9746 4 8.0 xdr_loadvector: Sender <10.225.36.46:9999> has a
different configuration
```

## Interaction with LSF\_SERVER\_HOSTS

You can use the same list of hosts, or a subset of the master host list defined in `LSF_MASTER_LIST`, in `LSF_SERVER_HOSTS`. If you include the primary master host in `LSF_SERVER_HOSTS`, you should define it as the last host of the list.

If `LSF_ADD_CLIENTS` is defined in `install.config` at installation, `lsfinstall` automatically appends the hosts in `LSF_MASTER_LIST` to the list of hosts in `LSF_SERVER_HOSTS` so that the primary master host is last. For example:

```
LSF_MASTER_LIST="lsfmaster hostE"
```

```
LSF_SERVER_HOSTS="hostB hostC hostD hostE lsfmaster"
```

The value of `LSF_SERVER_HOSTS` is not changed during upgrade.

## EGO parameter

`EGO_MASTER_LIST`

## Default

Defined at installation

## See also

`LSF_LIM_IGNORE_CHECKSUM`

# LSF\_MASTER\_NSLOOKUP\_TIMEOUT

## Syntax

```
LSF_MASTER_NSLOOKUP_TIMEOUT=time_milliseconds
```

## Description

Timeout in milliseconds that the master LIM waits for DNS host name lookup.

If LIM spends a lot of time calling DNS to look up a host name, LIM appears to hang.

This parameter is used by master LIM only. Only the master LIM detects this parameter and enable the DNS lookup timeout.

## Default

Not defined. No timeout for DNS lookup

## See also

`LSF_LIM_IGNORE_CHECKSUM`

# LSF\_MAX\_TRY\_ADD\_HOST

## Syntax

```
LSF_MAX_TRY_ADD_HOST=integer
```

## Description

When a slave LIM on a dynamically added host sends an add host request to the master LIM, but master LIM cannot add the host for some reason, the slave LIM tries again. `LSF_MAX_TRY_ADD_HOST` specifies how many times the slave LIM retries the add host request before giving up.

## Default

20

# LSF\_MC\_NON\_PRIVILEGED\_PORTS

## Syntax

**LSF\_MC\_NON\_PRIVILEGED\_PORTS=y | Y**

## Description

MultiCluster only. If this parameter is enabled in one cluster, it must be enabled in all clusters.

Specify Y to make LSF daemons use non-privileged ports for communication across clusters.

## Compatibility

This disables privileged port daemon authentication, which is a security feature. If security is a concern, you should use `eauth` for LSF daemon authentication (see `LSF_AUTH_DAEMONS` in `lsf.conf`).

## Default

Not defined. LSF daemons use privileged port authentication

# LSF\_MONITOR\_LICENSE\_TOOL

## Syntax

**LSF\_MONITOR\_LICENSE\_TOOL=y | Y**

## Description

Specify Y to enable data collection by `l i m` for the command option `l sadmi n l sf l i c`.

## Default

Not defined. `l i m` ignores requests from `l sadmi n`, closing the channel.

# LSF\_MISC

## Syntax

**LSF\_MISC=*directory***

## Description

Directory in which miscellaneous machine independent files, such as example source programs and scripts, are installed.

## Default

`LSF_CONFDIR/mi sc`

# LSF\_NIOS\_DEBUG

## Syntax

**LSF\_NIOS\_DEBUG=1**

## Description

Enables NIOS debugging for interactive jobs (if LSF\_NIOS\_DEBUG=1).

NIOS debug messages are written to standard error.

This parameter can also be defined as an environment variable.

When LSF\_NIOS\_DEBUG and LSF\_CMD\_LOGDIR are defined, NIOS debug messages are logged in `ni os. log. host_name` in the location specified by LSF\_CMD\_LOGDIR.

If LSF\_NIOS\_DEBUG is defined, and the directory defined by LSF\_CMD\_LOGDIR is inaccessible, NIOS debug messages are logged to `/tmp/ni os. log. host_name` instead of `stderr`.

On Windows, NIOS debug messages are also logged to the temporary directory.

## Default

Not defined

## See also

LSF\_CMD\_LOGDIR, LSF\_CMD\_LOG\_MASK, LSF\_LOG\_MASK, LSF\_LOGDIR

# LSF\_NIOS\_ERR\_LOGDIR

## Syntax

**LSF\_NIOS\_ERR\_LOGDIR=*directory***

## Description

Applies to Windows only.

If LSF\_NIOS\_ERR\_LOGDIR is specified, logs NIOS errors to *directory/ni os. error. log. hostname. txt*.

If the attempt fails, LSF tries to write to another directory instead. The order is:

1. the specified log directory
2. LSF\_TMPDIR
3. %TMP%
4. %TEMP%
5. the system directory, for example, C:\winnt

If LSF\_NIOS\_DEBUG is also specified, NIOS debugging overrides the LSF\_NIOS\_ERR\_LOGDIR setting.

LSF\_NIOS\_ERR\_LOGDIR is an alternative to using the NIOS debug functionality.

This parameter can also be defined as an environment variable.

## Default

Not defined

## See also

LSF\_NIOS\_DEBUG, LSF\_CMD\_LOGDIR

# LSF\_NIOS\_JOBSTATUS\_INTERVAL

## Syntax

**LSF\_NIOS\_JOBSTATUS\_INTERVAL**=*time\_minutes*

## Description

Applies only to interactive batch jobs.

Time interval at which NIOS polls `mbat chd` to check if a job is still running. Used to retrieve a job's exit status in the case of an abnormal exit of NIOS, due to a network failure for example.

Use this parameter if you run interactive jobs and you have scripts that depend on an exit code being returned.

When this parameter is not defined and a network connection is lost, `mbat chd` cannot communicate with NIOS and the return code of a job is not retrieved.

When this parameter is defined, before exiting, NIOS polls `mbat chd` on the interval defined by `LSF_NIOS_JOBSTATUS_INTERVAL` to check if a job is still running. NIOS continues to poll `mbat chd` until it receives an exit code or `mbat chd` responds that the job does not exist (if the job has already been cleaned from memory for example).

If an exit code cannot be retrieved, NIOS generates an error message and the code -11.

## Valid values

Any integer greater than zero

## Default

Not defined

## Notes

Set this parameter to large intervals such as 15 minutes or more so that performance is not negatively affected if interactive jobs are pending for too long. NIOS always calls `mbat chd` on the defined interval to confirm that a job is still pending and this may add load to `mbat chd`.

## See also

Environment variable `LSF_NIOS_PEND_TIMEOUT`

# LSF\_NIOS\_MAX\_TASKS

## Syntax

**LSF\_NIOS\_MAX\_TASKS**=*integer*

## Description

Specifies the maximum number of NIOS tasks.

## Default

Not defined

# LSF\_NIOS\_RES\_HEARTBEAT

## Syntax

**LSF\_NIOS\_RES\_HEARTBEAT**=*time\_minutes*

## Description

Applies only to interactive non-parallel batch jobs.

Defines how long NIOS waits before sending a message to RES to determine if the connection is still open.

Use this parameter to ensure NIOS exits when a network failure occurs instead of waiting indefinitely for notification that a job has been completed. When a network connection is lost, RES cannot communicate with NIOS and as a result, NIOS does not exit.

When this parameter is defined, if there has been no communication between RES and NIOS for the defined period of time, NIOS sends a message to RES to see if the connection is still open. If the connection is no longer available, NIOS exits.

## Valid values

Any integer greater than zero

## Default

Not defined

## Notes

The time you set this parameter to depends how long you want to allow NIOS to wait before exiting. Typically, it can be a number of hours or days. Too low a number may add load to the system.

# LSF\_NON\_PRIVILEGED\_PORTS

## Syntax

**LSF\_NON\_PRIVILEGED\_PORTS**=*y* | *Y*

## Description

Disables privileged ports usage.

By default, LSF daemons and clients running under root account use privileged ports to communicate with each other. Without LSF\_NON\_PRIVILEGED\_PORTS defined, and if LSF\_AUTH is not defined in `lsf.conf`, LSF daemons check privileged port of request message to do authentication.

If `LSF_NON_PRIVILEGED_PORTS=Y` is defined, LSF clients (LSF commands and daemons) do not use privileged ports to communicate with daemons and LSF daemons do not check privileged ports of incoming requests to do authentication.

## LSF\_PAM\_APPL\_CHKPNT

### Syntax

`LSF_PAM_APPL_CHKPNT=Y | N`

### Description

When set to Y, allows PAM to function together with application checkpointing support.

### Default

Y

## LSF\_PAM\_CLEAN\_JOB\_DELAY

### Syntax

`LSF_PAM_CLEAN_JOB_DELAY=time_seconds`

### Description

The number of seconds LSF waits before killing a parallel job with failed tasks. Specifying `LSF_PAM_CLEAN_JOB_DELAY` implies that if any parallel tasks fail, the entire job should exit without running the other tasks in the job. The job is killed if any task exits with a non-zero exit code.

Specify a value greater than or equal to zero (0).

Applies only to PAM jobs.

### Default

Undefined: LSF kills the job immediately

## LSF\_PAM\_HOSTLIST\_USE

### Syntax

`LSF_PAM_HOSTLIST_USE=unique`

### Description

Used to start applications that use both OpenMP and MPI.

### Valid values

`unique`

### Default

Not defined

## Notes

At job submission, LSF reserves the correct number of processors and PAM starts only 1 process per host. For example, to reserve 32 processors and run on 4 processes per host, resulting in the use of 8 hosts:

```
bsub -n 32 -R "span[ptile=4]" pam yourOpenMPJob
```

## Where defined

This parameter can alternatively be set as an environment variable. For example:

```
set env LSF_PAM_HOSTLIST_USE unique
```

# LSF\_PAM\_PLUGINDIR

## Syntax

```
LSF_PAM_PLUGINDIR=path
```

## Description

The path to `libpamvcl.so`. Used with Platform LSF HPC features.

## Default

Path to `LSF_LIBDIR`

# LSF\_PAM\_USE\_ASH

## Syntax

```
LSF_PAM_USE_ASH=y | Y
```

## Description

Enables LSF to use the SGI IRIX Array Session Handles (ASH) to propagate signals to the parallel jobs.

See the IRIX system documentation and the `array_session(5)` man page for more information about array sessions.

## Default

Not defined

# LSF\_PASSWD\_DIR

## Syntax

```
LSF_PASSWD_DIR=file_path
```

## Description

Defines a location for LSF to load and update the `passwd.lsfuser` file.

Specify the full path to a shared directory accessible by all master candidate hosts. The LSF `lim` daemon must have read and write permissions on this directory.

By default, `passwd.lsfuser` is located in `$LSF_CONFDIR`. The default location is only used if `LSF_PASSWD_DIR` is undefined; if you define a new location and `lim` fails to access `passwd.lsfuser` in `LSF_PASSWD_DIR`, it will not check `$LSF_CONFDIR`.

You must restart `lim` to make changes take effect.

## Default

Not defined (`passwd.lsfuser` is located in `$LSF_CONFDIR`)

# LSF\_PIM\_INFODIR

## Syntax

**LSF\_PIM\_INFODIR=*path***

## Description

The path to where PIM writes the `pim.info.host_name` file.

Specifies the path to where the process information is stored. The process information resides in the file `pim.info.host_name`. The PIM also reads this file when it starts so that it can accumulate the resource usage of dead processes for existing process groups.

## EGO parameter

EGO\_PIM\_INFODIR

## Default

Not defined. The system uses `/tmp`.

# LSF\_PIM\_LINUX\_ENHANCE

## Syntax

**LSF\_PIM\_LINUX\_ENHANCE=Y | N**

## Description

When enabled, the PIM daemon reports proportional memory utilization for each process attached to a shared memory segment. The sum total of memory utilization of all processes on the host is now accurately reflected in the total memory used. (The Linux kernel must be version 2.6.14 or newer.)

When `EGO_PIM_SWAP_REPORT` is set, the swap amount is correctly reported. The swap amount is the virtual memory minus the value of the `rss` value in the static Linux file.

Applies only to Linux operating systems and Red Hat Enterprise Linux 4.7.5.0.

## Default

Not defined.

## LSF\_PIM\_SLEEPTIME

### Syntax

**LSF\_PIM\_SLEEPTIME**=*time\_seconds*

### Description

The reporting period for PIM.

PIM updates the process information every 15 seconds unless an application queries this information. If an application requests the information, PIM updates the process information every LSF\_PIM\_SLEEPTIME seconds. If the information is not queried by any application for more than 5 minutes, the PIM reverts back to the 15 second update period.

### EGO parameter

EGO\_PIM\_SLEEPTIME

### Default

30 seconds

## LSF\_PIM\_SLEEPTIME\_UPDATE

### Syntax

**LSF\_PIM\_SLEEPTIME\_UPDATE**=*y* | *n*

### Description

UNIX only.

Use this parameter to improve job throughput and reduce a job's start time if there are many jobs running simultaneously on a host. This parameter reduces communication traffic between sbat chd and PIM on the same host.

When this parameter is not defined or set to n, sbat chd queries PIM as needed for job process information.

When this parameter is defined, sbat chd does not query PIM immediately as it needs information; sbat chd only queries PIM every LSF\_PIM\_SLEEPTIME seconds.

### Limitations

When this parameter is defined:

- sbat chd may be intermittently unable to retrieve process information for jobs whose run time is smaller than LSF\_PIM\_SLEEPTIME.
- It may take longer to view resource usage with `bj obs -l`.

### EGO parameter

EGO\_PIM\_SLEEPTIME\_UPDATE

## Default

Not defined

# LSF\_POE\_TIMEOUT\_BIND

## Syntax

**LSF\_POE\_TIMEOUT\_BIND**=*time\_seconds*

## Description

Specifies the time in seconds for the `poe_w` wrapper to keep trying to set up a server socket to listen on.

`poe_w` is the wrapper for the IBM poe driver program.

`LSF_POE_TIMEOUT_BIND` can also be set as an environment variable for `poe_w` to read.

## Default

120 seconds

# LSF\_POE\_TIMEOUT\_SELECT

## Syntax

**LSF\_POE\_TIMEOUT\_SELECT**=*time\_seconds*

## Description

Specifies the time in seconds for the `poe_w` wrapper to wait for connections from the `pmd_w` wrapper. `pmd_w` is the wrapper for `pmd` (IBM PE Partition Manager Daemon).

`LSF_POE_TIMEOUT_SELECT` can also be set as an environment variable for `poe_w` to read.

## Default

160 seconds

# LSF\_REMOTE\_COPY\_CMD

## Syntax

**LSF\_REMOTE\_COPY\_CMD**="*copy\_command*"

## Description

UNIX only. Specifies the shell command or script to use with the following LSF commands if RES fails to copy the file between hosts.

- `lsrcp`
- `bsub -i, -f, -i s, -Zs "Ci(s)`
- `bmod -Zs`

By default, `rcp` is used for these commands.

There is no need to restart any daemons when this parameter changes.

For example, to use `scp` instead of `rcp` for remote file copying, specify:

```
LSF_REMOTE_COPY_CMD="scp -B -o 'StrictHostKeyChecking no' "
```

You can also configure ssh options such as BatchMode, StrictHostKeyChecking in the global SSH\_ETC/ssh\_config file or \$HOME/.ssh/config.

When remote copy of a file via RES fails, the environment variable "LSF\_LSRCP\_ERRNO" is set to the system defined errno. You can use this variable in a self-defined shell script executed by lsrcp. The script can do the appropriate cleanup, recopy, or retry, or it can just exit without invoking any other copy command.

LSF automatically appends two parameters before executing the command:

- The first parameter is the source file path.
- The second parameter is the destination file path.

## Valid values

Values are passed directly through. Any valid scp, rcp, or custom copy commands and options are supported except for compound multi-commands. For example, set LSF\_REMOTE\_COPY\_CMD="**scp -B -o 'StrictHostKeyChecking no'**".

To avoid a recursive loop, the value of LSF\_REMOTE\_COPY\_CMD must not be lsrcp or a shell script executing lsrcp.

## Default

Not defined.

# LSF\_RES\_ACCT

## Syntax

```
LSF_RES_ACCT=time_milliseconds | 0
```

## Description

If this parameter is defined, RES logs information for completed and failed tasks by default (see lsf.acct).

The value for LSF\_RES\_ACCT is specified in terms of consumed CPU time (milliseconds). Only tasks that have consumed more than the specified CPU time are logged.

If this parameter is defined as LSF\_RES\_ACCT=0, then all tasks are logged.

For those tasks that consume the specified amount of CPU time, RES generates a record and appends the record to the task log file lsf.acct.host\_name. This file is located in the LSF\_RES\_ACCTDIR directory.

If this parameter is not defined, the LSF administrator must use the lsadmi n command (see lsadmi n) to turn task logging on after RES has started.

## Default

Not defined

## See also

LSF\_RES\_ACCTDIR

## LSF\_RES\_ACCTDIR

### Syntax

**LSF\_RES\_ACCTDIR**=*directory*

### Description

The directory in which the RES task log file `lsf.acct.host_name` is stored.

If **LSF\_RES\_ACCTDIR** is not defined, the log file is stored in the `/tmp` directory.

### Default

(UNIX) `/tmp`

(Windows) `C: \temp`

### See also

**LSF\_RES\_ACCT**

## LSF\_RES\_ACTIVE\_TIME

### Syntax

**LSF\_RES\_ACTIVE\_TIME**=*time\_seconds*

### Description

Time in seconds before LIM reports that RES is down.

### Minimum value

10 seconds

### Default

90 seconds

## LSF\_RES\_CLIENT\_TIMEOUT

### Syntax

**LSF\_RES\_CLIENT\_TIMEOUT**=*time\_minutes*

### Description

Specifies in minutes how long an application RES waits for a new task before exiting.

---

**Caution:**

If you use the LSF API to run remote tasks and you define this parameter with timeout, the remote execution of the new task fails (for example, `ls_rtask()`).

---

## Default

The parameter is not set; the application RES waits indefinitely for new task to come until client tells it to quit.

# LSF\_RES\_CONNECT\_RETRY

## Syntax

**LSF\_RES\_CONNECT\_RETRY=*integer* / 0**

## Description

The number of attempts by RES to reconnect to NIOS.

If LSF\_RES\_CONNECT\_RETRY is not defined, the default value is used.

## Default

0

## See also

LSF\_NIOS\_RES\_HEARTBEAT

# LSF\_RES\_DEBUG

## Syntax

**LSF\_RES\_DEBUG=1 / 2**

## Description

Sets RES to debug mode.

If LSF\_RES\_DEBUG is defined, the Remote Execution Server (RES) operates in single user mode. No security checking is performed, so RES should not run as root. RES does not look in the services database for the RES service port number. Instead, it uses port number 36002 unless LSF\_RES\_PORT has been defined.

Specify 1 for this parameter unless you are testing RES.

## Valid values

LSF\_RES\_DEBUG=1

RES runs in the background with no associated control terminal.

LSF\_RES\_DEBUG=2

RES runs in the foreground and prints error messages to `tty`.

## Default

Not defined

## See also

LSF\_LIM\_DEBUG, LSF\_CMD\_LOGDIR, LSF\_CMD\_LOG\_MASK, LSF\_LOG\_MASK, LSF\_LOGDIR

## LSF\_RES\_PORT

See LSF\_LIM\_PORT, LSF\_RES\_PORT, LSB\_MBD\_PORT, LSB\_SBD\_PORT.

## LSF\_RES\_RLIMIT\_UNLIM

### Syntax

**LSF\_RES\_RLIMIT\_UNLIM=cpu | fsize | data | stack | core | vmem**

### Description

By default, RES sets the hard limits for a remote task to be the same as the hard limits of the local process. This parameter specifies those hard limits which are to be set to unlimited, instead of inheriting those of the local process.

Valid values are `cpu`, `fsize`, `data`, `stack`, `core`, and `vmem`, for CPU, file size, data size, stack, core size, and virtual memory limits, respectively.

### Example

The following example sets the CPU, core size, and stack hard limits to be unlimited for all remote tasks:

```
LSF_RES_RLIMIT_UNLIM="cpu core stack"
```

## Default

Not defined

## LSF\_RES\_TIMEOUT

### Syntax

**LSF\_RES\_TIMEOUT=*time\_seconds***

### Description

Timeout when communicating with RES.

## Default

15

## LSF\_ROOT\_REX

### Syntax

**LSF\_ROOT\_REX=local**

## Description

UNIX only.

Allows root remote execution privileges (subject to identification checking) on remote hosts, for both interactive and batch jobs. Causes RES to accept requests from the superuser (root) on remote hosts, subject to identification checking.

If `LSF_ROOT_REX` is not defined, remote execution requests from user root are refused.

## Theory

Sites that have separate root accounts on different hosts within the cluster should not define `LSF_ROOT_REX`. Otherwise, this setting should be based on local security policies.

The `lsf.conf` file is host-type specific and not shared across different platforms. You must make sure that `lsf.conf` for all your host types are changed consistently.

## Default

Not defined. Root execution is not allowed.

## See also

`LSF_TIME_CMD`, `LSF_AUTH`

## LSF\_RSH

## Syntax

`LSF_RSH=command [command_options]`

## Description

Specifies shell commands to use when the following LSF commands require remote execution:

- `badmi n hstartup`
- `bpeek`
- `lsadmi n limstartup`
- `lsadmi n resstartup`
- `lsfrestart`
- `lsfshutdown`
- `lsfstartup`
- `lsrcp`

By default, `rsh` is used for these commands. Use `LSF_RSH` to enable support for `ssh`.

## EGO parameter

`EGO_RSH`

## Default

Not defined

## Example

To use an ssh command before trying rsh for LSF commands, specify:

```
LSF_RSH="ssh -o 'PasswordAuthentication no' -o 'StrictHostKeyChecking no' "
```

ssh options such as PasswordAuthentication and StrictHostKeyChecking can also be configured in the global SSH\_ETC/ssh\_conf ig file or \$HOME/. ssh/conf ig.

## See also

ssh, ssh\_conf ig

## LSF\_SECUREDIR

### Syntax

```
LSF_SECUREDIR=path
```

### Description

Windows only; mandatory if using lsf . sudoers.

Path to the directory that contains the file lsf . sudoers (shared on an NTFS file system).

## LSF\_SERVER\_HOSTS

### Syntax

```
LSF_SERVER_HOSTS="host_name ..."
```

### Description

Defines one or more server hosts that the client should contact to find a Load Information Manager (LIM). LSF server hosts are hosts that run LSF daemons and provide loading-sharing services. Client hosts are hosts that only run LSF commands or applications but do not provide services to any hosts.

---

#### Important:

LSF\_SERVER\_HOSTS is required for non-shared slave hosts.

---

Use this parameter to ensure that commands execute successfully when no LIM is running on the local host, or when the local LIM has just started. The client contacts the LIM on one of the LSF\_SERVER\_HOSTS and execute the command, provided that at least one of the hosts defined in the list has a LIM that is up and running.

If LSF\_SERVER\_HOSTS is not defined, the client tries to contact the LIM on the local host.

The host names in LSF\_SERVER\_HOSTS must be enclosed in quotes and separated by white space. For example:

```
LSF_SERVER_HOSTS="hostA hostD hostB"
```

The parameter string can include up to 4094 characters for UNIX or 255 characters for Windows.

## Interaction with LSF\_MASTER\_LIST

Starting in LSF 7, LSF\_MASTER\_LIST must be defined in `lsf.conf`. You can use the same list of hosts, or a subset of the master host list, in LSF\_SERVER\_HOSTS. If you include the primary master host in LSF\_SERVER\_HOSTS, you should define it as the last host of the list.

If LSF\_ADD\_CLIENTS is defined in `install.config` at installation, `lsfinstall` automatically appends the hosts in LSF\_MASTER\_LIST to the list of hosts in LSF\_SERVER\_HOSTS so that the primary master host is last. For example:

```
LSF_MASTER_LIST="lsfmaster hostE"
```

```
LSF_SERVER_HOSTS="hostB hostC hostD hostE lsfmaster"
```

```
LSF_ADD_CLIENTS="clientHostA"
```

The value of LSF\_SERVER\_HOSTS is not changed during upgrade.

## Default

Not defined

## See also

LSF\_MASTER\_LIST

## LSF\_SERVERDIR

### Syntax

```
LSF_SERVERDIR=directory
```

### Description

Directory in which all server binaries and shell scripts are installed.

These include `lim`, `res`, `ni os`, `sbat chd`, `mbat chd`, and `mbschd`. If you use `elim`, `eauth`, `eexec`, `esub`, etc, they are also installed in this directory.

## Default

LSF\_MACHDEP/etc

## See also

LSB\_ECHKPNT\_METHOD\_DIR

## LSF\_SHELL\_AT\_USERS

### Syntax

```
LSF_SHELL_AT_USERS="user_name user_name ..."
```

### Description

Applies to `lscsh` only. Specifies users who are allowed to use `@` for host redirection. Users not specified with this parameter cannot use host redirection in `lscsh`. To specify a Windows user account, include the domain name in uppercase letters (`DOMAIN_NAME\user_name`).

If this parameter is not defined, all users are allowed to use @ for host redirection in `l st csh`.

## Default

Not defined

# LSF\_SHIFT\_JIS\_INPUT

## Syntax

`LSF_SHIFT_JIS_INPUT=y | n`

## Description

Enables LSF to accept Shift-JIS character encoding for job information (for example, user names, queue names, job names, job group names, project names, commands and arguments, esub parameters, external messages, etc.)

## Default

n

# LSF\_STRICT\_CHECKING

## Syntax

`LSF_STRICT_CHECKING=Y`

## Description

If set, enables more strict checking of communications between LSF daemons and between LSF commands and daemons when LSF is used in an untrusted environment, such as a public network like the Internet.

If you enable this parameter, you must enable it in the entire cluster, as it affects all communications within LSF. If it is used in a MultiCluster environment, it must be enabled in all clusters, or none. Ensure that all binaries and libraries are upgraded to LSF Version 7, including `LSF_BINDIR`, `LSF_SERVERDIR` and `LSF_LIBDIR` directories, if you enable this parameter.

If your site uses any programs that use the LSF base and batch APIs, or LSF MPI (Message Passing Interface), they need to be recompiled using the LSF Version 7 APIs before they can work properly with this option enabled.

---

### Important:

You must shut down the entire cluster before enabling or disabling this parameter.

If `LSF_STRICT_CHECKING` is defined, and your cluster has slave hosts that are dynamically added, `LSF_STRICT_CHECKING` must be configured in the local `lsf.conf` on all slave hosts.

---

## Valid value

Set to Y to enable this feature.

## Default

Not defined. LSF is secure in trusted environments.

# LSF\_STRICT\_RESREQ

## Syntax

**LSF\_STRICT\_RESREQ=Y | N**

## Description

When LSF\_STRICT\_RESREQ=Y, the resource requirement selection string must conform to the stricter resource requirement syntax described in *Administering Platform LSF*. The strict resource requirement syntax only applies to the select section. It does not apply to the other resource requirement sections (order, rusage, same, span, or cu).

When LSF\_STRICT\_RESREQ=Y in lsf.conf, LSF rejects resource requirement strings where an rusage section contains a non-consumable resource.

When LSF\_STRICT\_RESREQ=N, the default resource requirement selection string evaluation is performed.

## Default

N

# LSF\_STRIP\_DOMAIN

## Syntax

**LSF\_STRIP\_DOMAIN=domain\_suffix[:domain\_suffix ...]**

## Description

(Optional) If all of the hosts in your cluster can be reached using short host names, you can configure LSF to use the short host names by specifying the portion of the domain name to remove. If your hosts are in more than one domain or have more than one domain name, you can specify more than one domain suffix to remove, separated by a colon (:).

For example, given this definition of LSF\_STRIP\_DOMAIN,

```
LSF_STRIP_DOMAIN=.foo.com:.bar.com
```

LSF accepts hostA, hostA.foo.com, and hostA.bar.com as names for host hostA, and uses the name hostA in all output. The leading period '.' is required.

Example:

```
LSF_STRIP_DOMAIN=.platform.com:.generic.com
```

In the above example, LSF accepts hostA, hostA.platform.com, and hostA.generic.com as names for hostA, and uses the name hostA in all output.

Setting this parameter only affects host names displayed through LSF, it does not affect DNS host lookup.

After adding or changing LSF\_STRIP\_DOMAIN, use lsadmin reconfig and badmin mbdrestart to reconfigure your cluster.

## EGO parameter

EGO\_STRIP\_DOMAIN

## Default

Not defined

## LSF\_TIME\_CMD

### Syntax

**LSF\_TIME\_CMD**=*timing\_level*

### Description

The timing level for checking how long LSF commands run. Time usage is logged in milliseconds. Specify a positive integer.

### Default

Not defined

### See also

LSB\_TIME\_MBD, LSB\_TIME\_SBD, LSB\_TIME\_CMD, LSF\_TIME\_LIM, LSF\_TIME\_RES

## LSF\_TIME\_LIM

### Syntax

**LSF\_TIME\_LIM**=*timing\_level*

### Description

The timing level for checking how long LIM routines run.  
Time usage is logged in milliseconds. Specify a positive integer.

## EGO parameter

EGO\_TIME\_LIM

## Default

Not defined

### See also

LSB\_TIME\_CMD, LSB\_TIME\_MBD, LSB\_TIME\_SBD, LSF\_TIME\_RES

## LSF\_TIME\_RES

### Syntax

**LSF\_TIME\_RES**=*timing\_level*

## Description

The timing level for checking how long RES routines run.

Time usage is logged in milliseconds. Specify a positive integer.

LSF\_TIME\_RES is not supported on Windows.

## Default

Not defined

## See also

LSB\_TIME\_CMD, LSB\_TIME\_MBD, LSB\_TIME\_SBD, LSF\_TIME\_LIM

# LSF\_TMPDIR

## Syntax

**LSF\_TMPDIR**=*directory*

## Description

Specifies the path and directory for temporary job output.

When LSF\_TMPDIR is defined in `lsf.conf`, LSF creates a temporary directory under the directory specified by LSF\_TMPDIR on the execution host when a job is started and sets the temporary directory environment variable (TMPDIR) for the job.

The name of the temporary directory has the following format:

```
$LSF_TMPDIR/job_ID.tmpdir
```

On UNIX, the directory has the permission 0700 and is owned by the execution user.

After adding LSF\_TMPDIR to `lsf.conf`, use `badm in hrestart all` to reconfigure your cluster.

If LSB\_SET\_TMPDIR=Y, the environment variable TMPDIR will be set equal to the path specified by LSF\_TMPDIR.

If the path specified by LSF\_TMPDIR does not exist, the value of TMPDIR is set to the default path `/tmp/job_ID.tmpdir`.

## Valid values

Specify any valid path up to a maximum length of 256 characters. The 256 character maximum path length includes the temporary directories and files that the system creates as jobs run. The path that you specify for LSF\_TMPDIR should be as short as possible to avoid exceeding this limit.

## UNIX

Specify an absolute path. For example:

```
LSF_TMPDIR=/usr/share/lsf_tmp
```

## Windows

Specify a UNC path or a path with a drive letter. For example:

```
LSF_TMPDIR=\\HostA\temp\lsf_tmp
```

```
LSF_TMPDIR=D:\temp\lsf_tmp
```

## Temporary directory for tasks launched by blaunch

By default, LSF creates a temporary directory for a job only on the first execution host. If `LSF_TMPDIR` is set in `lsf.conf`, the path of the job temporary directory on the first execution host is set to `LSF_TMPDIR/job_ID.tmpdir`.

If `LSB_SET_TMPDIR=Y`, the environment variable `TMPDIR` will be set equal to the path specified by `LSF_TMPDIR`.

Tasks launched through the `blaunch` distributed application framework make use of the LSF temporary directory specified by `LSF_TMPDIR`:

- When the environment variable `TMPDIR` is set on the first execution host, the `blaunch` framework propagates this environment variable to all execution hosts when launching remote tasks
- The job `RES` or the task `RES` creates the directory specified by `TMPDIR` if it does not already exist before starting the job
- The directory created by the job `RES` or task `RES` has permission `0700` and is owned by the execution user
- If the `TMPDIR` directory was created by the task `RES`, LSF deletes the temporary directory and its contents when the task is complete
- If the `TMPDIR` directory was created by the job `RES`, LSF will delete the temporary directory and its contents when the job is done
- If the `TMPDIR` directory is on a shared file system, it is assumed to be shared by all the hosts allocated to the `blaunch` job, so LSF *does not* remove `TMPDIR` directories created by the job `RES` or task `RES`

## Default

By default, `LSF_TMPDIR` is not enabled. If `LSF_TMPDIR` is not specified in `lsf.conf`, this parameter is defined as follows:

- On UNIX: `$TMPDIR/job_ID.tmpdir` or `/tmp/job_ID.tmpdir`
- On Windows: `%TMP%`, `%TEMP%`, or `%SystemRoot%`

## LSF\_ULDB\_DOMAIN

### Syntax

```
LSF_ULDB_DOMAIN="domain_name ..."
```

### Description

`LSF_ULDB_DOMAIN` specifies the name of the LSF domain in the ULDB domain directive. A domain definition of name `domain_name` must be configured in the SGI IRIX `joblimits.in` input file.

Used with IRIX User Limits Database (ULDB). Configures LSF so that jobs submitted to a host with the IRIX job limits option installed are subject to the job limits configured in the IRIX User Limits Database (ULDB).

The ULDB contains job limit information that system administrators use to control access to a host on a per user basis. The job limits in the ULDB override the system default values for both job limits and process limits. When a ULDB domain is configured, the limits are enforced as IRIX job limits.

If the ULDB domain specified in `LSF_ULDB_DOMAIN` is not valid or does not exist, LSF uses the limits defined in the domain named `batch`. If the `batch` domain does not exist, then the system default limits are set.

When an LSF job is submitted, an IRIX job is created, and the job limits in the ULDB are applied.

Next, LSF resource usage limits are enforced for the IRIX job under which the LSF job is running. LSF limits override the corresponding IRIX job limits. The ULDB limits are used for any LSF limits that are not defined. If the job reaches the IRIX job limits, the action defined in the IRIX system is used.

IRIX job limits in the ULDB apply only to batch jobs.

See the IRIX resource administration documentation for information about configuring ULDB domains in the `limits.in` file.

## LSF resource usage limits controlled by ULDB

- `PROSSLIMIT`: Corresponds to `IRIX_JLIMIT_NUMPROC`; `fork()` fails, but the existing processes continue to run
- `MEMLIMIT`: Corresponds to `JLIMIT_RSS`; Resident pages above the limit become prime swap candidates
- `DATALIMIT`: Corresponds to `LIMIT_DATA`; `malloc()` calls in the job fail with `errno` set to `ENOMEM`
- `CPULIMIT`: Corresponds to `JLIMIT_CPU`; IRIX sends `SIGXCPU` signal to job, then after the grace period expires, sends `SIGINT`, `SIGTERM`, and `SIGKILL`
- `FILELIMIT`: No corresponding IRIX limit; use process limit `RLIMIT_FSIZE`
- `STACKLIMIT`: No corresponding IRIX limit; use process limit `RLIMIT_STACK`
- `CORELIMIT`: No corresponding IRIX limit; use process limit `RLIMIT_CORE`
- `SWAPLIMIT`: Corresponds to `JLIMIT_VMEM`; use process limit `RLIMIT_VMEM`

## Increase the default MEMLIMIT for ULDB

In some pre-defined LSF queues, such as `normal`, the default `MEMLIMIT` is set to 5000 (5 MB). However, if ULDB is enabled (`LSF_ULDB_DOMAIN` is defined) the `MEMLIMIT` should be set greater than 8000 in `lsb` queues.

## Default

Not defined

## LSF\_UNIT\_FOR\_LIMITS

### Syntax

`LSF_UNIT_FOR_LIMITS=unit`

### Description

Enables scaling of large units in resource usage limits.

When set, `LSF_UNIT_FOR_LIMITS` applies cluster-wide to limits at the job-level (`bsub`), queue-level (`lsb` queues), and application level (`lsb` applications).

The limit unit specified by `LSF_UNIT_FOR_LIMITS` also applies to limits modified with `bmod`, and the display of resource usage limits in query commands (`bacct`, `bapp`, `bhist`, `bhosts`, `bjobs`, `bqueues`, `lslload`, and `lshosts`).

---

### Important:

Before changing the units of your resource usage limits, you should completely drain the cluster of all workload. There should be no running, pending, or finished jobs in the system.

---

In a MultiCluster environment, you should configure the same unit for all clusters.

## Example

A job is submitted with `bsub -M 100` and `LSF_UNIT_FOR_LIMITS=MB`; the memory limit for the job is 100 MB rather than the default 100 KB.

## Valid values

*unit* indicates the unit for the resource usage limit, one of:

- KB (kilobytes)
- MB (megabytes)
- GB (gigabytes)
- TB (terabytes)
- PB (petabytes)
- EB (exabytes)

## Default

KB

# LSF\_USE\_HOSTEQUIV

## Syntax

`LSF_USE_HOSTEQUIV=y | Y`

## Description

(UNIX only; optional)

If `LSF_USE_HOSTEQUIV` is defined, `RES` and `mbatchd` call the `ruserok()` function to decide if a user is allowed to run remote jobs.

The `ruserok()` function checks in the `/etc/hostsequiv` file and the user's `$HOME/.rhosts` file to decide if the user has permission to execute remote jobs.

If `LSF_USE_HOSTEQUIV` is not defined, all normal users in the cluster can execute remote jobs on any host.

If `LSF_ROOT_REX` is set, root can also execute remote jobs with the same permission test as for normal users.

## Default

Not defined

## See also

LSF\_ROOT\_REX

# LSF\_USER\_DOMAIN

## Syntax

**LSF\_USER\_DOMAIN**=*domain\_name:domain\_name:domain\_name...*

## Description

Enables the UNIX/Windows user account mapping feature, which allows cross-platform job submission and execution in a mixed UNIX/Windows environment. **LSF\_USER\_DOMAIN** specifies one or more Windows domains that LSF either strips from the user account name when a job runs on a UNIX host, or adds to the user account name when a job runs on a Windows host.

---

### Important:

Configure **LSF\_USER\_DOMAIN** immediately after you install LSF; changing this parameter in an existing cluster requires that you verify and possibly reconfigure service accounts, user group memberships, and user passwords.

---

Specify one or more Windows domains, separated by a colon (:). You can enter an unlimited number of Windows domains. A period (.) specifies a local account, not a domain.

## Examples

```
LSF_USER_DOMAIN=BUSINESS
```

```
LSF_USER_DOMAIN=BUSINESS:ENGINEERING:SUPPORT
```

## Default

The default depends on your LSF installation:

- If you upgrade a cluster to LSF version 7, the default is the existing value of **LSF\_USER\_DOMAIN**, if defined
- For a new cluster, this parameter is not defined, and UNIX/Windows user account mapping is not enabled

# LSF\_VPLUGIN

## Syntax

**LSF\_VPLUGIN**=*path*

## Description

The full path to the vendor MPI library `libxmpi.so`. Used with Platform LSF HPC features.

For PAM to access the SGI MPI `libxmpi.so` library, the file permission mode must be `755 (-rwxr-xr-x)`.

## Examples

- Platform MPI: `LSF_VPLUGIN=/opt/mpi/lib/pa1.1/libmpi.rml`
- SGI MPI: `LSF_VPLUGIN=/usr/lib32/libxmpi.so`
- SGI Linux (64-bit x-86 Linux 2.6, glibc 2.3.): `LSF_VPLUGIN=/usr/lib32/libxmpi.so: /usr/lib/libxmpi.so: /usr/lib64/libxmpi.so`

## Default

Not defined

# MC\_PLUGIN\_REMOTE\_RESOURCE

## Syntax

**MC\_PLUGIN\_REMOTE\_RESOURCE=y**

## Description

MultiCluster job forwarding model only. By default, the submission cluster does not consider remote resources. Define **MC\_PLUGIN\_REMOTE\_RESOURCE=y** in the submission cluster to allow consideration of remote resources.

---

### Note:

When `MC_PLUGIN_REMOTE_RESOURCE` is defined, only the following resource requirements (boolean only) are supported: `-R "type==type_name"`, `-R "same[type]"` and `-R "defined(resource_name)"`

---

### Note:

When `MC_PLUGIN_SCHEDULE_ENHANCE` in `lsb.params` is defined, remote resources are considered as if **MC\_PLUGIN\_REMOTE\_RESOURCE=Y** regardless of the actual value. In addition, details of the remote cluster workload are considered by the submission cluster scheduler.

---

## Default

Not defined. The submission cluster does not consider remote resources.

## See also

`MC_PLUGIN_SCHEDULE_ENHANCE` in `lsb.params`

# XLSF\_APPDIR

## Syntax

**XLSF\_APPDIR=directory**

## Description

(UNIX only; optional) Directory in which X application default files for LSF products are installed.

The LSF commands that use X look in this directory to find the application defaults. Users do not need to set environment variables to use the Platform LSF X applications. The application default files are platform-independent.

## Default

LSF\_I NDEP/mi sc

# XLSF\_UIDDIR

## Syntax

**XLSF\_UIDDIR**=*directory*

## Description

(UNIX only) Directory in which Motif User Interface Definition files are stored.

These files are platform-specific.

## Default

LSF\_LI BDI R/ui d

# lsf.licensescheduler

The `lsf.licensescheduler` file contains Platform License Scheduler configuration information. All sections except `ProjectGroup` are required.

The command `bl params` displays configuration information from this file.

## Changing lsf.licensescheduler configuration

After making any changes to `lsf.licensescheduler`, run the following commands:

- `bl admin reconfig` to reconfigure `bl d`
- `badmin mbdrestart` to restart `mbatchd`

## Parameters section

### Description

Required. Defines License Scheduler configuration parameters.

### Parameters section structure

The Parameters section begins and ends with the lines `Begin Parameters` and `End Parameters`. Each subsequent line describes one configuration parameter. All parameters are mandatory.

```
Begin Parameters
```

```
ADMIN=lsadmin
```

```
HOSTS=hostA hostB hostC
```

```
LMSTAT_PATH=/etc/lsf/lm/bin
```

```
LM_STAT_INTERVAL=30
```

```
PORT=9581
```

```
End Parameters
```

### Parameters

- ADMIN
- AUTH
- DISTRIBUTION\_POLICY\_VIOLATION\_ACTION
- ENABLE\_INTERACTIVE
- HOSTS
- LIB\_RECVTIMEOUT
- LM\_REMOVE\_INTERVAL
- LM\_STAT\_INTERVAL
- LMSTAT\_PATH
- LS\_DEBUG\_BLD
- LS\_LOG\_MASK
- LS\_MAX\_STREAM\_FILE\_NUMBER
- LS\_MAX\_STREAM\_SIZE
- LS\_MAX\_TASKMAN\_SESSIONS
- LS\_STREAM\_FILE

- LS\_PREEMPT\_PEER
- PORT
- BLC\_HEARTBEAT\_FACTOR

## ADMIN

### Syntax

**ADMIN=***user\_name* ...

### Description

Defines the License Scheduler administrator using a valid UNIX user account. You can specify multiple accounts.

## AUTH

### Syntax

**AUTH=Y**

### Description

Enables License Scheduler user authentication for projects for taskman jobs.

## DISTRIBUTION\_POLICY\_VIOLATION\_ACTION

### Syntax

**DISTRIBUTION\_POLICY\_VIOLATION\_ACTION=(***PERIOD* *reporting\_period* *CMD*  
*reporting\_command*)

#### ***reporting\_period***

Specify the keyword **PERIOD** with a positive integer representing the interval (a multiple of **LM\_STAT\_INTERVAL** periods) at which License Scheduler checks for distribution policy violations.

#### ***reporting\_command***

Specify the keyword **CMD** with the directory path and command that License Scheduler runs when reporting a violation.

### Description

Optional. Defines how License Scheduler handles distribution policy violations. Distribution policy violations are caused by non-LSF workloads; License Scheduler explicitly follows its distribution policies.

License Scheduler reports a distribution policy violation when the total number of licenses given to the LSF workload, both free and in use, is less than the LSF workload distribution specified in **WORKLOAD\_DISTRIBUTION**. If License Scheduler finds a distribution policy violation, it creates or overwrites the **LSF\_LOGDIR/bl d. v i o l a t i o n. s e r v i c e \_ d o m a i n \_ n a m e. l o g** file and runs the user command specified by the **CMD** keyword.

## Example

The `LicenseServer1` service domain has a total of 80 licenses, and its workload distribution and enforcement is configured as follows:

```
Begin Parameter
...
DISTRIBUTION_POLICY_VIOLATION_ACTION=(PERIOD 5 CMD /bin/mycmd)
...
End Parameter

Begin Feature
NAME=ApplicationX
DISTRIBUTION=LicenseServer1(Lp1 1 Lp2 2)
WORKLOAD_DISTRIBUTION=LicenseServer1(LSF 8 NON_LSF 2)
End Feature
```

According to this configuration, 80% of the available licenses, or 64 licenses, are available to the LSF workload. License Scheduler checks the service domain for a violation every five scheduling cycles, and runs the `/bin/mycmd` command if it finds a violation.

If the current LSF workload license usage is 50 and the number of free licenses is 10, the total number of licenses assigned to the LSF workload is 60. This is a violation of the workload distribution policy because this is less than the specified LSF workload distribution of 64 licenses.

## ENABLE\_INTERACTIVE

### Syntax

```
ENABLE_INTERACTIVE=Y
```

### Description

Optional. Globally enables one share of the licenses for interactive tasks.

---

#### Tip:

By default, `ENABLE_INTERACTIVE` is not set. License Scheduler allocates licenses equally to each cluster and does not distribute licenses for interactive tasks.

---

## HOSTS

### Syntax

```
HOSTS=host_name.domain_name ...
```

### Description

Defines License Scheduler hosts, including License Scheduler candidate hosts.

Specify a fully qualified host name such as `hostX.mycompany.com`. You can omit the domain name if all your License Scheduler clients run in the same DNS domain.

## LIB\_RECVTIMEOUT

### Syntax

**LIB\_RECVTIMEOUT**=*seconds*

### Description

Specifies a timeout value in seconds for communication between License Scheduler and LSF.

### Default

0 seconds

## LM\_REMOVE\_INTERVAL

### Syntax

**LM\_REMOVE\_INTERVAL**=*seconds*

### Description

Specifies the minimum time a job must have a license checked out before `l mremove` can remove the license. `l mremove` causes `l mgrd` and vendor daemons to close the TCP connection with the application. They then retry the license checkout.

This value overrides the `LS_WAIT_TO_PREEMPT` value if `LM_REMOVE_INTERVAL` is larger.

### Default

180 seconds

## LM\_STAT\_INTERVAL

### Syntax

**LM\_STAT\_INTERVAL**=*seconds*

### Description

Defines a time interval between calls that License Scheduler makes to collect license usage information from FlexNet license management.

### Default

60 seconds

## LMSTAT\_PATH

### Syntax

**LMSTAT\_PATH**=*path*

### Description

Defines the full path to the location of the FlexNet command `l mstat`.

## LS\_DEBUG\_BLD

### Syntax

**LS\_DEBUG\_BLD**=*log\_class*

### Description

Sets the debugging log class for the Platform License Scheduler bld daemon.

Specifies the log class filtering to be applied to bld. Messages belonging to the specified log class are recorded. Not all debug message are controlled by log class.

LS\_DEBUG\_BLD sets the log class and is used in combination with MASK, which sets the log level. For example:

```
LS_LOG_MASK=LOG_DEBUG LS_DEBUG_BLD="LC_TRACE"
```

To specify multiple log classes, use a space-separated list enclosed in quotation marks. For example:

```
LS_DEBUG_BLD="LC_TRACE"
```

You need to restart the bld daemon after setting LS\_DEBUG\_BLD for your changes to take effect.

If you use the command `bldadmin blddebug` to temporarily change this parameter without changing `lsf.licensescheduler`, you do not need to restart the daemons.

### Valid values

Valid log classes are:

- LC\_AUTH and LC2\_AUTH: Log authentication messages
- LC\_COMM and LC2\_COMM: Log communication messages
- LC\_FLEX - Log everything related to FLEX\_STAT or FLEX\_EXEC Flexera APIs
- LC\_LICENSE and LC2\_LICENSE : Log license management messages (LC\_LICENSE is also supported for backward compatibility)
- LC\_PREEMPT - Log license preemption policy messages
- LC\_RESREQ and LC2\_RESREQ: Log resource requirement messages
- LC\_TRACE and LC2\_TRACE: Log significant program walk steps
- LC\_XDR and LC2\_XDR: Log everything transferred by XDR

### Valid values

Valid log classes are the same as for LS\_DEBUG\_CMD.

### Default

Not defined.

## LS\_ENABLE\_MAX\_PREEMPT

### Syntax

**LS\_ENABLE\_MAX\_PREEMPT**=Y

### Description

Enables maximum preemption time checking for taskman jobs.

When `LS_ENABLE_MAX_PREEMPT` is disabled, preemption times for `taskman` job are not checked regardless of the value of parameters `LS_MAX_TASKMAN_PREEMPT` in `lsf.licensescheduler` and `MAX_JOB_PREEMPT` in `lsb.queues`, `lsb.applications`, or `lsb.params`.

## Default

N

## LS\_LOG\_MASK

### Syntax

`LS_LOG_MASK=message_log_level`

### Description

Specifies the logging level of error messages for Platform License Scheduler daemons. If `LS_LOG_MASK` is not defined in `lsf.licensescheduler`, the value of `LSF_LOG_MASK` in `lsf.conf` is used. If neither `LS_LOG_MASK` nor `LSF_LOG_MASK` is defined, the default is `LOG_WARNING`.

For example:

```
LS_LOG_MASK=LOG_DEBUG
```

The log levels in order from highest to lowest are:

- `LOG_WARNING`
- `LOG_DEBUG`
- `LOG_DEBUG1`
- `LOG_DEBUG2`
- `LOG_DEBUG3`

The most important License Scheduler log messages are at the `LOG_WARNING` level. Messages at the `LOG_DEBUG` level are only useful for debugging.

Although message log level implements similar functionality to UNIX `syslog`, there is no dependency on UNIX `syslog`. It works even if messages are being logged to files instead of `syslog`.

License Scheduler logs error messages in different levels so that you can choose to log all messages, or only log messages that are deemed critical. The level specified by `LS_LOG_MASK` determines which messages are recorded and which are discarded. All messages logged at the specified level or higher are recorded, while lower level messages are discarded.

For debugging purposes, the level `LOG_DEBUG` contains the fewest number of debugging messages and is used for basic debugging. The level `LOG_DEBUG3` records all debugging messages, and can cause log files to grow very large; it is not often used. Most debugging is done at the level `LOG_DEBUG2`.

## Default

`LOG_WARNING`

## LS\_MAX\_STREAM\_FILE\_NUMBER

### Syntax

`LS_MAX_STREAM_FILE_NUMBER=integer`

## Description

Sets the number of saved `bld.stream.timestamp` log files. When `LS_MAX_STREAM_FILE_NUMBER=2`, for example, the two most recent files are kept along with the current `bld.stream` file.

## Default

1 (one old `bld.stream` file is saved)

## LS\_MAX\_STREAM\_SIZE

### Syntax

`LS_MAX_STREAM_SIZE=integer`

## Description

Defines the maximum size of the `bld.stream` file in MB. once this size is reached an `EVENT_END_OF_STREAM` is logged, a new `bld.stream` file is created, and the old `bld.stream` file is renamed `bld.stream.timestamp`.

## Default

1024

## LS\_MAX\_TASKMAN\_PREEMPT

### Syntax

`LS_MAX_TASKMAN_PREEMPT=integer`

## Description

Defines the maximum number of times `taskman` jobs can be preempted.

Maximum preemption time checking for all jobs is enabled by `LS_ENABLE_MAX_PREEMPT`.

## Default

unlimited

## LS\_MAX\_TASKMAN\_SESSIONS

### Syntax

`LS_MAX_TASKMAN_SESSIONS=integer`

## Description

Defines the maximum number of `taskman` jobs that run simultaneously. This prevents system-wide performance issues that occur if there are a large number of `taskman` jobs running in License Scheduler.

## LS\_STREAM\_FILE

### Syntax

`LS_MAX_TASKMAN_PREEMPT=path`

## Description

Defines the full path and filename of the bld event log file, `bld.stream` by default.

---

### Note:

In Platform License Scheduler 8.0 the `bld.events` log file was replaced by the `bld.stream` log file.

---

## Default

LSF\_TOP/work/db/bld.stream

## LS\_PREEMPT\_PEER

### Syntax

**LS\_PREEMPT\_PEER=Y**

## Description

Enables bottom-up license token preemption in hierarchical project group configuration. License Scheduler attempts to preempt tokens from the closest projects in the hierarchy first. This balances token ownership from the bottom up.

## Default

Not defined. Token preemption in hierarchical project groups is top down.

## PORT

### Syntax

**PORT=*integer***

## Description

Defines the TCP listening port used by License Scheduler hosts, including candidate License Scheduler hosts. Specify any non-privileged port number.

## BLC\_HEARTBEAT\_FACTOR

### Syntax

**BLC\_HEARTBEAT\_FACTOR=*integer***

## Description

Enables bld to detect bldcollect failure. Defines the number of times that bld receives no response from a license collector daemon (bldcollect) before bld resets the values for that collector to zero. Each license usage reported to bld by the collector is treated as a heartbeat.

## Default

3

## Clusters section

### Description

Required. Lists the clusters that can use License Scheduler.

When configuring clusters for a WAN, the Clusters section of the master cluster must define its slave clusters.

### Clusters section structure

The Clusters section begins and ends with the lines `Begin Clusters` and `End Clusters`. The second line is the column heading, `CLUSTERS`. Subsequent lines list participating clusters, one name per line:

```
Begin Clusters
```

```
CLUSTERS
```

```
cluster1
```

```
cluster2
```

```
End Clusters
```

## CLUSTERS

Defines the name of each participating LSF cluster. Specify using one name per line.

## ServiceDomain section

### Description

Required. Defines License Scheduler service domains as groups of physical license server hosts that serve a specific network.

### ServiceDomain section structure

Define a section for each License Scheduler service domain.

This example shows the structure of the section:

```
Begin ServiceDomain
```

```
NAME=DesignCenterB
```

```
LIC_SERVERS=((1888@hostD)(1888@hostE))
```

```
LIC_COLLECTOR=CenterB
```

```
End ServiceDomain
```

### Parameters

- NAME
- LIC\_SERVERS
- LIC\_COLLECTOR
- LM\_STAT\_INTERVAL

### NAME

Defines the name of the service domain.

## LIC\_SERVERS

### Syntax

```
LIC_SERVERS=((host_name | port_number@host_name |( port_number@host_name
port_number@host_name port_number@host_name)) [ ...])
```

### Description

Defines the FlexNet license server hosts that make up the License Scheduler service domain. For each FlexNet license server host, specify the number of the port that FlexNet uses, then the at symbol (@), then the name of the host. If FlexNet uses the default port on a host, you can specify the host name without the port number. Put one set of parentheses around the list, and one more set of parentheses around each host, unless you have redundant servers (three hosts sharing one license file). If you have redundant servers, the parentheses enclose all three hosts.

### Examples

- One FlexNet license server host:  

```
LIC_SERVERS=(( 1700@hostA ))
```
- Multiple FlexNet license server hosts with unique license .dat files:  

```
LIC_SERVERS=(( ( 1700@hostA ) ( 1700@hostB ) ( 1700@hostC ) )
```
- Redundant FlexNet license server hosts sharing the same license .dat file:  

```
LIC_SERVERS=(( ( 1700@hostD 1700@hostE 1700@hostF )
```

## LIC\_COLLECTOR

### Syntax

```
LIC_COLLECTOR=license_collector_name
```

### Description

Optional. Defines a name for the license collector daemon (`bl collect`) to use in each service domain. `bl collect` collects license usage information from FlexNet and passes it to the License Scheduler daemon (`bl d`). It improves performance by allowing you to distribute license information queries on multiple hosts.

You can only specify one collector per service domain, but you can specify one collector to serve multiple service domains. Each time you run `bl collect`, you must specify the name of the collector for the service domain. You can use any name you want.

### Default

Undefined. The License Scheduler daemon uses one license collector daemon for the entire cluster.

## LM\_STAT\_INTERVAL

### Syntax

```
LM_STAT_INTERVAL=seconds
```

### Description

Defines a time interval between calls that License Scheduler makes to collect license usage information from FlexNet license management.

The value specified for a service domain overrides the global value defined in the Parameters section. Each service domain definition can specify a different value for this parameter.

## Default

Undefined: License Scheduler applies the global value.

# Feature section

## Description

Required. Defines license distribution policies.

## Feature section structure

Define a section for each feature managed by License Scheduler.

```

Begin Feature
NAME=vcs
FLEX_NAME=vcs
DISTRIBUTION=lanserver1 (Lp1 1 Lp2 4/6)
lanserver2 (Lp3 1 Lp4 10/8)
wanserver (Lp1 1 Lp2 1 Lp3 1 Lp4 1)
End Feature

```

## Parameters

- NAME
- FLEX\_NAME
- DISTRIBUTION
- ALLOCATION
- GROUP
- GROUP\_DISTRIBUTION
- LOCAL\_TO
- LS\_FEATURE\_PERCENTAGE
- NON\_SHARED\_DISTRIBUTION
- PREEMPT\_RESERVE
- SERVICE\_DOMAINS
- WORKLOAD\_DISTRIBUTION
- ENABLE\_DYNAMIC\_RUSAGE
- DYNAMIC
- LM\_REMOVE\_INTERVAL
- ENABLE\_MINJOB\_PREEMPTION
- ACCINUSE\_INCLUDES\_OWNERSHIP
- LS\_WAIT\_TO\_PREEMPT

## NAME

Required. Defines the token name—the name used by License Scheduler and LSF to identify the license feature.

Normally, license token names should be the same as the FlexNet Licensing feature names, as they represent the same license. However, LSF does not support names that start with a number, or names containing a dash or hyphen character (-), which may be used in the FlexNet Licensing feature name.

## FLEX\_NAME

Optional. Defines the feature name—the name used by FlexNet to identify the type of license. You only need to specify this parameter if the License Scheduler token name is not identical to the FlexNet feature name.

FLEX\_NAME allows the NAME parameter to be an alias of the FlexNet feature name. For feature names that start with a number or contain a dash (-), you must set both NAME and FLEX\_NAME, where FLEX\_NAME is the actual FlexNet Licensing feature name, and NAME is an arbitrary license token name you choose.

For example

```
Begin Feature
FLEX_NAME=201-AppZ
NAME=AppZ201
DISTRIBUTION=LanServer1(Lp1 1 Lp2 1)
End Feature
```

## DISTRIBUTION

### Syntax

**DISTRIBUTION**=[*service\_domain\_name*([*project\_name number\_shares*[/*number\_licenses\_owned*]] ... [default] )] ...

#### ***service\_domain\_name***

Specify a License Scheduler service domain (described in the ServiceDomain section) that distributes the licenses.

#### ***project\_name***

Specify a License Scheduler project (described in the Projects section) that is allowed to use the licenses.

#### ***number\_shares***

Specify a positive integer representing the number of shares assigned to the project.

The number of shares assigned to a project is only meaningful when you compare it to the number assigned to other projects, or to the total number assigned by the service domain. The total number of shares is the sum of the shares assigned to each project.

#### ***number\_licenses\_owned***

Optional. Specify a slash (/) and a positive integer representing the number of licenses that the project owns.

#### **default**

A reserved keyword that represents the default License Scheduler project if the job submission does not specify a project (bsub -Lp).

Default includes all projects that have not been defined in the PROJECTS section of `lsf.licensescheduler`. Jobs that belong to projects that are defined in `lsf.licensescheduler` do not get a share of the tokens when the project is not explicitly defined in the distribution.

## Description

Required if `GROUP_DISTRIBUTION` is not defined. Defines the distribution policies for the license. The name of each service domain is followed by its distribution policy, in parentheses. The distribution policy determines how the licenses available in each service domain are distributed among the clients.

The distribution policy is a space-separated list with each project name followed by its share assignment. The share assignment determines what fraction of available licenses is assigned to each project, in the event of competition between projects. Optionally, the share assignment is followed by a slash and the number of licenses owned by that project. License ownership enables a preemption policy. (In the event of competition between projects, projects that own licenses preempt jobs. Licenses are returned to the owner immediately.)

`GROUP_DISTRIBUTION` and `DISTRIBUTION` are mutually exclusive. If they are both defined in the same feature, the License Scheduler daemon returns an error and ignores this feature.

## Examples

```
DISTRIBUTION=wanserver (Lp1 1 Lp2 1 Lp3 1 Lp4 1)
```

In this example, the service domain named `wanserver` shares licenses equally among four License Scheduler projects. If all projects are competing for a total of eight licenses, each project is entitled to two licenses at all times. If all projects are competing for only two licenses in total, each project is entitled to a license half the time.

```
DISTRIBUTION=lanserver1 (Lp1 1 Lp2 2/6)
```

In this example, the service domain named `lanserver1` allows `Lp1` to use one third of the available licenses and `Lp2` can use two thirds of the licenses. However, `Lp2` is always entitled to six licenses, and can preempt another project to get the licenses immediately if they are needed. If the projects are competing for a total of 12 licenses, `Lp2` is entitled to eight licenses (six on demand, and two more as soon as they are free). If the projects are competing for only six licenses in total, `Lp2` is entitled to all of them, and `Lp1` can only use licenses when `Lp2` does not need them.

## ALLOCATION

### Syntax

```
ALLOCATION=project_name (cluster_name [number_shares] ... ) ...
```

***cluster\_name***

Specify LSF cluster names that licenses are to be allocated to.

***project\_name***

Specify a License Scheduler project (described in the PROJECTS section) that is allowed to use the licenses.

***number\_shares***

Specify a positive integer representing the number of shares assigned to the cluster.

The number of shares assigned to a cluster is only meaningful when you compare it to the number assigned to other clusters. The total number of shares is the sum of the shares assigned to each cluster.

## Description

Defines the allocation of license features across clusters and between LSF jobs and non-LSF jobs.

ALLOCATION ignores the global setting of the ENABLE\_INTERACTIVE parameter because ALLOCATION is configured for the license feature.

You can configure the allocation of license shares to:

- Change the share number between clusters for a feature
- Limit the scope of license usage and change the share number between LSF jobs and interactive tasks for a feature

---

### Tip:

To manage interactive (non-LSF) tasks in License Scheduler projects, you require the LSF Task Manager, `taskman`. The Task Manager utility is supported by License Scheduler. For more information about `taskman`, contact Platform.

---

## Default

Undefined. If ENABLE\_INTERACTIVE is not set, each cluster receives one share, and interactive tasks receive no shares.

Each example contains two clusters and 12 licenses of a specific feature.

## Example 1

ALLOCATION is not configured. The ENABLE\_INTERACTIVE parameter is not set.

```
Begin Parameters
```

```
...
```

```
ENABLE_INTERACTIVE=n
```

```
...
```

```
End Parameters
```

```
Begin Feature
```

```
NAME=Appl i cati onX
```

```
DI STRI BUTI ON=Li censeServer1 (Lp1 1)
```

```
End Feature
```

Six licenses are allocated to each cluster. No licenses are allocated to interactive tasks.

## Example 2

ALLOCATION is not configured. The ENABLE\_INTERACTIVE parameter is set.

```
Begin Parameters
```

```
...
```

```
ENABLE_INTERACTIVE=y
```

```
...
```

```
End Parameters
```

```
Begin Feature
```

```
NAME=ApplicationX
```

```
DISTRIBUTION=LicenceServer1 (Lp1 1)
```

```
End Feature
```

Four licenses are allocated to each cluster. Four licenses are allocated to interactive tasks.

## Example 3

In the following example, the ENABLE\_INTERACTIVE parameter does not affect the ALLOCATION configuration of the feature.

ALLOCATION is configured. The ENABLE\_INTERACTIVE parameter is set.

```
Begin Parameters
```

```
...
```

```
ENABLE_INTERACTIVE=y
```

```
...
```

```
End Parameters
```

```
Begin Feature
```

```
NAME=ApplicationY
```

```
DISTRIBUTION=LicenceServer1 (Lp2 1)
```

```
ALLOCATION=Lp2(cluster1 1 cluster2 0 interactive 1)
```

```
End Feature
```

The ENABLE\_INTERACTIVE setting is ignored. Licenses are shared equally between cluster1 and interactive tasks. Six licenses of ApplicationY are allocated to cluster1. Six licenses are allocated to interactive tasks.

## Example 4

In the following example, the ENABLE\_INTERACTIVE parameter does not affect the ALLOCATION configuration of the feature.

ALLOCATION is configured. The ENABLE\_INTERACTIVE parameter is not set.

```
Begin Parameters
```

```
...
```

```
ENABLE_INTERACTIVE=n
```

```
...
```

```
End Parameters
```

```
Begin Feature
```

```
NAME=ApplicationZ
```

```
DISTRIBUTION=LicenseServer1 (Lp1 1)
```

```
ALLOCATION=Lp1(cluster1 0 cluster2 1 interactive 2)
```

```
End Feature
```

The ENABLE\_INTERACTIVE setting is ignored. Four licenses of ApplicationZ are allocated to cluster2. Eight licenses are allocated to interactive tasks.

## GROUP

### Syntax

```
GROUP=[group_name{project_name...}] ...
```

#### ***group\_name***

Specify a name for a group of projects.

#### ***project\_name***

Specify a License Scheduler project (described in the PROJECTS section) that is allowed to use the licenses. The project must appear in the DISTRIBUTION.

A project should only belong to one group.

### Description

Optional. Defines groups of projects and specifies the name of each group. The groups defined here are used for group preemption and replace single projects with group projects.

This parameter is ignored if GROUP\_DISTRIBUTION is also defined.

## GROUP\_DISTRIBUTION

### Syntax

```
GROUP_DISTRIBUTION=top_level_hierarchy_name
```

#### ***top\_level\_hierarchy\_name***

Specify the name of the top level hierarchical group.

### Description

Required if DISTRIBUTION is not defined. Defines the name of the hierarchical group containing the distribution policy attached to this feature.

GROUP\_DISTRIBUTION and DISTRIBUTION are mutually exclusive. If they are both defined in the same feature, the License Scheduler daemon returns an error and ignores this feature.

If GROUP is also defined, it is ignored in favor of GROUP\_DISTRIBUTION.

## Example

The following example shows the GROUP\_DISTRIBUTION parameter hierarchical scheduling for the top-level hierarchical group named groups. The SERVICE\_DOMAINS parameter defines a list of service domains that provide tokens for the group.

```
Begin Feature
NAME = myjob2
GROUP_DISTRIBUTION = groups
SERVICE_DOMAINS = LanServer wanServer
End Feature
```

## LOCAL\_TO

### Syntax

**LOCAL\_TO**=*cluster\_name* | *location\_name*(*cluster\_name* [*cluster\_name* ...])

### Description

Configures token locality for the license feature. You must configure different feature sections for same feature based on their locality. By default, if LOCAL\_TO is not defined, the feature is available to all clients and is not restricted by geographical location. When LOCAL\_TO is configured, for a feature, License Scheduler treats license features served to different locations as different token names, and distributes the tokens to projects according the distribution and allocation policies for the feature.

LOCAL\_TO allows you to limit features from different service domains to specific clusters, so License Scheduler only grants tokens of a feature to jobs from clusters that are entitled to them.

For example, if your license servers restrict the serving of license tokens to specific geographical locations, use LOCAL\_TO to specify the locality of a license token if any feature cannot be shared across all the locations. This avoids having to define different distribution and allocation policies for different service domains, and allows hierarchical group configurations.

License Scheduler manages features with different localities are different resources. Use `blinfo` and `blstat` to see the different resource information for the features depending on their cluster locality.

License features with different localities must be defined in different feature sections. The same Service Domain can appear only once in the configuration for a given license feature.

A configuration like LOCAL\_TO=Site1(clusterA clusterB) configures the feature for more than one cluster.

A configuration like LOCAL\_TO=clusterA configures locality for only one cluster. This is the same as LOCAL\_TO=clusterA(clusterA).

Cluster names must be the names of clusters defined in the Clusters section of `lsf.licensescheduler`.

## Examples

```

Begin Feature
NAME = hspice
DISTRIBUTION = SD1 (Lp1 1 Lp2 1)
LOCAL_TO = siteUS(clusterA clusterB)
End Feature

Begin Feature
NAME = hspice
DISTRIBUTION = SD2 (Lp1 1 Lp2 1)
LOCAL_TO = clusterA
End Feature

Begin Feature
NAME = hspice
DISTRIBUTION = SD3 (Lp1 1 Lp2 1) SD4 (Lp1 1 Lp2 1)
End Feature

Or use the hierarchical group configuration (GROUP_DISTRIBUTION):
Begin Feature
NAME = hspice
GROUP_DISTRIBUTION = group1
SERVICE_DOMAINS = SD1
LOCAL_TO = clusterA
End Feature

Begin Feature
NAME = hspice
GROUP_DISTRIBUTION = group1
SERVICE_DOMAINS = SD2
LOCAL_TO = clusterB
End Feature

Begin Feature
NAME = hspice
GROUP_DISTRIBUTION = group1
SERVICE_DOMAINS = SD3 SD4
End Feature

```

## Default

Not defined. The feature is available to all clusters and taskman jobs, and is not restricted by cluster.

## LS\_FEATURE\_PERCENTAGE

### Syntax

```
LS_FEATURE_PERCENTAGE=Y | N
```

### Description

Configures license ownership in percentages instead of absolute numbers. When not combined with hierarchical projects, affects DISTRIBUTED and NON\_SHARED\_DISTRIBUTION values only. When using hierarchical projects, percentage is applied to OWNERSHIP, LIMITS, and NON\_SHARED values.

### Example 1

```
Begin Feature
LS_FEATURE_PERCENTAGE = Y
DISTRIBUTION = LanServer (p1 1 p2 1 p3 1/20)
...
End Feature
```

The service domain LanServer shares licenses equally among three License Scheduler projects. P3 is always entitled to 20% of the total licenses, and can preempt another project to get the licenses immediately if they are needed.

### Example 2

With LS\_FEATURE\_PERCENTAGE=Y in feature section and using hierarchical project groups:

```
Begin ProjectGroup
GROUP      SHARES    OWNERSHIP  LIMITS  NON_SHARED
(R (A p4)) (1 1)      ()         ()       ()
(A (B p3)) (1 1)      (- 10)     (- 20)   ()
(B (p1 p2)) (1 1)      (30 -)     ()       (- 5)
End ProjectGroup
```

Project p1 owns 30% of the total licenses, and project p3 owns 10% of total licenses. P3's LIMITS is 20% of total licenses, and p2's NON\_SHARED is 5%.

### Default

N (Ownership is not configured with percentages but with absolute numbers.)

## NON\_SHARED\_DISTRIBUTION

### Syntax

```
NON_SHARED_DISTRIBUTION=service_domain_name (project_name
number_non_shared_licenses] ... ) ...
```

#### ***service\_domain\_name***

Specify a License Scheduler service domain (described in the ServiceDomain section) that distributes the licenses.

#### ***project\_name***

Specify a License Scheduler project (described in the Projects section) that is allowed to use the licenses.

### ***number\_non\_shared\_licenses***

Specify a positive integer representing the number of non-shared licenses that the project owns.

## Description

Optional. Defines non-shared licenses. Non-shared licenses are not shared with other license projects. They are available only to one project.

Use `blinfo -a` to display `NON_SHARED_DISTRIBUTION` information.

## Example

```
Begin Feature
NAME=f1 # total 15 on LanServer and 15 on WanServer
FLEX_NAME=VCS- RUNTIME
DISTRIBUTION=LanServer(Lp1 4 Lp2 1) WanServer (Lp1 1 Lp2 1/3)
NON_SHARED_DISTRIBUTION=LanServer(Lp1 10) WanServer (Lp1 5 Lp2 3)
PREEMPT_RESERVE=Y
End Feature
```

In this example:

- 10 non-shared licenses are defined for the Lp1 project on LanServer
- 5 non-shared licenses are defined for the Lp1 project on WanServer
- 3 non-shared licenses are defined for the Lp2 project on WanServer

The remaining licenses are distributed as follows:

- LanServer: The remaining 5 (15-10=5) licenses on LanServer is distributed to the Lp1 and Lp2 projects with a 4:1 ratio.
- WanServer: The remaining 7 (15-5-3=7) licenses on WanServer is distributed to the Lp1 and Lp2 projects with a 1:1 ratio. If Lp2 uses fewer than 6 (3 privately owned+ 3 owned) licenses, then a job in the Lp2 can preempt Lp1 jobs.

## PREEMPT\_LSF

### Syntax

```
PREEMPT_LSF=Y
```

### Description

Optional. With the flex grid interface integration installed, enables on-demand preemption of LSF jobs for important non-managed workload. This guarantees that important non-managed jobs do not fail because of lack of licenses.

### Default

LSF workload is not preemptable.

## PREEMPT\_RESERVE

### Syntax

**PREEMPT\_RESERVE=Y**

### Description

Optional. Enables License Scheduler to preempt either licenses that are reserved or already in use by other projects. The number of jobs must be greater than the number of licenses owned.

### Default

Y. Reserved licenses are preemptable.

## SERVICE\_DOMAINS

### Syntax

**SERVICE\_DOMAINS=***service\_domain\_name* ...

***service\_domain\_name***

Specify the name of the service domain.

### Description

Required if GROUP\_DISTRIBUTION is defined. Specifies the service domains that provide tokens for this feature.

## WORKLOAD\_DISTRIBUTION

### Syntax

**WORKLOAD\_DISTRIBUTION=**[*service\_domain\_name*(**LSF** *lsf\_distribution* **NON\_LSF** *non\_lsf\_distribution*)] ...

***service\_domain\_name***

Specify a License Scheduler service domain (described in the ServiceDomain section) that distributes the licenses.

***lsf\_distribution***

Specify the share of licenses dedicated to LSF workloads. The share of licenses dedicated to LSF workloads is a ratio of *lsf\_distribution:non\_lsf\_distribution*.

***non\_lsf\_distribution***

Specify the share of licenses dedicated to non-LSF workloads. The share of licenses dedicated to non-LSF workloads is a ratio of *non\_lsf\_distribution:lsf\_distribution*.

### Description

Optional. Defines the distribution given to each LSF and non-LSF workload within the specified service domain.

Use `blinfo -a` to display WORKLOAD\_DISTRIBUTION configuration.

## Example

```

Begin Feature
NAME=ApplicationX
DISTRIBUTION=LicenceServer1(Lp1 1 Lp2 2)
WORKLOAD_DISTRIBUTION=LicenceServer1(LSF 8 NON_LSF 2)
End Feature

```

On the LicenceServer1 domain, the available licenses are dedicated in a ratio of 8:2 for LSF and non-LSF workloads. This means that 80% of the available licenses are dedicated to the LSF workload, and 20% of the available licenses are dedicated to the non-LSF workload.

If LicenceServer1 has a total of 80 licenses, this configuration indicates that 64 licenses are dedicated to the LSF workload, and 16 licenses are dedicated to the non-LSF workload.

## ENABLE\_DYNAMIC\_RUSAGE

### Syntax

```
ENABLE_DYNAMIC_RUSAGE=Y
```

### Description

Enforces license distribution policies for class-C license features.

When set, ENABLE\_DYNAMIC\_RUSAGE enables all class-C license checkouts to be considered managed checkout, instead of unmanaged (or OTHERS).

## DYNAMIC

### Syntax

```
DYNAMIC=Y
```

### Description

If you specify DYNAMIC=Y, you must specify a duration in an rusage resource requirement for the feature. This enables License Scheduler to treat the license as a dynamic resource and prevents License Scheduler from scheduling tokens for the feature when they are not available, or reserving license tokens when they should actually be free.

## LM\_REMOVE\_INTERVAL

### Syntax

```
LM_REMOVE_INTERVAL=seconds
```

### Description

Specifies the minimum time a job must have a license checked out before lremove can remove the license. lremove causes lmgrd and vendor daemons to close the TCP connection with the application. They then retry the license checkout.

The value specified for a feature overrides the global value defined in the Parameters section. Each feature definition can specify a different value for this parameter.

## Default

Undefined: License Scheduler applies the global value.

## ENABLE\_MINJOB\_PREEMPTION

### Syntax

**ENABLE\_MINJOB\_PREEMPTION=Y**

### Description

Minimizes the overall number of preempted jobs by enabling job list optimization. For example, for a job that requires 10 licenses, License Scheduler preempts one job that uses 10 or more licenses rather than 10 jobs that each use one license.

## Default

Undefined: License Scheduler does not optimize the job list when selecting jobs to preempt.

## ACCINUSE\_INCLUDES\_OWNERSHIP

### Syntax

**ACCINUSE\_INCLUDES\_OWNERSHIP=Y**

### Description

When defined, the value calculated for the accumulated INUSE amount includes the number of tokens owned. This is useful for projects that have a very high ownership set when considered against the total number of tokens available for LSF workload. Projects can be starved for tokens when the ownership is set too high and this parameter is not set.

When it is not set, the INUSE amount includes only the actual tokens in use plus any reserved tokens.

## Default

N, not enabled.

## LS\_WAIT\_TO\_PREEMPT

### Syntax

**LS\_WAIT\_TO\_PREEMPT=*seconds***

### Description

Defines the number of seconds that jobs must wait (time since it was dispatched) before it can be preempted. Applies to LSF and taskman jobs.

When LM\_REMOVE\_INTERVAL is also defined, the LM\_REMOVE\_INTERVAL value overrides the LS\_WAIT\_TO\_PREEMPT value.

## Default

0. The job can be preempted even if it was just dispatched.

## FeatureGroup section

### Description

Optional. Collects license features into groups. Put FeatureGroup sections after Feature sections in `lsf.licensescheduler`.

### FeatureGroup section structure

The FeatureGroup section begins and ends with the lines `Begin FeatureGroup` and `End FeatureGroup`. Feature group definition consists of a unique name and a list of features contained in the feature group.

### Example

```
Begin FeatureGroup
NAME = Synposys
FEATURE_LIST = ASTRO VCS_Runtime_Net Hsim Hspice
End FeatureGroup
Begin FeatureGroup
NAME = Cadence
FEATURE_LIST = Encounter NCSim NCVerilog
End FeatureGroup
```

### Parameters

- NAME
- FEATURE\_LIST

### NAME

Required. Defines the name of the feature group. The name must be unique.

### FEATURE\_LIST

Required. Lists the license features contained in the feature group. The feature names in `FEATURE_LIST` must already be defined in Feature sections. Feature names cannot be repeated in the `FEATURE_LIST` of one feature group. The `FEATURE_LIST` cannot be empty. Different feature groups can have the same features in their `FEATURE_LIST`.

## ProjectGroup section

### Description

Optional. Defines the hierarchical relationships of projects.

The hierarchical groups can have multiple levels of grouping. You can configure a tree-like scheduling policy, with the leaves being the license projects that jobs can belong to. Each project group in the tree has a set of values, including shares, limits, ownership and non-shared, or exclusive, licenses.

Use `blstat -G` to view the hierarchical dynamic license information.

Use `blinfo -G` to view the hierarchical configuration.

## ProjectGroup section structure

Define a section for each hierarchical group managed by License Scheduler.

The keywords `GROUP`, `SHARES`, `OWNERSHIP`, `LIMITS`, and `NON_SHARED` are required. The keywords `PRIORITY` and `DESCRIPTION` are optional. Empty brackets are allowed only for `OWNERSHIP`, `LIMITS`, and `PRIORITY`. `SHARES` must be specified.

```

Begin      ProjectGroup
GROUP SHARES OWNERSHIP LIMITS NON_SHARED PRIORITY
(root(A B C)) (1 1 1)          ()      ()      ()      (3 2 -)
(A (P1 D))    (1 1)           ()      ()      ()      (3 5)
(B (P4 P5))   (1 1)           ()      ()      ()      ()
(C (P6 P7 P8)) (1 1 1)         ()      ()      ()      (8 3 0)
(D (P2 P3))   (1 1)           ()      ()      ()      (2 1)
End ProjectGroup

```

## Parameters

- `GROUP`
- `SHARES`
- `OWNERSHIP`
- `LIMITS`
- `NON_SHARED`
- `PRIORITY`
- `DESCRIPTION`

## GROUP

Defines the project names in the hierarchical grouping and its relationships. Each entry specifies the name of the hierarchical group and its members.

For better readability, you should specify the projects in the order from the root to the leaves as in the example.

Specify the entry as follows:

*(group (member ...))*

## SHARES

Required. Defines the shares assigned to the hierarchical group member projects. Specify the share for each member, separated by spaces, in the same order as listed in the `GROUP` column.

## OWNERSHIP

Defines the level of ownership of the hierarchical group member projects. Specify the ownership for each member, separated by spaces, in the same order as listed in the `GROUP` column.

You can only define `OWNERSHIP` for hierarchical group member projects, not hierarchical groups. Do not define `OWNERSHIP` for the top level (root) project group. Ownership of a given internal node is the sum of the ownership of all child nodes it directly governs.

A dash (-) is equivalent to a zero, which means there are no owners of the projects. You can leave the parentheses empty () if desired.

## Valid values

A positive integer between the NON\_SHARED and LIMITS values defined for the specified hierarchical group.

- If defined as less than NON\_SHARED, OWNERSHIP is set to NON\_SHARED.
- If defined as greater than LIMITS, OWNERSHIP is set to LIMITS.

## LIMITS

Defines the maximum number of licenses that can be used at any one time by the hierarchical group member projects. Specify the maximum number of licenses for each member, separated by spaces, in the same order as listed in the GROUP column.

A dash (-) is equivalent to INFINIT\_INT, which means there is no maximum limit and the project group can use as many licenses as possible.

You can leave the parentheses empty () if desired.

## NON\_SHARED

Defines the number of licenses that the hierarchical group member projects use exclusively. Specify the number of licenses for each group or project, separated by spaces, in the same order as listed in the GROUP column.

A dash (-) is equivalent to a zero, which means there are no licenses that the hierarchical group member projects use exclusively.

Normally, the total number of non-shared licenses should be less than the total number of license tokens available. License tokens may not be available to project groups if the total non-shared licenses for all groups is greater than the number of shared tokens available.

For example, feature p4\_4 is configured as follows, with a total of 4 tokens:

```
Begin Feature
```

```
NAME =p4_4 # total token value is 4
```

```
GROUP_DISTRIBUTION=final
```

```
SERVICE_DOMAINS=LanServer
```

```
End Feature
```

The correct configuration is:

GROUP	SHARES	OWNERSHIP	LIMITS	NON_SHARED
(final (G2 G1))	(1 1)	()	()	(2 0)
(G1 (AP2 AP1))	(1 1)	()	()	(1 1)

## Valid values

Any positive integer up to the LIMITS value defined for the specified hierarchical group.

If defined as greater than LIMITS, NON\_SHARED is set to LIMITS.

## PRIORITY

Optional. Defines the priority assigned to the hierarchical group member projects. Specify the priority for each member, separated by spaces, in the same order as listed in the GROUP column.

"0" is the lowest priority, and a higher number specifies a higher priority. This column overrides the default behavior. Instead of preempting based on the accumulated `inuse` usage of each project, the projects are preempted according to the specified priority from lowest to highest.

By default, priorities are evaluated top down in the project group hierarchy. The priority of a given node is first decided by the priority of the parent groups. When two nodes have the same priority, priority is determined by the accumulated `inuse` usage of each project at the time the priorities are evaluated. Specify `LS_PREEMPT_PEER=Y` in the `Parameters` section to enable bottom-up license token preemption in hierarchical project group configuration.

A dash (-) is equivalent to a zero, which means there is no priority for the project. You can leave the parentheses empty () if desired.

Use `blinfo -G` to view hierarchical project group priority information.

## Priority of default project

If not explicitly configured, the default project has the priority of 0. You can override this value by explicitly configuring the default project in `Projects` section with the chosen priority value.

## DESCRIPTION

Optional. Description of the project group.

The text can include any characters, including white space. The text can be extended to multiple lines by ending the preceding line with a backslash (\). The maximum length for the text is 64 characters. When the `DESCRIPTION` column is not empty it should contain one entry for each project group member.

For example:

GROUP	SHARES	OWNERSHIP	LIMITS	NON_SHARED	DESCRIPTION
(R (A B))	(1 1)	()	()	(10 10)	()
(A (p1 p2))	(1 1)	(40 60)	()	()	("p1 desc. " "")
(B (p3 p4))	(1 1)	()	()	()	("p3 desc. " "p4 desc. ")

Use `blinfo -G` to view hierarchical project group descriptions.

## Projects section

### Description

Required. Lists the License Scheduler projects.

### Projects section structure

The `Projects` section begins and ends with the lines `Begin Projects` and `End Projects`. The second line consists of the required column heading `PROJECTS` and the optional column heading `PRIORITY`. Subsequent lines list participating projects, one name per line.

## Examples

The following example lists the projects without defining the priority:

```
Begin Projects
```

```
PROJECTS
```

```
Lp1
```

```
Lp2
```

```
Lp3
```

```
Lp4
```

```
...
```

```
End Projects
```

The following example lists the projects and defines the priority of each project:

```
Begin Projects
```

```
PROJECTS          PRI OR I T Y
```

```
Lp1                3
```

```
Lp2                4
```

```
Lp3                2
```

```
Lp4                1
```

```
default           0
```

```
...
```

```
End Projects
```

## Parameters

- PROJECTS
- PRIORITY
- DESCRIPTION

## PROJECTS

Defines the name of each participating project. Specify using one name per line.

## PRIORITY

Optional. Defines the priority for each project where “0” is the lowest priority, and the higher number specifies a higher priority. This column overrides the default behavior. Instead of preempting in order the projects are listed under PROJECTS based on the accumulated `inuse` usage of each project, the projects are preempted according to the specified priority from lowest to highest.

When 2 projects have the same priority number configured, the first project listed has higher priority, like LSF queues.

Use `blinfo -Lp` to view project priority information.

## Priority of default project

If not explicitly configured, the default project has the priority of 0. You can override this value by explicitly configuring the default project in Projects section with the chosen priority value.

## DESCRIPTION

Optional. Description of the project.

The text can include any characters, including white space. The text can be extended to multiple lines by ending the preceding line with a backslash (\). The maximum length for the text is 64 characters.

Use `bl info -Lp` to view the project description.

## Automatic time-based configuration

Variable configuration is used to automatically change License Scheduler license token distribution policy configuration based on time windows. You define automatic configuration changes in `lsf.licensescheduler` by using if-else constructs and time expressions in the Feature section. After you change the file, check the configuration with the `bl admin ckconfig` command, and restart License Scheduler the cluster with the `bl admin reconfig` command.

The expressions are evaluated by License Scheduler every 10 minutes based on the `bl d` start time. When an expression evaluates true, License Scheduler dynamically changes the configuration based on the associated configuration statements, restarting `bl d` automatically.

## Example

```
Begin Feature
NAME = f1
#if time(5:16:30-1:8:30 20:00-8:30)
DISTRIBUTION=Lan(P1 2/5 P2 1)
#elif time(3:8:30-3:18:30)
DISTRIBUTION=Lan(P3 1)
#else
DISTRIBUTION=Lan(P1 1 P2 2/5)
#endif
End Feature
```

# lsf.shared

The `lsf.shared` file contains common definitions that are shared by all load sharing clusters defined by `lsf.cluster.cluster_name` files. This includes lists of cluster names, host types, host models, the special resources available, and external load indices, including indices required to submit jobs using JSDL files.

This file is installed by default in the directory defined by `LSF_CONFDIR`.

## Changing lsf.shared configuration

After making any changes to `lsf.shared`, run the following commands:

- `lsadmin reconfig` to reconfigure LIM
- `lsadmin mbdrestart` to restart `mbatchd`

## Cluster section

(Required) Lists the cluster names recognized by the LSF system

### Cluster section structure

The first line must contain the mandatory keyword `ClusterName`. The other keyword is optional.

The first line must contain the mandatory keyword `ClusterName` and the keyword `Servers` in a `MultiCluster` environment.

Each subsequent line defines one cluster.

### Example Cluster section

```
Begin Cluster
ClusterName Servers
cluster1      hostA
cluster2      hostB
End Cluster
```

## ClusterName

Defines all cluster names recognized by the LSF system.

All cluster names referenced anywhere in the LSF system must be defined here. The file names of cluster-specific configuration files must end with the associated cluster name.

By default, if `MultiCluster` is installed, all clusters listed in this section participate in the same `MultiCluster` environment. However, individual clusters can restrict their `MultiCluster` participation by specifying a subset of clusters at the cluster level (`lsf.cluster.cluster_name RemoteClusters` section).

## Servers

`MultiCluster` only. List of hosts in this cluster that LIMs in remote clusters can connect to and obtain information from.

For other clusters to work with this cluster, one of these hosts must be running `mbatchd`.

## HostType section

(Required) Lists the valid host types in the cluster. All hosts that can run the same binary executable are in the same host type.

---

### Caution:

If you remove NTX86, NTX64, or NTIA64 from the HostType section, the functionality of `lspasswd.exe` is affected. The `lspasswd` command registers a password for a Windows user account.

---

## HostType section structure

The first line consists of the mandatory keyword `TYPENAME`.

Subsequent lines name valid host types.

## Example HostType section

```
Begin HostType
```

```
TYPENAME
```

```
SOL64
```

```
SOLSPARC
```

```
LI NUX86LI NUXPPC
```

```
LI NUX64
```

```
NTX86
```

```
NTX64
```

```
NTIA64
```

```
End HostType
```

## TYPENAME

Host type names are usually based on a combination of the hardware name and operating system. If your site already has a system for naming host types, you can use the same names for LSF.

## HostModel section

(Required) Lists models of machines and gives the relative CPU scaling factor for each model. All hosts of the same relative speed are assigned the same host model.

LSF uses the relative CPU scaling factor to normalize the CPU load indices so that jobs are more likely to be sent to faster hosts. The CPU factor affects the calculation of job execution time limits and accounting. Using large or inaccurate values for the CPU factor can cause confusing results when CPU time limits or accounting are used.

## HostModel section structure

The first line consists of the mandatory keywords `MODELNAME`, `CPUFACTOR`, and `ARCHITECTURE`.

Subsequent lines define a model and its CPU factor.

## Example HostModel section

```

Begin HostModel
MODELNAME    CPUFACTOR    ARCHITECTURE
PC400        13.0         (i86pc_400 i686_400)
PC450        13.2         (i86pc_450 i686_450)
Sparc5F      3.0          (SUNWSPARCstation5_170_sparc)
Sparc20      4.7          (SUNWSPARCstation20_151_sparc)
Ultra5S      10.3         (SUNWUltra5_270_sparcv9 SUNWUltra510_270_sparcv9)
End HostModel

```

## ARCHITECTURE

(Reserved for system use only) Indicates automatically detected host models that correspond to the model names.

## CPUFACTOR

Though it is not required, you would typically assign a CPU factor of 1.0 to the slowest machine model in your system and higher numbers for the others. For example, for a machine model that executes at twice the speed of your slowest model, a factor of 2.0 should be assigned.

## MODELNAME

Generally, you need to identify the distinct host types in your system, such as MIPS and SPARC first, and then the machine models within each, such as SparcIPC, Sparc1, Sparc2, and Sparc10.

## About automatically detected host models and types

When you first install LSF, you do not necessarily need to assign models and types to hosts in `lsf.cluster.cluster_name`. If you do not assign models and types to hosts in `lsf.cluster.cluster_name`, LIM automatically detects the model and type for the host.

If you have versions earlier than LSF 4.0, you may have host models and types already assigned to hosts. You can take advantage of automatic detection of host model and type also.

Automatic detection of host model and type is useful because you no longer need to make changes in the configuration files when you upgrade the operating system or hardware of a host and reconfigure the cluster. LSF will automatically detect the change.

## Mapping to CPU factors

Automatically detected models are mapped to the short model names in `lsf.shared` in the ARCHITECTURE column. Model strings in the ARCHITECTURE column are only used for mapping to the short model names.

Example `lsf.shared` file:

```

Begin HostModel
MODELNAME    CPUFACTOR    ARCHITECTURE
SparcU5      5.0          (SUNWUltra510_270_sparcv9)
PC486        2.0          (i486_33 i486_66)
PowerPC      3.0          (PowerPC12 PowerPC16 PowerPC31)
End HostModel

```

If an automatically detected host model cannot be matched with the short model name, it is matched to the best partial match and a warning message is generated.

If a host model cannot be detected or is not supported, it is assigned the DEFAULT model name and an error message is generated.

## Naming convention

Models that are automatically detected are named according to the following convention:

```
hardware_platform [ _processor_speed[ _processor_type] ]
```

where:

- *hardware\_platform* is the only mandatory component
- *processor\_speed* is the optional clock speed and is used to differentiate computers within a single platform
- *processor\_type* is the optional processor manufacturer used to differentiate processors with the same speed
- Underscores ( `_` ) between *hardware\_platform*, *processor\_speed*, *processor\_type* are mandatory.

## Resource section

Optional. Defines resources (must be done by the LSF administrator).

### Resource section structure

The first line consists of the keywords. RESOURCENAME and DESCRIPTION are mandatory. The other keywords are optional. Subsequent lines define resources.

### Example Resource section

```
Begin Resource
```

RESOURCENAME	TYPE	INTERVAL	INCREASING	CONSUMABLE	DESCRIPTION	# Keywords
patchrev	Numeric	()	Y	()	(Patch revision)	
specman	Numeric	()	N	()	(Specman)	
switch	Numeric	()	Y	N	(Network Switch)	
rack	String	()	()	()	(Server room rack)	
owner	String	()	()	()	(Owner of the host)	
elimres	Numeric	10	Y	()	(elim generated index)	
ostype	String	()	()	()	(Operating system and version)	
lmhostid	String	()	()	()	(FlexLM s lmhostid)	
limversion	String	()	()	()	(Version of LIM binary)	

```
End Resource
```

## RESOURCENAME

The name you assign to the new resource. An arbitrary character string.

- A resource name cannot begin with a number.
- A resource name cannot contain any of the following characters:

```
: . ( ) [ + - * / ! & | < > @ =
```

- A resource name cannot be any of the following reserved names:

```
cpu cpuf io logins ls idle maxmem maxswp maxtmp type model status it
```

```
mem ncpus define_ncpus_cores define_ncpus_procs
```

```
define_ncpus_threads ndisks pg r15m r15s r1m swap swp tmp ut
```

- To avoid conflict with `inf` and `nan` keywords in 3rd-party libraries, resource names should not begin with `inf` or `nan` (upper case or lower case). Resource requirement strings, such as `-R "infra"` or `-R "nano"` will cause an error. Use `-R "defined(infxx)"` or `-R "defined(nanxx)"`, to specify these resource names.
- Resource names are case sensitive
- Resource names can be up to 39 characters in length
- For Solaris machines, the keyword `int` is reserved and cannot be used.

## TYPE

The type of resource:

- Boolean—Resources that have a value of 1 on hosts that have the resource and 0 otherwise.
- Numeric—Resources that take numerical values, such as all the load indices, number of processors on a host, or host CPU factor.
- String—Resources that take string values, such as host type, host model, host status.

### Default

If `TYPE` is not given, the default type is Boolean.

## INTERVAL

Optional. Applies to dynamic resources only.

Defines the time interval (in seconds) at which the resource is sampled by the ELIM.

If `INTERVAL` is defined for a numeric resource, it becomes an external load index.

### Default

If `INTERVAL` is not given, the resource is considered static.

## INCREASING

Applies to numeric resources only.

If a larger value means greater load, `INCREASING` should be defined as `Y`. If a smaller value means greater load, `INCREASING` should be defined as `N`.

## CONSUMABLE

Explicitly control if a resource is consumable. Applies to static or dynamic numeric resources.

Static and dynamic numeric resources can be specified as consumable. `CONSUMABLE` is optional. The defaults for the consumable attribute are:

- Built-in indicies:
  - The following are consumable: `r15s`, `r1m`, `r15m`, `ut`, `pg`, `io`, `ls`, `it`, `tmp`, `swp`, `mem`.
  - All other built-in static resources are not consumable. (e.g., `ncpus`, `ndisks`, `maxmem`, `maxswp`, `maxtmp`, `cpuf`, `type`, `model`, `status`, `rexpri`, `server`, `hname`).

- External shared resources:
  - All numeric resources are consumable.
  - String and boolean resources are not consumable.

You should only specify consumable resources in the `rusage` section of a resource requirement string. Non-consumable resources are ignored in `rusage` sections.

A non-consumable resource should not be releasable. Non-consumable numeric resource should be able to be used in `order`, `select` and `same` sections of a resource requirement string.

When `LSF_STRICT_RESREQ=Y` in `lsf.conf`, LSF rejects resource requirement strings where an `rusage` section contains a non-consumable resource.

## DESCRIPTION

Brief description of the resource.

The information defined here will be returned by the `ls_info()` API call or printed out by the `lsinfo` command as an explanation of the meaning of the resource.

## RELEASE

Applies to numeric shared resources only, such as floating licenses.

Controls whether LSF releases the resource when a job using the resource is suspended. When a job using a shared resource is suspended, the resource is held or released by the job depending on the configuration of this parameter.

Specify `N` to hold the resource, or specify `Y` to release the resource.

## Default

Y

# lsf.sudoers

## About lsf.sudoers

The `lsf.sudoers` file is an optional file to configure security mechanisms. It is not installed by default.

You use `lsf.sudoers` to set the parameter `LSF_EAUTH_KEY` to configure a key for `eauth` to encrypt and decrypt user authentication data.

On UNIX, you also use `lsf.sudoers` to grant permission to users other than root to perform certain operations as root in LSF, or as a specified user.

These operations include:

- LSF daemon startup/shutdown
- User ID for LSF authentication
- User ID for LSF pre- and post-execution commands.
- User ID for external LSF executables

If `lsf.sudoers` does not exist, only root can perform these operations in LSF on UNIX.

On UNIX, this file is located in `/etc`.

There is one `lsf.sudoers` file per host.

On Windows, this file is located in the directory specified by the parameter `LSF_SECUREDIR` in `lsf.conf`.

## Changing lsf.sudoers configuration

After making any changes to `lsf.sudoers`, run `badm in reconfi g` to reload the configuration files.

## lsf.sudoers on UNIX

In LSF, certain operations such as daemon startup can only be performed by root. The `lsf.sudoers` file grants root privileges to specific users or user groups to perform these operations.

### Location

`lsf.sudoers` must be located in `/etc` on each host.

### Permissions

`lsf.sudoers` must have permission 600 and be readable and writable only by root.

## lsf.sudoers on Windows

The `lsf.sudoers` file is shared over an NTFS network, not duplicated on every Windows host.

By default, LSF installs `lsf.sudoers` in the `%SYSTEMROOT%` directory.

The location of `lsf.sudoers` on Windows must be specified by `LSF_SECUREDIR` in `lsf.conf`. You must configure the `LSF_SECUREDIR` parameter in `lsf.conf` if using `lsf.sudoers` on Windows.

## Windows permissions

---

### Restriction:

The owner of `lsf.sudoers` on Windows be **Administrators**. If not, `eauth` may not work.

---

The permissions on `lsf.sudoers` for Windows are:

#### Workgroup Environment

- Local Admins (W)
- Everyone (R)

#### Domain Environment

- Domain Admins (W)
- Everyone (R)

## File format

The format of `lsf.sudoers` is very similar to that of `lsf.conf`.

Each entry can have one of the following forms:

- NAME=VALUE
- NAME=
- NAME= "STRING1 STRING2 ..."

The equal sign = must follow each NAME even if no value follows and there should be no space beside the equal sign.

NAME describes an authorized operation.

VALUE is a single string or multiple strings separated by spaces and enclosed in quotation marks.

Lines starting with a pound sign (#) are comments and are ignored. Do not use `#if` as this is reserved syntax for time-based configuration.

## Example lsf.sudoers File

```
LSB_PRE_POST_EXEC_USER=user100
LSF_STARTUP_PATH=/usr/share/lsf/etc
LSF_STARTUP_USERS="user1 user10 user55"
```

## Creating and modifying lsf.sudoers

You can create and modify `lsf.sudoers` with a text editor.

After you modify `lsf.sudoers`, you must run `badm in hrest art al l` to restart all `sbatchds` in the cluster with the updated configuration.

## Parameters

- LSB\_PRE\_POST\_EXEC\_USER
- LSF\_EAUTH\_KEY
- LSF\_EAUTH\_USER
- LSF\_EEXEC\_USER
- LSF\_EGO\_ADMIN\_PASSWD
- LSF\_EGO\_ADMIN\_USER
- LSF\_LOAD\_PLUGINS

- LSF\_STARTUP\_PATH
- LSF\_STARTUP\_USERS

## LSB\_PRE\_POST\_EXEC\_USER

### Syntax

**LSB\_PRE\_POST\_EXEC\_USER**=*user\_name*

### Description

Specifies the UNIX user account under which pre- and post-execution commands run. This parameter applies only to pre- and post-execution commands configured at the queue level; by default, pre-execution and post-execution commands defined at the application or job level run under the account of the user who submits the job.

You can specify only one user account. If the pre-execution or post-execution commands perform privileged operations that require `root` permissions on UNIX hosts, specify a value of `root`.

If you configure this parameter as `root`, the `LD_PRELOAD` and `LD_LIBRARY_PATH` variables are removed from the pre-execution, post-execution, and `eexec` environments for security purposes.

### Default

Not defined. Pre-execution and post-execution commands run under the user account of the user who submits the job.

## LSF\_EAUTH\_KEY

### Syntax

**LSF\_EAUTH\_KEY**=*key*

### Description

Applies to UNIX, Windows, and mixed UNIX/Windows clusters.

Specifies the key that `eauth` uses to encrypt and decrypt user authentication data. Defining this parameter enables increased security at your site. The key must contain at least six characters and must use only printable characters.

For UNIX, you must edit the `lsf.sudoers` file on all hosts within the cluster and specify the same encryption key. For Windows, you must edit the shared `lsf.sudoers` file.

### Default

Not defined. The `eauth` executable encrypts and decrypts authentication data using an internal key.

## LSF\_EAUTH\_USER

### Syntax

**LSF\_EAUTH\_USER**=*user\_name*

### Description

UNIX only.

Specifies the UNIX user account under which the external authentication executable `eauth` runs.

## Default

Not defined. The `eauth` executable runs under the account of the primary LSF administrator.

## LSF\_EEXEC\_USER

### Syntax

**LSF\_EEXEC\_USER**=*user\_name*

### Description

UNIX only.

Specifies the UNIX user account under which the external executable `eexec` runs.

## Default

Not defined. The `eexec` executable runs under root or the account of the user who submitted the job.

## LSF\_EGO\_ADMIN\_PASSWD

### Syntax

**LSF\_EGO\_ADMIN\_PASSWD**=*password*

### Description

When the EGO Service Controller (EGOSC) is configured to control LSF daemons, enables UNIX and Windows users to bypass the additional login required to start `res` and `sbatchd`. Bypassing the EGO administrator login enables the use of scripts to automate system startup.

Specify the Admin EGO cluster administrator password as clear text. You must also define the `LSF_EGO_ADMIN_USER` parameter.

## Default

Not defined. With EGOSC daemon control enabled, the `lsadmin` and `badmin` startup subcommands invoke the `egosh user logon` command to prompt for the Admin EGO cluster administrator credentials.

## LSF\_EGO\_ADMIN\_USER

### Syntax

**LSF\_EGO\_ADMIN\_USER**=Admin

### Description

When the EGO Service Controller (EGOSC) is configured to control LSF daemons, enables UNIX and Windows users to bypass the additional login required to start `res` and `sbatchd`. Bypassing the EGO administrator login enables the use of scripts to automate system startup.

Specify the Admin EGO cluster administrator account. You must also define the `LSF_EGO_ADMIN_PASSWD` parameter.

## Default

Not defined. With EGOSC daemon control enabled, the `lsadm in` and `badmi n` startup subcommands invoke the `egosh user logon` command to prompt for the Admin EGO cluster administrator credentials.

## LSF\_LOAD\_PLUGINS

### Syntax

`LSF_LOAD_PLUGINS=y | Y`

### Description

If defined, LSF loads plugins from `LSB_LSBDIR`. Used for Kerberos authentication and to enable the LSF `cpuset` plugin for SGI.

## Default

Not defined. LSF does not load plugins.

## LSF\_STARTUP\_PATH

### Syntax

`LSF_STARTUP_PATH=path`

### Description

UNIX only. Enables the LSF daemon startup control feature when `LSF_STARTUP_USERS` is also defined. Define both parameters when you want to allow users other than `root` to start LSF daemons.

Specifies the absolute path name of the directory in which the LSF daemon binary files (`lim`, `res`, `sbat chd`, and `mbat chd`) are installed. LSF daemons are usually installed in the path specified by `LSF_SERVERDIR` defined in the `cs hrc. lsf`, `profi le. lsf` or `lsf. conf` files.

---

#### Important:

For security reasons, you should move the LSF daemon binary files to a directory other than `LSF_SERVERDIR` or `LSF_BINDIR`. The user accounts specified by `LSF_STARTUP_USERS` can start any binary in the `LSF_STARTUP_PATH`.

---

## Default

Not defined. Only the `root` user account can start LSF daemons.

## LSF\_STARTUP\_USERS

### Syntax

`LSF_STARTUP_USERS=all_admins | "user_name..."`

### Description

UNIX only. Enables the LSF daemon startup control feature when `LSF_STARTUP_PATH` is also defined. Define both parameters when you want to allow users other than `root` to start LSF daemons. On Windows, the Platform `services adm in` group is equivalent to `LSF_STARTUP_USERS`.

On UNIX hosts, by default only `root` can start LSF daemons. To manually start LSF daemons, a user runs the commands `lsadm n` and `badmi n`, which have been installed as `setuid root`. `LSF_STARTUP_USERS` specifies a list of user accounts that can successfully run the commands `lsadm n` and `badmi n` to start LSF daemons.

#### **all\_admins**

- Allows all UNIX users defined as LSF administrators in the file `lsf.cluster.cluster_name` to start LSF daemons as `root` by running the `lsadm n` and `badmi n` commands.
- Not recommended due to the security risk of a non-root LSF administrator adding to the list of administrators in the `lsf.cluster.cluster_name` file.
- Not required for Windows hosts because all users with membership in the `Platform services_admin` group can start LSF daemons.

#### **"user\_name..."**

- Allows the specified user accounts to start LSF daemons by running the `lsadm n` and `badmi n` commands.
- Separate multiple user names with a space.
- For a single user, do not use quotation marks.

## Default

Not defined. Only the `root` user account can start LSF daemons.

## See also

`LSF_STARTUP_PATH`

## lsf.task

Users should not have to specify a resource requirement each time they submit a job. LSF supports the concept of a task list. This chapter describes the files used to configure task lists: `lsf.task`, `lsf.task.cluster_name`, and `.lsftask`.

## Changing task list configuration

After making any changes to the task list files, run the following commands:

- `lsadmin reconfig` to reconfigure LIM
- `badmin reconfig` to reload the configuration files

## About task lists

A task list is a list in LSF that keeps track of the default resource requirements for different applications and task eligibility for remote execution.

The term task refers to an application name. With a task list defined, LSF automatically supplies the resource requirement of the job whenever users submit a job unless one is explicitly specified at job submission.

LSF takes the job's command name as the task name and uses that name to find the matching resource requirement for the job from the task list. If a task does not have an entry in the task list, LSF assumes the default resource requirement; that is, a host that has the same host type as the submission host will be chosen to run the job.

An application listed in a task file is considered for load sharing by its placement in either the local tasks or remote tasks list.

- A local task is typically an application or command that it does not make sense to run remotely such as `ls`.
- A remote task is an application or command that can be run on another machine in the LSF cluster. The `compress` command is an example of a remote task.

Some applications require resources other than the default. LSF can store resource requirements for specific applications in remote task list files, so that LSF automatically chooses candidate hosts that have the correct resources available.

For frequently used commands and software packages, the LSF administrator can set up cluster-wide resource requirements that apply to all users in the cluster.

Users can modify and add to these requirements by setting up additional resource requirements that apply only to their own jobs.

## Cluster-wide resource requirements

The resource requirements of applications are stored in the remote task list file.

LSF automatically picks up a job's default resource requirement string from the remote task list files, unless you explicitly override the default by specifying the resource requirement string on the command line.

## User-level resource requirements

You may have applications that you need to control yourself. Perhaps your administrator did not set them up for load sharing for all users, or you need a non-standard setup. You can use LSF commands to find

out resource names available in your system, and tell LSF about the needs of your applications. LSF stores the resource requirements for you from then on.

You can specify resource requirements when tasks are added to the user's remote task list. If the task to be added is already in the list, its resource requirements are replaced.

```
lsrtasks + myjob/swap>=100 && cpu
```

This adds `myjob` to the remote tasks list with its resource requirements.

## Task files

There are 3 task list files that can affect a job:

- `lsf.task` — system-wide defaults apply to all LSF users, even across multiple clusters if MultiCluster is installed
- `lsf.task.cluster_name` — cluster-wide defaults apply to all users in the cluster
- `$HOME/.lsftask` — user-level defaults apply to a single user. This file lists applications to be added to or removed from the default system lists for your jobs. Resource requirements specified in this file override those in the system lists.

The clusterwide task file is used to augment the systemwide file. The user's task file is used to augment the systemwide and clusterwide task files.

LSF combines the systemwide, clusterwide, and user-specific task lists for each user's view of the task list. In cases of conflicts, such as different resource requirements specified for the same task in different lists, the clusterwide list overrides the systemwide list, and the user-specific list overrides both.

### LSF\_CONFDIR/lsf.task

Systemwide task list applies to all clusters and all users.

This file is used in a MultiCluster environment.

### LSF\_CONFDIR/lsf.task.cluster\_name

Clusterwide task list applies to all users in the same cluster.

### \$HOME/.lsftask

User task list, one per user, applies only to the specific user. This file is automatically created in the user's home directory whenever a user first updates his task lists using the `lsrtasks` or `lsltasks` commands. For details about task eligibility lists, see the `ls_task(3)` API reference man page.

## Permissions

Only the LSF administrator can modify the systemwide task list (`lsf.task`) and the clusterwide task list (`lsf.task.cluster_name`).

A user can modify his own task list (`.lsftask`) with the `lsrtasks` and `lsltasks` commands.

## Format of task files

Each file consists of two sections, Local Tasks and RemoteTasks. For example:

```

Begin LocalTasks
ps
hostname
uname
crontab
End LocalTasks
Begin RemoteTasks
+ "newjob/mem>25"
+ "verilog/select[type==any && swp>100]"
make/cpu
nroff/-
End RemoteTasks

```

Tasks are listed one per line. Each line in a section consists of a task name, and, for the RemoteTasks section, an optional resource requirement string separated by a slash (/).

A plus sign (+) or a minus sign (-) can optionally precede each entry. If no + or - is specified, + is assumed.

A + before a task name means adding a new entry (if non-existent) or replacing an entry (if already existent) in the task list. A - before a task name means removing an entry from the application's task lists if it was already created by reading higher level task files.

### LocalTasks section

The section starts with `Begin LocalTasks` and ends with `End LocalTasks`.

This section lists tasks that are not eligible for remote execution, either because they are trivial tasks or because they need resources on the local host.

### RemoteTasks section

The section starts with `Begin RemoteTasks` and ends with `End RemoteTasks`.

This section lists tasks that are eligible for remote execution. You can associate resource requirements with each task name.

See *Administering Platform LSF* for information about resource requirement strings. If the resource requirement string is not specified for a remote task, the default is `"select[type==local] order [r15s:pg]"`.

# setup.config

## About setup.config

The `setup.config` file contains options for Platform License Scheduler installation and configuration for systems without Platform LSF. You only need to edit this file if you are installing License Scheduler as a standalone product without LSF.

## Template location

A template `setup.config` is included in the License Scheduler installation script tar file and is located in the directory created when you uncompress and extract the installation script tar file. Edit the file and uncomment the options you want in the template file. Replace the example values with your own settings to specify the options for your new License Scheduler installation.

---

### Important:

The sample values in the `setup.config` template file are examples only. They are not default installation values.

---

After the License Scheduler installation, the `setup.config` containing the options you specified is located in `LS_TOP/8.0/install/`.

## Format

Each entry in `setup.config` has the form:

```
NAME="STRING1 STRING2 ..."
```

The equal sign `=` must follow each `NAME` even if no value follows and there should be no spaces around the equal sign.

A value that contains multiple strings separated by spaces must be enclosed in quotation marks.

Blank lines and lines starting with a pound sign (`#`) are ignored.

## Parameters

- `LS_ADMIN`
- `LS_HOSTS`
- `LS_LICENSE_FILE`
- `LS_LMSTAT_PATH`
- `LS_TOP`

## LS\_ADMIN

### Syntax

```
LS_ADMIN="user_name [user_name ...]"
```

### Description

Lists the License Scheduler administrators. The first user account name in the list is the primary License Scheduler administrator.

The primary License Scheduler administrator account is typically named `lsadmin`.

---

**Caution:**

You should *not* configure the root account as the primary License Scheduler administrator.

---

## Valid Values

User accounts for License Scheduler administrators must exist on all hosts using License Scheduler prior to installation.

## Example

```
LS_ADMI NS="lsadmin user1 user2"
```

## Default

The user running the License Scheduler installation script.

## LS\_HOSTS

### Syntax

```
LS_HOSTS="host_name [host_name ... ]"
```

### Description

Defines a list of hosts that are candidates to become License Scheduler master hosts. Provide at least one host from which the License Scheduler daemon will run.

## Valid Values

Any valid License Scheduler host name.

## Example

```
LS_HOSTS="host_name1 host_name2"
```

## Default

The local host in which the License Scheduler installation script is running.

## LS\_LICENSE\_FILE

### Syntax

```
LS_LICENSE_FILE="/path/license_file"
```

### Description

Defines the full path to, and name of the License Scheduler license file.

## Valid Values

Any valid file name and directory path.

## Example

```
LS_LICENSE_FILE="/usr/share/l s/conf/l i cense. dat "
```

setup.config

## Default

`$LS_TOP/conf/license.dat`

# LS\_LMSTAT\_PATH

## Syntax

```
LS_LMSTAT_PATH="/path"
```

## Description

Defines the full path to the `lmstat` program. License Scheduler uses `lmstat` to gather the FlexNet license information for scheduling. This path does not include the name of the `lmstat` program itself.

## Example

```
LS_LMSTAT_PATH="/usr/bin"
```

## Default

The installation script attempts to find a working copy of `lmstat` on the current system. If it is unsuccessful, the path is set as blank (`""`).

# LS\_TOP

## Syntax

```
LS_TOP="/path"
```

## Description

Defines the full path to the top level License Scheduler installation directory.

## Valid Values

Must be an absolute path to a shared directory that is accessible to all hosts using License Scheduler. Cannot be the root directory (`/`).

## Recommended Value

The file system containing `LS_TOP` must have enough disk space for all host types (approximately 300 MB per host type).

## Example

```
LS_TOP="/usr/share/lis"
```

## Default

None — required variable

# slave.config

## About slave.config

Dynamically added LSF hosts that will not be master candidates are *slave hosts*. Each dynamic slave host has its own LSF binaries and local `lsf.conf` and shell environment scripts (`cshrc.lsf` and `profile.lsf`). You must install LSF on each slave host.

The `slave.config` file contains options for installing and configuring a slave host that can be dynamically added or removed.

Use `lsfinstall -s -f slave.config` to install LSF using the options specified in `slave.config`.

## Template location

A template `slave.config` is located in the installation script directory created when you extract the LSF installation script tar file. Edit the file and uncomment the options you want in the template file. Replace the example values with your own settings to specify the options for your new LSF installation.

---

### Important:

The sample values in the `slave.config` template file are examples only. They are not default installation values.

---

## Format

Each entry in `slave.config` has the form:

```
NAME="STRING1 STRING2 ..."
```

The equal sign = must follow each NAME even if no value follows and there should be no spaces around the equal sign.

A value that contains multiple strings separated by spaces must be enclosed in quotation marks.

Blank lines and lines starting with a pound sign (#) are ignored.

## Parameters

- EGO\_DAEMON\_CONTROL
- ENABLE\_EGO
- EP\_BACKUP
- LSF\_ADMINS
- LSF\_LIM\_PORT
- LSF\_SERVER\_HOSTS
- LSF\_TARDIR
- LSF\_LOCAL\_RESOURCES
- LSF\_TOP

## EGO\_DAEMON\_CONTROL

### Syntax

```
EGO_DAEMON_CONTROL="Y" | "N"
```

## Description

Enables Platform EGO to control LSF `res` and `sbatchd`. Set the value to "Y" if you want EGO Service Controller to start `res` and `sbatchd`, and restart if they fail.

All hosts in the cluster must use the same value for this parameter (this means the value of `EGO_DAEMON_CONTROL` in this file must be the same as the specification for `EGO_DAEMON_CONTROL` in `install.config`).

To avoid conflicts, leave this parameter undefined if you use a script to start up LSF daemons.

---

### Note:

If you specify `EGO_ENABLE="N"`, this parameter is ignored.

---

## Example

```
EGO_DAEMON_CONTROL="N"
```

## Default

N (`res` and `sbatchd` are started manually)

# ENABLE\_EGO

## Syntax

```
ENABLE_EGO="Y" | "N"
```

## Description

Enables Platform EGO functionality in the LSF cluster.

`ENABLE_EGO="Y"` causes `lsfi nstall` to uncomment `LSF_EGO_ENVDIR` and sets `LSF_ENABLE_EGO="Y"` in `lsf.conf`.

`ENABLE_EGO="N"` causes `lsfi nstall` to comment out `LSF_EGO_ENVDIR` and sets `LSF_ENABLE_EGO="N"` in `lsf.conf`.

Set the value to "Y" if you want to take advantage of the following LSF features that depend on EGO:

- LSF daemon control by EGO Service Controller
- EGO-enabled SLA scheduling

## Default

N (EGO is disabled in the LSF cluster)

# EP\_BACKUP

## Syntax

```
EP_BACKUP="Y" | "N"
```

## Description

Enables backup and rollback for enhancement packs. Set the value to "N" to disable backups when installing enhancement packs (you will not be able to roll back to the previous patch level after installing an EP, but you will still be able to roll back any fixes installed on the new EP).

You may disable backups to speed up install time, to save disk space, or because you have your own methods to back up the cluster.

## Default

Y (backup and rollback are fully enabled)

# LSF\_ADMINS

## Syntax

```
LSF_ADMINS="user_name [ user_name ... ]"
```

## Description

Required. List of LSF administrators.

The first user account name in the list is the primary LSF administrator. It cannot be the root user account.

Typically this account is named `lsfadmin`. It owns the LSF configuration files and log files for job events. It also has permission to reconfigure LSF and to control batch jobs submitted by other users. It typically does not have authority to start LSF daemons. Usually, only `root` has permission to start LSF daemons.

All the LSF administrator accounts must exist on all hosts in the cluster before you install LSF. Secondary LSF administrators are optional.

## Valid Values

Existing user accounts

## Example

```
LSF_ADMINS="lsfadmin user1 user2"
```

## Default

None—required variable

# LSF\_LIM\_PORT

## Syntax

```
LSF_LIM_PORT="port_number"
```

## Description

TCP service port for slave host.

Use the same port number as `LSF_LIM_PORT` in `lsf.conf` on the master host.

## Default

7869

# LSF\_SERVER\_HOSTS

## Syntax

```
LSF_SERVER_HOSTS="host_name [host_name ...]"
```

## Description

Required for non-shared slave host installation. This parameter defines a list of hosts that can provide host and load information to client hosts. If you do not define this parameter, clients will contact the master LIM for host and load information. List of LSF server hosts in the cluster to be contacted.

Recommended for large clusters to decrease the load on the master LIM. Do not specify the master host in the list. Client commands will query the LIMs on the LSF\_SERVER\_HOSTS, which off-loads traffic from the master LIM.

Define this parameter to ensure that commands execute successfully when no LIM is running on the local host, or when the local LIM has just started.

You should include the list of hosts defined in LSF\_MASTER\_LIST in `lsf.conf`; specify the primary master host last. For example:

```
LSF_MASTER_LIST="lsfmaster hostE"
```

```
LSF_SERVER_HOSTS="hostB hostC hostD hostE lsfmaster"
```

Specify a list of host names two ways:

- Host names separated by spaces
- Name of a file containing a list of host names, one host per line.

## Valid Values

Any valid LSF host name

## Examples

List of host names:

```
LSF_SERVER_HOSTS="hosta hostb hostc hostd"
```

Host list file:

```
LSF_SERVER_HOSTS=:lsf_server_hosts
```

The file `lsf_server_hosts` contains a list of hosts:

```
hosta hostb hostc hostd
```

## Default

None

# LSF\_TARDIR

## Syntax

```
LSF_TARDIR="!path"
```

## Description

Full path to the directory containing the LSF distribution tar files.

## Example

```
LSF_TARDIR="/usr/local/lsf_distribution"
```

## Default

The parent directory of the current working directory. For example, if `lsfinstall` is running under `usr/share/lsf_distribution/lsfinstall` the `LSF_TARDIR` default value is `usr/share/lsf_distribution`.

# LSF\_LOCAL\_RESOURCES

## Syntax

```
LSF_LOCAL_RESOURCES="resource ..."
```

## Description

Defines instances of local resources residing on the slave host.

- For numeric resources, define name-value pairs:  

```
"[resourcemap value*resource_name]"
```
- For Boolean resources, define the resource name in the form:  

```
"[resource resource_name]"
```

When the slave host calls the master host to add itself, it also reports its local resources. The local resources to be added must be defined in `lsf.shared`.

If the same resource is already defined in `lsf.shared` as `default` or `all`, it cannot be added as a local resource. The shared resource overrides the local one.

---

### Tip:

`LSF_LOCAL_RESOURCES` is usually set in the `slave.config` file during installation. If `LSF_LOCAL_RESOURCES` are already defined in a local `lsf.conf` on the slave host, `lsfinstall` does not add resources you define in `LSF_LOCAL_RESOURCES` in `slave.config`. You should not have duplicate `LSF_LOCAL_RESOURCES` entries in `lsf.conf`. If local resources are defined more than once, only the last definition is valid.

---

### Important:

Resources must already be mapped to hosts in the ResourceMap section of `lsf.cluster.cluster_name`. If the ResourceMap section does not exist, local resources are not added.

---

## Example

```
LSF_LOCAL_RESOURCES="[resourcemap 1*verilog] [resource linux]"
```

## Default

None

# LSF\_TOP

## Syntax

```
LSF_TOP="/path"
```

## Description

Required. Full path to the top-level LSF installation directory.

---

**Important:**

You must use the same path for every slave host you install.

---

## Valid value

The path to LSF\_TOP cannot be the root directory (/).

## Example

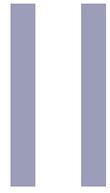
```
LSF_TOP="/usr/local/lsf"
```

## Default

None—required variable

P A R T

---



# Environment Variables



# Environment variables set for job execution

LSF transfers most environment variables between submission and execution hosts.

Environment variables related to file names and job spooling directories support paths that contain up to 4094 characters for UNIX and Linux, or up to 255 characters for Windows.

Environment variables related to command names and job names can contain up to 4094 characters for UNIX and Linux, or up to 255 characters for Windows.

In addition to environment variables inherited from the user environment, LSF also sets several other environment variables for batch jobs:

- **LSB\_ERRORFILE:** Name of the error file specified with a `bsub -e`.
- **LSB\_JOBID:** Job ID assigned by LSF.
- **LSB\_JOBINDEX:** Index of the job that belongs to a job array.
- **LSB\_CHKPNT\_DIR:** This variable is set each time a checkpointed job is submitted. The value of the variable is `chkpnt_dir/job_Id`, a subdirectory of the checkpoint directory that is specified when the job is submitted. The subdirectory is identified by the job ID of the submitted job.
- **LSB\_HOSTS:** The list of hosts that are used to run the batch job. For sequential jobs, this is only one host name. For parallel jobs, this includes multiple host names.
- **LSB\_RESIZABLE:** Indicates that a job is resizable or auto-resizable.
- **LSB\_QUEUE:** The name of the queue the job is dispatched from.
- **LSB\_JOBNAME:** Name of the job.
- **LSB\_RESTART:** Set to 'Y' if the job is a restarted job or if the job has been migrated. Otherwise this variable is not defined.
- **LSB\_EXIT\_PRE\_ABORT:** Set to an integer value representing an exit status. A pre-execution command should exit with this value if it wants the job to be aborted instead of requeued or executed.
- **LSB\_EXIT\_REQUEUE:** Set to the `REQUEUE_EXIT_VALUES` parameter of the queue. This variable is not defined if `REQUEUE_EXIT_VALUES` is not configured for the queue.
- **LSB\_INTERACTIVE:** Set to 'Y' if the job is submitted with the `-I` option. Otherwise, it is not defined.
- **LS\_JOBPID:** Set to the process ID of the job.
- **LS\_SUBCWD:** This is the directory on the submission when the job was submitted. This is different from `PWD` only if the directory is not shared across machines or when the execution account is different from the submission account as a result of account mapping.
- **LSB\_BIND\_JOB:** Set to the value of binding option. But when the binding option is `USER`, `LSB_BIND_JOB` is set to the real binding decision of end user.

---

## Note:

If the binding option is `Y`, `LSB_BIND_JOB` is set to `BALANCE`. If the binding option is `N`, `LSB_BIND_JOB` is set to `NONE`.

- **LSB\_BIND\_CPU\_LIST:** Set to the actual CPU list used when the job is sequential job and single host parallel job.

If the job is a multi-host parallel job, `LSB_BIND_CPU_LIST` is set to the value in submission environment variable `$LSB_USER_BIND_CPU_LIST`. If there is no such submission environment variable in user's environment, `LSB_BIND_CPU_LIST` is set to an empty string.

# Environment variables for resize notification command

All environment variables that are set for a job are also set when a job is resized.

The following (additional) environment variables apply only to the resize notification command environment (when using resizable jobs):

- **LSB\_RESIZE\_NOTIFY\_OK**: A notification command should exit with this variable if the allocation resize notification command succeeds.

LSF updates the job allocation to reflect the new allocation.

- **LSB\_RESIZE\_NOTIFY\_FAIL**: A notification command should exit with this variable if the allocation resize notification command fails.

For an allocation grow event, LSF schedules the pending allocation request.

For an allocation shrink event, LSF fails the release request.

- **LSB\_RESIZE\_EVENT** = grow | shrink: Indicates why the notification command was called. Grow means add more resources to an existing allocation. Shrink means remove some resources from existing allocation.
- **LSB\_RESIZE\_HOSTS** = hostA numA hostB numB ... hostZ numZ: Lists the additional slots for a grow event, or the released slots for a shrink event.

# Environment variables for session scheduler (ssched)

By default, all environment variables that are set as part of the session are available in each task's execution environment.

## Variables for the execution host of each task

The following environment variables are reset according to the execution host of each task:

- EGO\_SERVERDIR
- LSB\_TRAPSIGS
- LSF\_SERVERDIR
- HOSTTYPE
- LSB\_HOSTS
- LSF\_BINDIR
- EGO\_BINDIR
- PWD
- HOME
- LSB\_ERRORFILE
- LSB\_OUTPUTFILE
- TMPDIR
- LSF\_LIBDIR
- EGO\_LIBDIR
- LSB\_MCPU\_HOSTS
- PATH (prepend LSF\_BINDIR)
- LD\_LIBRARY\_PATH (prepend LSF\_LIBDIR and EGO\_LIBDIR)

## Environment variables NOT available in the task environment

- LSB\_JOBRES\_PID
- LSB\_EEXEC\_REAL\_UID
- LS\_EXEC\_T
- LSB\_INTERACTIVE
- LSB\_CHKFILENAME
- SPOOLDIR
- LSB\_ACCT\_FILE
- LSB\_EEXEC\_REAL\_GID
- LSB\_CHKPNT\_DIR
- LSB\_CHKPNT\_PERIOD
- LSB\_JOB\_STARTER
- LSB\_EXIT\_REQUEUE
- LSB\_DJOB\_RU\_INTERVAL
- LSB\_DJOB\_HB\_INTERVAL
- LSB\_DJOB\_HOSTFILE

Environment variables for session scheduler (ssched)

- LSB\_JOBEXIT\_INFO
- LSB\_JOBPEND
- LSB\_EXECHOSTS

## Environment variables corresponding to the session job

- LSB\_JOBID
- LSB\_JOBINDEX
- LSB\_JOBINDEX\_STEP
- LSB\_JOBINDEX\_END
- LSB\_JOBPID
- LSB\_JOBNAME
- LSB\_JOBFILENAME

## Environment variables set individually for each task

- LSB\_TASKID—The current task ID
- LSB\_TASKINDEX—The current task index

# Environment variable reference

BSUB_BLOCK	BSUB_CHK_RESREQ	
BSUB_QUIET	BSUB_QUIET2	
BSUB_STDERR	CLEARCASE_DRIVE	CLEARCASE_MOUNTDIR
CLEARCASE_ROOT	ELIM_ABORT_VALUE	LM_LICENSE_FILE
LS_EXEC_T	LS_JOBPID	LS_LICENSE_SERVER_ <i>feature</i>
LS_SUBCWD		LSB_CHKPNT_DIR
LSB_DEBUG	LSB_DEBUG_CMD	
LSB_DEBUG_MBD	LSB_DEBUG_NQS	LSB_DEBUG_SBD
LSB_DEBUG_SCH	LSB_DEFAULT_JOBGROUP	LSB_DEFAULTPROJECT
LSB_DEFAULTQUEUE	LSB_DJOB_COMMFAIL_ACTION	LSB_DJOB_ENV_SCRIPT
LSB_DJOB_NUMPROC	LSB_ECHKPNT_METHOD	LSB_ECHKPNT_METHOD_DIR
LSB_ECHKPNT_KEEP_OUTPUT	LSB_ERESTART_USRCMD	LSB_EXEC_RUSAGE
LSB_EXECHOSTS	LSB_EXIT_IF_CWD_NOTEXIST	LSB_EXIT_PRE_ABORT
LSB_EXIT_REQUEUE	LSB_FRAMES	LSB_HOSTS
LSB_INTERACTIVE	LSB_JOB_INCLUDE_POSTPROC	LSB_JOBEXIT_INFO
LSB_JOBEXIT_STAT	LSB_JOBFILENAME	LSB_JOBGROUP
LSB_JOBID	LSB_JOBINDEX	LSB_JOBINDEX_STEP
LSB_JOBNAME	LSB_JOBPEND	LSB_JOBPGIDS
LSB_JOBPIIDS	LSB_MAILSIZE	LSB_MCPU_HOSTS
LSB_NQS_PORT	LSB_NTRIES	LSB_OLD_JOBID
LSB_OUTPUT_TARGETFAILED	LSB_QUEUE	LSB_REMOTEINDEX
LSB_REMOTEJID	LSB_RESIZABLE	LSB_RESIZE_NOTIFY_OK
LSB_RESIZE_NOTIFY_FAIL	LSB_RESTART_PGID	LSB_RESTART
LSB_RESTART_PID	LSB_RTASK_GONE_ACTION	
LSB_SUB_CLUSTER	LSB_SUB_COMMAND_LINE	LSB_SUB_EXTSCHED_PARAM
LSB_SUB_JOB_ACTION_WARNING_TIME	LSB_SUB_JOB_WARNING_ACTION	
LSB_SUB_PARM_FILE	LSB_SUCCESS_EXIT_VALUES	LSB_SUSP_REASONS
LSB_SUSP_SUBREASONS	LSB_UNIXGROUP	LSB_USER_BIND_CPU_LIST

LSB_USER_BIND_JOB	LSF_CMD_LOGDIR	LSF_DEBUG_CMD
LSF_DEBUG_LIM	LSF_DEBUG_RES	LSF_EAUTH_AUX_DATA
LSF_EAUTH_AUX_PASS	LSF_EAUTH_CLIENT	LSF_EAUTH_SERVER
LSF_EAUTH_UID	LSF_EXECUTE_DOMAIN	LSF_INTERACTIVE_STDERR
LSF_INVOKE_CMD	LSF_JOB_STARTER	LSF_LD_PRELOAD
LSF_LD_LIBRARY_PATH	LSF_LIM_API_NTRIES	LSF_LIM_DEBUG
LSF_LOGDIR	LSF_MASTER	LSF_NIOS_DEBUG
LSF_NIOS_ERR_LOGDIR		
LSF_NIOS_DIE_CMD	LSF_NIOS_IGNORE_SIGWINDO W	LSF_NIOS_PEND_TIMEOUT
LSF_NIOS_PORT_RANGE	LSF_RESOURCES	LSF_TS_LOGON_TIME
LSF_USE_HOSTEQUIV	LSF_USER_DOMAIN	

## BSUB\_BLOCK

### Description

If set, tells NIOS that it is running in batch mode.

### Default

Not defined

### Notes

If you submit a job with the -K option of bsub, which is synchronous execution, then BSUB\_BLOCK is set. Synchronous execution means you have to wait for the job to finish before you can continue.

### Where defined

Set internally

### See also

The -K option of bsub

## BSUB\_CHK\_RESREQ

### Syntax

**BSUB\_CHK\_RESREQ**=*any\_value*

### Description

When BSUB\_CHK\_RESREQ is set, bsub checks the syntax of the resource requirement selection string without actually submitting the job for scheduling and dispatch. Use BSUB\_CHK\_RESREQ to check the compatibility of your existing resource requirement select strings against the stricter syntax enabled by

LSF\_STRICT\_RESREQ=y in `lsf.conf`. LSF\_STRICT\_RESREQ does not need to be set to check the resource requirement selection string syntax.

`bsub` only checks the select section of the resource requirement. Other sections in the resource requirement string are not checked.

## Default

Not defined

## Where defined

From the command line

## Example

```
BSUB_CHK_RESREQ=1
```

# BSUB\_QUIET

## Syntax

**BSUB\_QUIET**=*any\_value*

## Description

Controls the printing of information about job submissions. If set, `bsub` will not print any information about job submission. For example, it will not print `<Job is submitted to default queue <normal>, nor <Waiting for dispatch>`.

## Default

Not defined

## Where defined

From the command line

## Example

```
BSUB_QUIET=1
```

# BSUB\_QUIET2

## Syntax

**BSUB\_QUIET2**=*any\_value*

## Description

Suppresses the printing of information about job completion when a job is submitted with the `bsub -K` option.

If set, `bsub` will not print information about job completion to `stdout`. For example, when this variable is set, the message `<<Job is finished>>` will not be written to `stdout`.

If `BSUB_QUIET` and `BSUB_QUIET2` are both set, no job messages will be printed to `stdout`.

## Default

Not defined

## Where defined

From the command line

## Example

```
BSUB_QUEUE2=1
```

# BSUB\_STDERR

## Syntax

```
BSUB_STDERR=y
```

## Description

Redirects LSF messages for bsub to stderr.

By default, when this parameter is not set, LSF messages for bsub are printed to stdout.

When this parameter is set, LSF messages for bsub are redirected to stderr.

## Default

Not defined

## Where defined

From the command line on UNIX. For example, in csh:

```
setenv BSUB_STDERR Y
```

From the Control Panel on Windows, as an environment variable

# CLEARCASE\_DRIVE

## Syntax

```
CLEARCASE_DRIVE=drive_letter.
```

## Description

Optional, Windows only.

Defines the virtual drive letter for a Rational ClearCase view to the drive. This is useful if you wish to map a Rational ClearCase view to a virtual drive as an alias.

If this letter is unavailable, Windows attempts to map to another drive. Therefore, CLEARCASE\_DRIVE only defines the default drive letter to which the Rational ClearCase view is mapped, not the final selected drive letter. However, the PATH value is automatically updated to the final drive letter if it is different from CLEARCASE\_DRIVE.

## Notes:

CLEARCASE\_DRIVE is case insensitive.

## Where defined

From the command line

## Example

```
CLEARCASE_DRIVE=F:
```

```
CLEARCASE_DRIVE=f:
```

## See also

CLEARCASE\_MOUNTDIR, CLEARCASE\_ROOT

# CLEARCASE\_MOUNTDIR

## Syntax

```
CLEARCASE_MOUNTDIR=path
```

## Description

Optional.

Defines the Rational ClearCase mounting directory.

## Default

```
/vobs
```

## Notes:

CLEARCASE\_MOUNTDIR is used if any of the following conditions apply:

- A job is submitted from a UNIX environment but run in a Windows host.
- The Rational ClearCase mounting directory is not the default */vobs*

## Where defined

From the command line

## Example

```
CLEARCASE_MOUNTDIR=/myvobs
```

## See also

CLEARCASE\_DRIVE, CLEARCASE\_ROOT

# CLEARCASE\_ROOT

## Syntax

```
CLEARCASE_ROOT=path
```

## Description

The path to the Rational ClearCase view.

In Windows, this path must define an absolute path starting with the default ClearCase drive and ending with the view name without an ending backslash (\).

## Notes

CLEARCASE\_ROOT must be defined if you want to submit a batch job from a ClearCase view.

For interactive jobs, use `bsub -I` to submit the job.

## Where defined

In the job starter, or from the command line

## Example

In UNIX:

```
CLEARCASE_ROOT=/view/myview
```

In Windows:

```
CLEARCASE_ROOT=F:\myview
```

## See also

CLEARCASE\_DRIVE, CLEARCASE\_MOUNTDIR, LSF\_JOB\_STARTER

# ELIM\_ABORT\_VALUE

## Syntax

**ELIM\_ABORT\_VALUE**

## Description

Used when writing an `elim` executable to test whether the `elim` should run on a particular host. If the host does not have or share any of the resources listed in the environment variable `LSF_RESOURCES`, your `elim` should exit with `SELIM_ABORT_VALUE`.

When the MELIM finds an `elim` that exited with `ELIM_ABORT_VALUE`, the MELIM marks the `elim` and does not restart it on that host.

## Where defined

Set by the master `elim` (MELIM) on the host when the MELIM invokes the `elim` executable

# LM\_LICENSE\_FILE

## Syntax

**LM\_LICENSE\_FILE**=*file\_name*

## Description

The path to where the license file is found. The file name is the name of the license file.

## Default

`/usr/share/flexlm/licenses/license.dat`

## Notes

A FlexNet variable read by the `lmgrd` daemon.

## Where defined

From the command line

## See also

`LSF_LICENSE_FILE` in `lsf.conf`

# LS\_EXEC\_T

## Syntax

`LS_EXEC_T=START | END | CHPNT | JOB_CONTROLS`

## Description

Indicates execution type for a job. `LS_EXEC_T` is set to:

- `START` or `END` for a job when the job begins executing or when it completes execution
- `CHPNT` when the job is checkpointed
- `JOB_CONTROLS` when a control action is initiated

## Where defined

Set by `sbatchd` during job execution

# LS\_JOBPID

## Description

The process ID of the job.

## Where defined

During job execution, `sbatchd` sets `LS_JOBPID` to be the same as the process ID assigned by the operating system.

# LS\_LICENSE\_SERVER\_feature

## Syntax

`LS_LICENSE_SERVER_feature="domain.server.num_available ..."`

*server* is of the format `port@host`

## Description

The license server information provided to the job. The purpose of this environment variable is to provide license server information to the job.

## Where defined

During the license job execution, `sbat chd` sets `LS_LICENSE_SERVER_feature` to be the same as the license server information defined in the job's `rusage` string. This is only used by the job and logged in the `mbat chd log` file if `DEBUG1` and `LC_LICSCHEM` are defined in `lsf.conf`.

## LS\_SUBCWD

### Description

The current working directory (`cwd`) of the submission host where the remote task command was executed.

The current working directory can be up to 4094 characters long for UNIX and Linux or up to 255 characters for Windows.

### How set

1. LSF looks for the `PWD` environment variable. If it finds it, sets `LS_SUBCWD` to `PWD`.
2. If the `PWD` environment variable does not exist, LSF looks for the `CWD` environment variable. If it finds `CWD`, sets `LS_SUBCWD` to `CWD`.
3. If the `CWD` environment variable does not exist, LSF calls the `getwd()` system function to retrieve the current working directory path name. LSF sets `LS_SUBCWD` to the value that is returned.

## Where defined

Set by `sbat chd`

## LSB\_CHKPNT\_DIR

### Syntax

```
LSB_CHKPNT_DIR=checkpoint_dir/job_ID
```

### Description

The directory containing files related to the submitted checkpointable job.

### Valid values

The value of `checkpoint_dir` is the directory you specified through the `-k` option of `bsub` when submitting the checkpointable job.

The value of `job_ID` is the job ID of the checkpointable job.

## Where defined

Set by LSF, based on the directory you specified when submitting a checkpointable job with the `-k` option of `bsub`.

## LSB\_DEBUG

This parameter can be set from the command line or from `lsf.conf`. See `LSB_DEBUG` in `lsf.conf`.

## LSB\_DEBUG\_CMD

This parameter can be set from the command line or from `lsf.conf`. See `LSB_DEBUG_CMD` in `lsf.conf`.

## LSB\_DEBUG\_MBD

This parameter can be set from the command line with `badm in mbddebug` or from `lsf.conf`. See `LSB_DEBUG_MBD` in `lsf.conf`.

## LSB\_DEBUG\_NQS

This parameter can be set from the command line or from `lsf.conf`. See `LSB_DEBUG_NQS` in `lsf.conf`.

## LSB\_DEBUG\_SBD

This parameter can be set from the command line with `badm in sbddebug` or from `lsf.conf`. See `LSB_DEBUG_SBD` in `lsf.conf`.

## LSB\_DEBUG\_SCH

This parameter can be set from the command line or from `lsf.conf`. See `LSB_DEBUG_SCH` in `lsf.conf`.

## LSB\_DEFAULT\_JOBGROUP

### Syntax

**LSB\_DEFAULT\_JOBGROUP**=*job\_group\_name*

### Description

The name of the default job group.

When you submit a job to LSF without explicitly specifying a job group, LSF associates the job with the specified job group. `LSB_DEFAULT_JOBGROUP` overrides the setting of `DEFAULT_JOBGROUP` in `lsb.params`. The `bsub -g job_group_name` option overrides both `LSB_DEFAULT_JOBGROUP` and `DEFAULT_JOBGROUP`.

If you submit a job without the `-g` option of `bsub`, but you defined `LSB_DEFAULT_JOBGROUP`, then the job belongs to the job group specified in `LSB_DEFAULT_JOBGROUP`.

Job group names must follow this format:

- Job group names must start with a slash character (/). For example, `LSB_DEFAULT_JOBGROUP=/A/B/C` is correct, but `LSB_DEFAULT_JOBGROUP=A/B/C` is not correct.
- Job group names cannot end with a slash character (/). For example, `LSB_DEFAULT_JOBGROUP=/A/` is not correct.
- Job group names cannot contain more than one slash character (/) in a row. For example, job group names like `LSB_DEFAULT_JOBGROUP=/A//B` or `LSB_DEFAULT_JOBGROUP=A///B` are not correct.
- Job group names cannot contain spaces. For example, `LSB_DEFAULT_JOBGROUP=/A/B C/D` is not correct.

- Project names and user names used for macro substitution with %p and %u cannot start or end with slash character (/).
- Project names and user names used for macro substitution with %p and %u cannot contain spaces or more than one slash character (/) in a row.
- Project names or user names containing slash character (/) will create separate job groups. For example, if the project name is `canada/projects`, `LSB_DEFAULT_JOBGROUP=/%p` results in a job group hierarchy `/canada/projects`.

## Where defined

From the command line

## Example

```
LSB_DEFAULT_JOBGROUP=/canada/projects
```

## Default

Not defined

## See also

DEFAULT\_JOBGROUP in `lsb.params`, the `-g` option of `bsub`

# LSB\_DEFAULTPROJECT

## Syntax

```
LSB_DEFAULTPROJECT=project_name
```

## Description

The name of the project to which resources consumed by a job will be charged.

## Default

Not defined

## Notes

Project names can be up to 59 characters long.

If the LSF administrator defines a default project in the `lsb.params` configuration file, the system uses this as the default project. You can change the default project by setting `LSB_DEFAULTPROJECT` or by specifying a project name with the `-P` option of `bsub`.

If you submit a job without the `-P` option of `bsub`, but you defined `LSB_DEFAULTPROJECT`, then the job belongs to the project specified in `LSB_DEFAULTPROJECT`.

If you submit a job with the `-P` option of `bsub`, the job belongs to the project specified through the `-P` option.

## Where defined

From the command line, or through the `-P` option of `bsub`

## Example

```
LSB_DEFAULTPROJECT=engi neeri ng
```

## See also

DEFAULT\_PROJECT in `lsb.params`, the `-P` option of `bsub`

# LSB\_DEFAULTQUEUE

## Syntax

```
LSB_DEFAULTQUEUE=queue_name
```

## Description

Defines the default LSF queue.

## Default

`mbatchd` decides which is the default queue. You can override the default by defining `LSB_DEFAULTQUEUE`.

## Notes

If the LSF administrator defines a default queue in the `lsb.params` configuration file, then the system uses this as the default queue. Provided you have permission, you can change the default queue by setting `LSB_DEFAULTQUEUE` to a valid queue (see `bqueues` for a list of valid queues).

## Where defined

From the command line

## See also

DEFAULT\_QUEUE in `lsb.params`

# LSB\_DJOB\_NUMPROC

## Syntax

```
LSB_DJOB_NUMPROC=num
```

## Description

The number of processors (slots) allocated to the job.

## Default

Not defined

## Where defined

Set by `sbatchd` before starting a job on the execution host.

## See Also

LSB\_MCPU\_HOSTS

## LSB\_ECHKPNT\_METHOD

This parameter can be set as an environment variable and/or in `lsf.conf`. See `LSB_ECHKPNT_METHOD` in `lsf.conf`.

## LSB\_ECHKPNT\_METHOD\_DIR

This parameter can be set as an environment variable and/or in `lsf.conf`. See `LSB_ECHKPNT_METHOD_DIR` in `lsf.conf`.

## LSB\_ECHKPNT\_KEEP\_OUTPUT

This parameter can be set as an environment variable and/or in `lsf.conf`. See `LSB_ECHKPNT_KEEP_OUTPUT` in `lsf.conf`.

## LSB\_ERESTART\_USRCMD

### Syntax

**LSB\_ERESTART\_USRCMD**=*command*

### Description

Original command used to start the job.

This environment variable is set by `erestart` to pass the job's original start command to a custom `erestart` method `erestart.method_name`. The value of this variable is extracted from the job file of the checkpointed job.

If a job starter is defined for the queue to which the job was submitted, the job starter is also included in `LSB_ERESTART_USRCMD`. For example, if the job starter is `/bin/sh -c "%USRCMD"` in `lsb.queues`, and the job name is `myapp -d`, `LSB_ERESTART_USRCMD` will be set to:

```
/bin/sh -c "myapp -d"
```

### Where defined

Set by `erestart` as an environment variable before a job is restarted

### See also

`LSB_ECHKPNT_METHOD`, `erestart`, `echkpnt`

## LSB\_EXEC\_RUSAGE

### Syntax

**LSB\_EXEC\_RUSAGE**="*resource\_name1 resource\_value1 resource\_name2 resource\_value2...*"

### Description

Indicates which `rusage` string is satisfied to permit the job to run. This environment variable is necessary because the OR (`|`) operator specifies alternative `rusage` strings for running jobs.

## Valid values

`resource_value1, resource_value2,...` refer to the resource values on `resource_name1, resource_name2,...` respectively.

## Default

Not defined

## Where defined

Set by LSF after reserving a resource for the job.

# LSB\_EXECHOSTS

## Description

A list of hosts on which a batch job will run.

## Where defined

Set by `sbatchd`

## Product

MultiCluster

# LSB\_EXIT\_IF\_CWD\_NOTEXIST

## Syntax

`LSB_EXIT_IF_CWD_NOTEXIST=Y | y | N | n`

## Description

Indicates that the job will exit if the current working directory specified by `bsub -cwd` or `bmod -cwd` is not accessible on the execution host.

## Default

Not defined

## Where defined

From the command line

# LSB\_EXIT\_PRE\_ABORT

## Description

The queue-level or job-level `pre_exec_command` can exit with this value if the job is to be aborted instead of being queued or executed

## Where defined

Set by `sbatchd`

## See also

See `PRE_EXEC` in `l sb. queues`, or the `-E` option of `bsub`

# LSB\_EXIT\_REQUEUE

## Syntax

```
LSB_EXIT_REQUEUE="exit_value1 exit_value2..."
```

## Description

Contains a list of exit values found in the queue's `REQUEUE_EXIT_VALUES` parameter defined in `l sb. queues`.

## Valid values

Any positive integers

## Default

Not defined

## Notes

If `LSB_EXIT_REQUEUE` is defined, a job will be requeued if it exits with one of the specified values. `LSB_EXIT_REQUEUE` is not defined if the parameter `REQUEUE_EXIT_VALUES` is not defined.

## Where defined

Set by the system based on the value of the parameter `REQUEUE_EXIT_VALUES` in `l sb. queues`

## Example

```
LSB_EXIT_REQUEUE=" 7 31"
```

## See also

`REQUEUE_EXIT_VALUES` in `l sb. queues`

# LSB\_FRAMES

## Syntax

```
LSB_FRAMES=start_number,end_number,step
```

## Description

Determines the number of frames to be processed by a frame job.

## Valid values

The values of *start\_number*, *end\_number*, and *step* are positive integers. Use commas to separate the values.

## Default

Not defined

## Notes

When the job is running, `LSB_FRAMES` will be set to the relative frames with the format `LSB_FRAMES=start_number,end_number,step`.

From the *start\_number*, *end\_number*, and *step*, the frame job can know how many frames it will process.

## Where defined

Set by `sbat chd`

## Example

```
LSB_FRAMES=10, 20, 1
```

# LSB\_HOSTS

## Syntax

```
LSB_HOSTS="host_name..."
```

## Description

A list of hosts selected by LSF to run the job.

## Notes

If a job is run on a single processor, the system sets `LSB_HOSTS` to the name of the host used.

For parallel jobs, the system sets `LSB_HOSTS` to the names of all the hosts used.

## Where defined

Set by `sbat chd` when the job is executed. `LSB_HOSTS` is set only when the list of host names is less than 4096 bytes.

## See also

`LSB_MCPU_HOSTS`

# LSB\_INTERACTIVE

## Syntax

```
LSB_INTERACTIVE=Y
```

## Description

Indicates an interactive job. When you submit an interactive job using `bsub -I`, the system sets `LSB_INTERACTIVE` to Y.

## Valid values

LSB\_INTERACTIVE=Y (if the job is interactive)

## Default

Not defined (if the job is not interactive)

## Where defined

Set by `sbat chd`

# LSB\_JOB\_INCLUDE\_POSTPROC

## Syntax

**LSB\_JOB\_INCLUDE\_POSTPROC=Y | y | N | n**

## Description

Enables the post-execution processing of the job to be included as part of the job.

LSB\_JOB\_INCLUDE\_POSTPROC in the user environment overrides the value of JOB\_INCLUDE\_POSTPROC in `lsb.params` and `lsb.appl icat ions`.

## Default

Not defined

## Where defined

From the command line

# LSB\_JOBEXIT\_INFO

## Syntax

**LSB\_JOBEXIT\_INFO="SIGNAL *signal\_value* *signal\_name*"**

## Description

Contains information about signal that caused a job to exit.

Applies to post-execution commands. Post-execution commands are set with POST\_EXEC in `lsb.queues`.

When the post-execution command is run, the environment variable LSB\_JOBEXIT\_INFO is set if the job is signalled internally. If the job ends successfully, or the job is killed or signalled externally, LSB\_JOBEXIT\_INFO is not set.

## Examples

```
LSB_JOBEXIT_INFO="SIGNAL - 1 SIG_CHKPNT" LSB_JOBEXIT_INFO="SIGNAL - 14 SIG_TERM_USER"  
LSB_JOBEXIT_INFO="SIGNAL - 23 SIG_KILL_REQUEUE"
```

## Default

Not defined

## Where defined

Set by `sbatchd`

# LSB\_JOBEXIT\_STAT

## Syntax

**LSB\_JOBEXIT\_STAT**=*exit\_status*

## Description

Indicates a job's exit status.

Applies to post-execution commands. Post-execution commands are set with `POST_EXEC` in `lsb.queues`.

When the post-execution command is run, the environment variable `LSB_JOBEXIT_STAT` is set to the exit status of the job. Refer to the man page for the `wait(2)` command for the format of this exit status.

The post-execution command is also run if a job is requeued because the job's execution environment fails to be set up, or if the job exits with one of the queue's `REQUEUE_EXIT_VALUES`. The `LSB_JOBPEND` environment variable is set if the job is requeued. If the job's execution environment could not be set up, `LSB_JOBEXIT_STAT` is set to 0.

## Valid values

Any positive integer

## Where defined

Set by `sbatchd`

# LSB\_JOBFILENAME

## Syntax

**LSB\_JOBFILENAME**=*file\_name*

## Description

The path to the batch executable job file that invokes the batch job. The batch executable job file is a `/bin/sh` script on UNIX systems or a `.BAT` command script on Windows systems.

# LSB\_JOBGROUP

## Syntax

**LSB\_JOBGROUP**=*job\_group\_name*

## Description

The name of the job group associated with the job. When a job is dispatched, if it belongs to a job group, the runtime variable `LSB_JOBGROUP` is defined as its group. For example, if a dispatched job belongs to job group `/X`, `LSB_JOBGROUP=/X`.

## Where defined

Set during job execution based on bsub options or the default job group defined in DEFAULT\_JOBGROUP in lsb.params and the LSB\_DEFAULT\_JOBGROUP environment variable.

## Default

Not defined

# LSB\_JOBID

## Syntax

**LSB\_JOBID**=*job\_ID*

## Description

The job ID assigned by sbat chd. This is the ID of the job assigned by LSF, as shown by bj obs.

## Valid values

Any positive integer

## Where defined

Set by sbat chd, defined by mbat chd

## See also

LSB\_REMOTEJID

# LSB\_JOBINDEX

## Syntax

**LSB\_JOBINDEX**=*index*

## Description

Contains the job array index.

## Valid values

Any integer greater than zero but less than the maximum job array size.

## Notes

LSB\_JOBINDEX is set when each job array element is dispatched. Its value corresponds to the job array index. LSB\_JOBINDEX is set for all jobs. For non-array jobs, LSB\_JOBINDEX is set to zero (0).

## Where defined

Set during job execution based on bsub options.

## Example

You can use `LSB_JOBINDEX` in a shell script to select the job command to be performed based on the job array index.

For example:

```
if [ $LSB_JOBINDEX -eq 1 ]; then cmd1 fi if [ $LSB_JOBINDEX -eq 2 ]; then cmd2 fi
```

## See also

`LSB_JOBINDEX_STEP`, `LSB_REMOTEINDEX`

# LSB\_JOBINDEX\_STEP

## Syntax

**LSB\_JOBINDEX\_STEP**=*step*

## Description

Step at which single elements of the job array are defined.

## Valid values

Any integer greater than zero but less than the maximum job array size

## Default

1

## Notes

`LSB_JOBINDEX_STEP` is set when a job array is dispatched. Its value corresponds to the step of the job array index. This variable is set only for job arrays.

## Where defined

Set during job execution based on `bsub` options.

## Example

The following is an example of an array where a step of 2 is used:

```
array[1-10:2] elements: 1 3 5 7 9
```

If this job array is dispatched, then `LSB_JOBINDEX_STEP=2`

## See also

`LSB_JOBINDEX`

# LSB\_JOBNAME

## Syntax

**LSB\_JOBNAME**=*job\_name*

## Description

The name of the job defined by the user at submission time.

## Default

The job's command line

## Notes

The name of a job can be specified explicitly when you submit a job. The name does not have to be unique. If you do not specify a job name, the job name defaults to the actual batch command as specified on the `bsub` command line.

The job name can be up to 4094 characters long for UNIX and Linux or up to 255 characters for Windows.

## Where defined

Set by `sbat chd`

## Example

When you submit a job using the `-J` option of `bsub`, for example:

```
% bsub -J "myjob" job
```

`sbat chd` sets `LSB_JOBNAME` to the job name that you specified:

```
LSB_JOBNAME=myjob
```

# LSB\_JOBPEND

## Description

Set if the job is queued.

## Where defined

Set by `sbat chd` for `POST_EXEC` only

## See also

`LSB_JOBEXIT_STAT`, `REQUEUE_EXIT_VALUES`, `POST_EXEC`

# LSB\_JOBPGIDS

## Description

A list of the current process group IDs of the job.

## Where defined

The process group IDs are assigned by the operating system, and `LSB_JOBPGIDS` is set by `sbat chd`.

## See also

`LSB_JOBPIIDS`

## LSB\_JOBPIIDS

### Description

A list of the current process IDs of the job.

### Where defined

The process IDs are assigned by the operating system, and LSB\_JOBPIIDS is set by sbat chd.

### See also

LSB\_JOBPGIDS

## LSB\_MAILSIZE

### Syntax

**LSB\_MAILSIZE=***value*

### Description

Gives an estimate of the size of the batch job output when the output is sent by email. It is not necessary to configure LSB\_MAILSIZE\_LIMIT.

LSF sets LSB\_MAILSIZE to the size in KB of the job output, allowing the custom mail program to intercept output that is larger than desired.

LSB\_MAILSIZE is not recognized by the LSF default mail program. To prevent large job output files from interfering with your mail system, use LSB\_MAILSIZE\_LIMIT to explicitly set the maximum size in KB of the email containing the job information.

### Valid values

#### A positive integer

If the output is being sent by email, LSB\_MAILSIZE is set to the estimated mail size in kilobytes.

-1

If the output fails or cannot be read, LSB\_MAILSIZE is set to -1 and the output is sent by email using LSB\_MAILPROG if specified in `lsf.conf`.

#### Not defined

If you use the `-o` or `-e` options of `bsub`, the output is redirected to an output file. Because the output is not sent by email in this case, LSB\_MAILSIZE is not used and LSB\_MAILPROG is not called.

If the `-N` option is used with the `-o` option of `bsub`, LSB\_MAILSIZE is not set.

### Where defined

Set by sbat chd when the custom mail program specified by LSB\_MAILPROG in `lsf.conf` is called.

# LSB\_MCPU\_HOSTS

## Syntax

```
LSB_MCPU_HOSTS="host_nameA num_processors1 host_nameB num_processors2..."
```

## Description

Contains a list of the hosts and the number of CPUs used to run a job.

## Valid values

num\_processors1, num\_processors2,... refer to the number of CPUs used on host\_nameA, host\_nameB,..., respectively

## Default

Not defined

## Notes

The environment variables LSB\_HOSTS and LSB\_MCPU\_HOSTS both contain the same information, but the information is presented in different formats. LSB\_MCPU\_HOSTS uses a shorter format than LSB\_HOSTS. As a general rule, sbat chd sets both these variables. However, for some parallel jobs, LSB\_HOSTS is not set.

For parallel jobs, several CPUs are used, and the length of LSB\_HOSTS can become very long. sbat chd needs to spend a lot of time parsing the string. If the size of LSB\_HOSTS exceeds 4096 bytes, LSB\_HOSTS is ignored, and sbat chd sets only LSB\_MCPU\_HOSTS.

To verify the hosts and CPUs used for your dispatched job, check the value of LSB\_HOSTS for single CPU jobs, and check the value of LSB\_MCPU\_HOSTS for parallel jobs.

## Where defined

Set by sbat chd before starting a job on the execution host

## Example

When the you submit a job with the -m and -n options of bsub, for example,

```
% bsub -m "hostA hostB" -n 6 job
```

sbat chd sets the environment variables LSB\_HOSTS and LSB\_MCPU\_HOSTS as follows:

```
LSB_HOSTS= "hostA hostA hostA hostB hostB hostB"  
LSB_MCPU_HOSTS="hostA 3 hostB 3"
```

Both variables are set in order to maintain compatibility with earlier versions.

## See also

LSB\_HOSTS

# LSB\_NQS\_PORT

This parameter can be defined in lsf.conf or in the services database such as /etc/services.

See LSB\_NUM\_NIOS\_CALLBACK\_THREADS in lsf.conf for more details.

## LSB\_NTRIES

### Syntax

**LSB\_NTRIES**=*integer*

### Description

The number of times that LSF libraries attempt to contact `mbat chd` or perform a concurrent jobs query.

For example, if this parameter is not defined, when you type `bj obs`, LSF keeps displaying "batch system not responding" if `mbat chd` cannot be contacted or if the number of pending jobs exceeds `MAX_PEND_JOBS` specified in `l sb. params` or `l sb. users`.

If this parameter is set to a value, LSF only attempts to contact `mbat chd` the defined number of times and then quits. LSF will wait for a period of time equal to `SUB_TRY_INTERVAL` specified in `l sb. params` before attempting to contact `mbat chd` again.

### Valid values

Any positive integer

### Default

`INFINIT_INT` (The default is to continue the attempts to contact `mbat chd`)

## LSB\_OLD\_JOBID

### Syntax

**LSB\_OLD\_JOBID**=*job\_ID*

### Description

The job ID of a job at the time it was checkpointed.

When a job is restarted, it is assigned a new job ID and `LSB_JOBID` is replaced with the new job ID. `LSB_OLD_JOBID` identifies the original ID of a job before it is restarted.

### Valid values

Any positive integer

### Where defined

Set by `sbat chd`, defined by `mbat chd`

### See also

`LSB_JOBID`

## LSB\_OUTPUT\_TARGETFAILED

### Syntax

**LSB\_OUTPUT\_TARGETFAILED**=Y

## Description

Indicates that LSF cannot access the output file specified for a job submitted the `bsub -o` option.

## Valid values

Set to Y if the output file cannot be accessed; otherwise, it is not defined.

## Where defined

Set by `sbat chd` during job execution

# LSB\_DJOB\_COMMFAIL\_ACTION

## Syntax

```
LSB_DJOB_COMMFAIL_ACTION="KILL_TASKS"
```

## Description

Defines the action LSF should take if it detects a communication failure with one or more remote parallel or distributed tasks. If defined, LSF will try to kill all the current tasks of a parallel or distributed job associated with the communication failure. If not defined, the job RES notifies the task RES to terminate all tasks, and shut down the entire job.

## Default

Terminate all tasks, and shut down the entire job

## Valid values

KILL\_TASKS

## Where defined

Set by the system based on the value of the parameter `DJOB_COMMFAIL_ACTION` in `lsb. appl i cat i ons` when running `bsub -app` for the specified application

## See also

`DJOB_COMMFAIL_ACTION` in `lsb. appl i cat i ons`

# LSB\_DJOB\_ENV\_SCRIPT

## Syntax

```
LSB_DJOB_ENV_SCRIPT=script_name
```

## Description

Defines the name of a user-defined script for setting and cleaning up the parallel or distributed job environment. This script will be executed by LSF with the argument `set up` before launching a parallel or distributed job, and with argument `clean up` after the parallel job is finished.

The script will run as the user, and will be part of the job.

If a full path is specified, LSF will use the path name for the execution. Otherwise, LSF will look for the executable from `$LSF_BINDIR`.

## Where defined

Set by the system to the value of the parameter `DJOB_ENV_SCRIPT` in `lsb. applications` when running `bsub -app` for the specified application

## See also

`DJOB_ENV_SCRIPT` in `lsb. applications`

# LSB\_QUEUE

## Syntax

**LSB\_QUEUE**=*queue\_name*

## Description

The name of the queue from which the job is dispatched.

## Where defined

Set by `sbatchd`

# LSB\_REMOTEINDEX

## Syntax

**LSB\_REMOTEINDEX**=*index*

## Description

The job array index of a remote MultiCluster job. `LSB_REMOTEINDEX` is set only if the job is an element of a job array.

## Valid values

Any integer greater than zero, but less than the maximum job array size

## Where defined

Set by `sbatchd`

## See also

`LSB_JOBINDEX`, `MAX_JOB_ARRAY_SIZE` in `lsb. params`

# LSB\_REMOTEJID

## Syntax

**LSB\_REMOTEJID**=*job\_ID*

## Description

The job ID of a remote MultiCluster job.

## Where defined

Set by `sbat chd`, defined by `mbat chd`

## See also

`LSB_JOBID`

# LSB\_RESTART

## Syntax

`LSB_RESTART=Y`

## Description

Indicates that a job has been restarted or migrated.

## Valid values

Set to Y if the job has been restarted or migrated; otherwise, it is not defined.

## Notes

If a batch job is submitted with the `-r` option of `bsub`, and is restarted because of host failure, then `LSB_RESTART` is set to Y. If a checkpointable job is submitted with the `-k` option of `bsub`, then `LSB_RESTART` is set to Y when the job is restarted. If `bmi g` is used to migrate a job, then `LSB_RESTART` is set to Y when the migrated job is restarted.

If the job is not a restarted job, then `LSB_RESTART` is not set.

## Where defined

Set by `sbat chd` during job execution

## See also

`LSB_RESTART_PGID`, `LSB_RESTART_PID`

# LSB\_RESTART\_PGID

## Syntax

`LSB_RESTART_PGID=pgid`

## Description

The process group ID of the checkpointed job when the job is restarted.

## Notes

When a checkpointed job is restarted, the operating system assigns a new group process ID to the job. LSF sets `LSB_RESTART_PGID` to the new group process ID.

## Where defined

Set during restart of a checkpointed job.

## See also

LSB\_RESTART\_PID, LSB\_RESTART

# LSB\_RESTART\_PID

## Syntax

**LSB\_RESTART\_PID=*pid***

## Description

The process ID of the checkpointed job when the job is restarted.

## Notes

When a checkpointed job is restarted, the operating system assigns a new process ID to the job. LSF sets LSB\_RESTART\_PID to the new process ID.

## Where defined

Defined during restart of a checkpointed job

## See also

LSB\_RESTART\_PGID, LSB\_RESTART

# LSB\_RTASK\_GONE\_ACTION

## Syntax

**LSB\_RTASK\_GONE\_ACTION=*task\_action*...**

## Description

Defines the actions LSF should take if it detects that a remote task of a parallel job is gone. Where *task\_action* is:

### **IGNORE\_TASKCRASH**

A remote task crashes. The job RES does nothing.

### **KILLJOB\_TASKDONE**

A remote task exits with zero value. The job RES notifies the task RES to terminate all tasks in the job.

### **KILLJOB\_TASKEEXIT**

A remote task exits with non-zero value. The job RES notifies the task RES to terminate all tasks in the job.

## Where defined

Set by the system based on the value of the parameter `RTASK_GONE_ACTION` in `lsb. applications` when running `bsub -app` for the specified application

## See also

`RTASK_GONE_ACTION` in `lsb. applications`

# LSB\_SUB\_CLUSTER

## Description

Name of submission cluster (MultiCluster only)

## Where defined

Set on the submission environment and passed to the execution cluster environment. The parameter will ONLY be valid in Multi Cluster environment. For jobs on a local cluster, the parameter is not set when using any daemon wrappers such as job starter, post-, pre- or eexec scripts.

# LSB\_SUB\_COMMAND\_LINE

## Description

The job command line.

The job command line can be up to 4094 characters long for UNIX and Linux or up to 255 characters for Windows.

## Where defined

Set by `esub` before a job is submitted.

# LSB\_SUB\_EXTSCHED\_PARAM

## Description

Value of external scheduling options specified by `bsub -extsched`, or queue-level `MANDATORY_EXTSCHED` or `DEFAULT_EXTSCHED`.

## Where defined

Set by `esub` before a job is submitted.

# LSB\_SUB\_JOB\_ACTION\_WARNING\_TIME

## Description

Value of job warning time period specified by `bsub -wt`.

## Where defined

Set by `esub` before a job is submitted.

## LSB\_SUB\_JOB\_WARNING\_ACTION

### Description

Value of job warning action specified by `bsub -wa`.

### Where defined

Set by `esub` before a job is submitted.

## LSB\_SUB\_PARM\_FILE

### Syntax

**LSB\_SUB\_PARM\_FILE**=*file\_name*

### Description

Points to a temporary file that LSF uses to store the `bsub` options entered in the command line. An `esub` reads this file at job submission and either accepts the values, changes the values, or rejects the job. Job submission options are stored as name-value pairs on separate lines in the format `option_name=value`. A typical use of this file is to control job submission options.

### Where defined

Set by LSF on the submission host before running `esub`. Not defined when `l srun` or `l sgrun` are used for interactive remote execution.

## LSB\_SUCCESS\_EXIT\_VALUES

### Syntax

**LSB\_SUCCESS\_EXIT\_VALUES**=[*exit\_code ...*]

### Description

Specifies the exit values that indicate successful execution for applications that successfully exit with non-zero values. Use spaces to separate multiple exit codes. `exit_code` should be the value between 0 and 255.

User-defined `LSB_SUCCESS_EXIT_VALUES` overrides application profile level specification of `SUCCESS_EXIT_VALUES` in `lsb.appl icat ions`.

## LSB\_SUSP\_REASONS

### Syntax

**LSB\_SUSP\_REASONS**=*integer*

### Description

An integer representing suspend reasons. Suspend reasons are defined in `lsbat ch. h`.

This parameter is set when a job goes to system-suspended (SSUSP) or user-suspended status (USUSP). It indicates the exact reason why the job was suspended.

To determine the exact reason, you can test the value of `LSB_SUSP_REASONS` against the symbols defined in `lsbatch.h`.

## Where defined

Set during job execution

## See also

`LSB_SUSP_SUBREASONS`

# LSB\_SUSP\_SUBREASONS

## Syntax

`LSB_SUSP_SUBREASONS=integer`

## Description

An integer representing the load index that caused a job to be suspended.

When the suspending reason `SUSP_LOAD_REASON` (suspended by load) is set in `LSB_SUSP_REASONS`, `LSB_SUSP_SUBREASONS` set to one of the load index values defined in `lsbatch.h`.

Use `LSB_SUSP_REASONS` and `LSB_SUSP_SUBREASONS` together in your custom job control to determine the exact load threshold that caused a job to be suspended.

Load index values are defined in `lsbatch.h`.

Load Index	Value
R15S	0
R1M	1
R15M	2
UT	3
PG	4
IO	5
LS	6
IT	7
TMP	8
SWP	9
MEM	10

## Default

Not defined

## Where defined

Set during job execution

## See also

LSB\_SUSP\_REASONS

# LSB\_UNIXGROUP

## Description

Specifies the UNIX user group of the submitting user.

## Notes

This variable is useful if you want pre- or post-execution processing to use the user group of the user who submitted the job, and not `sys(1)`.

## Where defined

Set during job execution

# LSB\_USER\_BIND\_CPU\_LIST

The binding requested at job submission takes effect when `LSF_BIND_JOB=USER_CPU_LIST` in `lsf.conf` or `BIND_JOB=USER_CPU_LIST` in an application profile in `lsb.appl icat ions`. LSF makes sure that the value is in the correct format, but does not check that the value is valid for the execution hosts.

The correct format is a list which may contain multiple items, separated by comma, and ranges. For example: 0,5,7,9-11.

# LSB\_USER\_BIND\_JOB

The binding requested at job submission takes effect when `LSF_BIND_JOB=USER` in `lsf.conf` or `BIND_JOB=USER` in an application profile in `lsb.appl icat ions`. This value must be one of Y, N, NONE, BALANCE, PACK, or ANY. Any other value is treated as ANY.

# LSF\_CMD\_LOGDIR

This parameter can be set from the command line or from `lsf.conf`.

See `LSF_CMD_LOGDIR` in `lsf.conf`.

# LSF\_DEBUG\_CMD

This parameter can be set from the command line or from `lsf.conf`.

See `LSB_DEBUG_MBD` in `lsf.conf`.

# LSF\_DEBUG\_LIM

This parameter can be set from the command line or from `lsf.conf`.

See `LSF_DEBUG_LIM` in `lsf.conf`.

## LSF\_DEBUG\_RES

This parameter can be set from the command line or from `lsf.conf`.

See `LSF_DEBUG_RES` in `lsf.conf`.

## LSF\_EAUTH\_AUX\_DATA

### Syntax

`LSF_EAUTH_AUX_DATA=path/file_name`

### Description

Used in conjunction with LSF daemon authentication, specifies the full path to the temporary file on the local file system that stores auxiliary authentication information (such as credentials required by a remote host for use during job execution). Provides a way for `eauth -c`, `mbatchd`, and `sbatchd` to communicate the location of auxiliary authentication data. Set internally by the LSF libraries in the context of `eauth`.

For Kerberos authentication, used for forwarding credentials to the execution host.

## LSF\_EAUTH\_AUX\_PASS

### Syntax

`LSF_EAUTH_AUX_PASS=yes`

### Description

Enables forwarding of credentials from a submission host to an execution host when daemon authentication is enabled. `LSF_EAUTH_AUX_PASS=yes` indicates that a credential can be added to the execution context of a job. Set to `yes` by `bsub` during job submission or by `bmod` during job modification so that `eauth -c` can forward credentials.

## LSF\_EAUTH\_CLIENT

### Syntax

`LSF_EAUTH_CLIENT=mbatchd | sbatchd | pam | res | user`

### Description

Used with LSF daemon authentication, specifies the LSF daemon, command, or user that invokes `eauth -c`. Used when writing a customized `eauth` executable to set the context for the call to `eauth`. Set internally by the LSF libraries or by the LSF daemon, command, or user calling `eauth -c`.

## LSF\_EAUTH\_SERVER

### Syntax

`LSF_EAUTH_SERVER=mbatchd | sbatchd | pam | res`

### Description

Used with LSF daemon authentication, specifies the daemon that invokes `eauth -s`. Used when writing a customized `eauth` executable to set the context for the call to `eauth`. Set internally by the LSF libraries or by the LSF daemon calling `eauth -s`.

## LSF\_EAUTH\_UID

### Syntax

**LSF\_EAUTH\_UID**=*user\_ID*

### Description

Specifies the user account under which `eauth -s` runs. Set by the LSF daemon that executes `eauth`. Set by the LSF daemon that executes `eauth`.

## LSF\_EXECUTE\_DOMAIN

### Syntax

**LSF\_EXECUTE\_DOMAIN**=*domain\_name*`setenv LSF_EXECUTE_DOMAIN` *domain\_name*

### Description

If UNIX/Windows user account mapping is enabled, specifies the preferred Windows execution domain for a job submitted by a UNIX user. The execution domain must be one of the domains listed in `LSF_USER_DOMAIN`.

`LSF_EXECUTE_DOMAIN` is defined in the user environment (`.cshrc` or `.profile`) or from the command line. Specify only one domain.

Use this parameter in conjunction with the `bsub`, `l sr un`, and `l sgr un` commands to bypass the order of the domains listed in `LSF_USER_DOMAIN` and run the job using the specified domain. If you do not have a Windows user account in the execution domain, LSF tries to run the job using one of the other domains defined by `LSF_USER_DOMAIN`. Once you submit a job with an execution domain defined, you cannot change the execution domain for that particular job.

## LSF\_INTERACTIVE\_STDERR

This parameter can be defined in `lsf.conf`.

See `LSF_INTERACTIVE_STDERR` in `lsf.conf` for more details.

## LSF\_INVOKE\_CMD

### Syntax

**LSF\_INVOKE\_CMD**=*invoking\_command\_name*

### Description

Indicates the name of the last LSF command that invoked an external executable (for example, `esub` or `eexec`).

External executables get called by different LSF commands, such as `bsub`, `bmod`, or `l sr un`.

### Default

Not defined

### Where defined

Set internally within by LSF

# LSF\_JOB\_STARTER

## Syntax

**LSF\_JOB\_STARTER**=*binary*

## Description

Specifies an executable program that has the actual job as an argument.

## Default

Not defined

## Interactive Jobs

If you want to run an interactive job that requires some preliminary setup, LSF provides a job starter function at the command level. A command-level job starter allows you to specify an executable file that will run prior to the actual job, doing any necessary setup and running the job when the setup is complete.

If LSF\_JOB\_STARTER is properly defined, RES will invoke the job starter (rather than the job itself), supplying your commands as arguments.

## Batch Jobs

A job starter can also be defined at the queue level using the JOB\_STARTER parameter, although this can only be done by the LSF administrator.

## Where defined

From the command line

## Example: UNIX

The job starter is invoked from within a Bourne shell, making the command-line equivalent:

```
/bin/sh -c "$LSF_JOB_STARTER command [argument...]"
```

where *command* [*argument...*] are the command line arguments you specified in `l srun`, `l sgrun`, or `ch`.

If you define LSF\_JOB\_STARTER as follows:

```
set env LSF_JOB_STARTER "/bin/csh -c"
```

and run a simple C-shell job:

```
l srun "' a.out; echo hi '"
```

The following will be invoked to correctly start the job:

```
/bin/sh -c "/bin/csh -c ' a.out; echo hi '"
```

## Example: Windows

RES runs the job starter, passing it your commands as arguments:

```
LSF_JOB_STARTER command [argument...]
```

If you define LSF\_JOB\_STARTER as follows:

```
set LSF_JOB_STARTER=C:\cmd.exe /C
```

and run a simple DOS shell job:

```
C: \> lsrun dir /p
```

then the following will be invoked to correctly start the job:

```
C: \cmd.exe /C dir /p
```

## See also

JOB\_STARTER in `lsb.queues`

# LSF\_LD\_LIBRARY\_PATH

## Description

When `LSF_LD_SECURITY=Y` in `lsf.conf`, contains the value of the `LD_LIBRARY_PATH` environment variable, which is removed from the job environment during job initialization to ensure enhanced security against users obtaining root privileges. `LSF_LD_LIBRARY_PATH` allows the `LD_LIBRARY_PATH` environment variable to be put back before the job runs.

## Where defined

For jobs submitted using `bsub -Is` or `bsub -Ip` only.

## See also

`LSF_LD_PRELOAD`, `LSF_LD_SECURITY` in `lsf.conf`

# LSF\_LD\_PRELOAD

## Description

When `LSF_LD_SECURITY=Y` in `lsf.conf`, contains the value of the `LD_PRELOAD` environment variable, which is removed from the job environment during job initialization to ensure enhanced security against users obtaining root privileges. `LSF_LD_PRELOAD` allows the `LD_PRELOAD` environment variable to be put back before the job runs.

## Where defined

For jobs submitted using `bsub -Is` or `bsub -Ip` only.

## See also

`LSF_LD_LIBRARY_PATH`, `LSF_LD_SECURITY` in `lsf.conf`

# LSF\_LIM\_API\_NTRIES

## Syntax

**LSF\_LIM\_API\_NTRIES**=*integer*

## Description

Defines the number of times LSF commands will retry to communicate with the LIM API when LIM is not available. `LSF_LIM_API_NTRIES` is ignored by LSF and EGO daemons and EGO commands. The `LSF_LIM_API_NTRIES` environment variable overrides the value of `LSF_LIM_API_NTRIES` in `lsf.conf`.

## Valid values

1 to 65535

## Where defined

From the command line or from `lsf.conf`

## Default

Not defined. If not defined in `lsf.conf`, LIM API exits without retrying.

## LSF\_LIM\_DEBUG

This parameter can be set from the command line or from `lsf.conf`.

See `LSF_LIM_DEBUG` in `lsf.conf`.

## LSF\_LOGDIR

This parameter can be set from the command line or from `lsf.conf`.

See `LSF_LOGDIR` in `lsf.conf`.

## LSF\_MASTER

### Description

Set by the LIM to identify the master host. The value is Y on the master host and N on all other hosts. An `elim` executable can use this parameter to check the host on which the `elim` is currently running.

Used when the external load indices feature is enabled.

### When defined

Set by the LIM when it starts the master external load information manager (MELIM).

### See also

`LSF_RESOURCES`

## LSF\_NIOS\_DEBUG

This parameter can be set from the command line or from `lsf.conf`.

See `LSF_NIOS_DEBUG` in `lsf.conf`.

## LSF\_NIOS\_ERR\_LOGDIR

This parameter can be set from the command line or from `lsf.conf`.

See `LSF_NIOS_ERR_LOGDIR` in `lsf.conf`.

## LSF\_NIOS\_DIE\_CMD

### Syntax

`LSF_NIOS_DIE_CMD=command`

## Description

If set, the command defined by `LSF_NIOS_DIE_CMD` is executed before NIOS exits.

## Default

Not defined

## Where defined

From the command line

# LSF\_NIOS\_IGNORE\_SIGWINDOW

## Syntax

`LSF_NIOS_IGNORE_SIGWINDOW=any_value`

## Description

If defined, the NIOS will ignore the SIGWINDOW signal.

## Default

Not defined

## Notes

When the signal SIGWINDOW is defined, some tasks appear to die when they receive the SIGWINDOW while doing I/O. By defining `LSF_NIOS_IGNORE_SIGWINDOW`, these tasks are given the chance to ignore the signal.

## Where defined

From the command line

# LSF\_NIOS\_PEND\_TIMEOUT

## Syntax

`LSF_NIOS_PEND_TIMEOUT=minutes`

## Description

Applies only to interactive batch jobs.

Maximum amount of time that an interactive batch job can remain pending.

If this parameter is defined, and an interactive batch job is pending for longer than the specified time, the interactive batch job is terminated.

## Valid values

Any integer greater than zero

## Default

Not defined

# LSF\_NIOS\_PORT\_RANGE

## Syntax

**LSF\_NIOS\_PORT\_RANGE**=*min\_port\_number-max\_port\_number*

## Description

Defines a range of listening ports for NIOS to use.

## Example

```
LSF_NIOS_PORT_RANGE=5000-6000
```

## Default

Not defined. LSF randomly assigns a NIOS port number.

# LSF\_RESOURCES

## Syntax

**LSF\_RESOURCES**=*dynamic\_external\_resource\_name...*

## Description

Space-separated list of dynamic external resources. When the LIM starts a master external load information manager (MELIM) on a host, the LIM checks the resource mapping defined in the ResourceMap section of `lsf.cluster.cluster_name`. Based on the mapping (default, all, or a host list), the LIM sets `LSF_RESOURCES` to the list of resources expected on the host and passes the information to the MELIM.

Used when the external load indices feature is enabled.

## When defined

Set by the MELIM on the host when the MELIM invokes the `elim` executable.

## See also

`LSF_MASTER`

# LSF\_TS\_LOGON\_TIME

## Syntax

**LSF\_TS\_LOGON\_TIME**=*milliseconds*

## Description

Specifies the time to create a Windows Terminal Service session. Configure `LSF_TS_LOGON_TIME` according to the load on your network environment.

The default, 30000 milliseconds, is suitable for most environments. If you set `LSF_TS_LOGON_TIME` too small, the LSF tries multiple times before it succeeds in making a TS session with the TS server, which can cause the job wait a long time before it runs. For a congested network. set `LSF_TS_LOGON_TIME=1000000`.

## Where defined

From the command line

## Default

30000 milliseconds

# LSF\_USE\_HOSTEQUIV

## Syntax

`LSF_USE_HOSTEQUIV=y | Y`

## Description

Used for authentication purposes. If `LSF_USE_HOSTEQUIV` is defined, `RES` and `mbatchd` call the `ruserok(3)` function to decide if a user is allowed to run remote jobs. LSF trusts all hosts configured in the LSF cluster that are defined in `hosts.equiv`, or in `.rhosts` in the user's home directory.

The `ruserok(3)` function checks in the `/etc/hosts.equiv` file and the user's `$HOME/.rhosts` file to decide if the user has permission to execute remote jobs.

If `LSF_USE_HOSTEQUIV` is not defined, all normal users in the cluster can execute remote jobs on any host.

If `LSF_ROOT_REX` is set, root can also execute remote jobs with the same permission test as for normal users.

## Default

Not defined

## See also

`LSF_ROOT_REX` and `LSF_AUTH` in `lsf.conf`

# LSF\_USER\_DOMAIN

## Syntax

`LSF_USER_DOMAIN=domain_name / .`

## Description

Set during LSF installation or setup. If you modify this parameter in an existing cluster, you probably have to modify passwords and configuration files also.

Windows or mixed UNIX-Windows clusters only.

Enables default user mapping, and specifies the LSF user domain. The period ( `.` ) specifies local accounts, not domain accounts.

- A user name specified without a domain is interpreted (on a Windows host) as belonging to the LSF user domain
- A user name specified with the domain name of the LSF user domain is not valid
- In a mixed cluster, this parameter defines a 2-way, 1:1 user map between UNIX user accounts and Windows user accounts belonging to the specified domain, as long as the accounts have the same user name. This means jobs submitted by the Windows user account can run on a UNIX host, and jobs submitted by the UNIX account can run on any Windows host that is available to the Windows user account.

If this parameter is not defined, the default user mapping is not enabled. You can still configure user mapping at the user or system level. User account mapping is required to run cross-platform jobs in a UNIX-Windows mixed cluster.

## Where defined

lsf.conf

## Default

- If you upgrade from LSF 4.0.1 or earlier, the default is the existing LSF user domain.
- For a new, Windows-only cluster, this parameter is not defined (no LSF user domain, no default user mapping).
- For a new, mixed UNIX-Windows cluster, the default is the domain that the Windows installation account belongs to. This can be modified during LSF installation.

---

# Index

.lsftask file 477

## A

ABS\_RUNLIMIT

Isb.params file 41, 132

ACCESS\_CONTROL

Isb.serviceclasses file 273

ACCINUSE\_INCLUDES\_OWNERSHIP

Isf.licensescheduler file Parameters section 458

ACCT\_ARCHIVE\_AGE

Isb.params file 133

ACCT\_ARCHIVE\_SIZE

Isb.params file 133

ACCT\_ARCHIVE\_TIME

Isb.params file 134

ADJUST\_DURATION

Isf.cluster file 295

ADMIN

Isb.hosts file 123

Isf.licensescheduler file Parameters section 437

administrator

user group 285

ADMINISTRATORS

Isb.queues file 188

Isf.cluster file 305

ADVRSV\_USER\_LIMIT

Isb.params file 134

ALLOCATION

Isf.licensescheduler file Feature section 448

APS\_PRIORITY

Isb.queues file 188

ARCHITECTURE

Isf.shared file 467

AUTH

Isf.licensescheduler file Parameters section 437

AUTO\_ATTACH

Isb.serviceclasses file 274

automatic time-based configuration

Isb.hosts 125

Isb.params 186

Isb.queues 238

Isb.resources 270

Isb.users 290

Isf.licensescheduler 464

## B

BACKFILL

Isb.queues file 189

bconf

enable 400

BIND\_JOB 42

BLC\_HEARTBEAT\_FACTOR

Isf.licensescheduler file Parameters section 443

bld

License Scheduler daemon 445

bld.license.acct file 7

bld.stream file 441, 442

BSUB\_BLOCK variable 496

BSUB\_QUIET variable 496, 497

BSUB\_QUIET2 variable 497

BSUB\_STDERR variable 498

## C

CACHE\_INTERVAL

Isf.cluster file 313

CHKPNT

Isb.hosts file 112

Isb.queues file 189

CHKPNT\_DIR

Isb.applications file 42

CHKPNT\_INITPERIOD

Isb.applications file 43

CHKPNT\_METHOD

Isb.applications file 44

CHKPNT\_PERIOD  
     Isb.applications file 43  
 chunk jobs  
     CHKPNT parameter in Isb.queues 190  
     MIG parameter in Isb.queues 55, 214  
     rerunnable 62, 224  
 CHUNK\_JOB\_DURATION  
     Isb.params file 134  
 CHUNK\_JOB\_SIZE  
     Isb.applications file 44  
     Isb.queues file 190  
 CLEAN\_PERIOD  
     Isb.params file 135  
 CLEARCASE\_DRIVE variable 498  
 CLEARCASE\_MOUNTDIR variable 499  
 CLEARCASE\_ROOT variable 499  
 CLOSE\_ON\_DEMAND  
     Isb.resources file GuaranteedResourcePool section 260  
 ClusterName  
     Isf.shared file 465  
 CLUSTERNAME  
     Isf.cluster file 312  
 CLUSTERS  
     Isf.licensescheduler file Clusters section 444  
 Clusters section  
     Isf.licensescheduler file  
         description 444  
 COMMITTED\_RUN\_TIME\_FACTOR  
     Isb.params file 135  
     Isb.queues file 191  
 COMPUTE\_UNIT\_TYPES  
     Isb.params file 136  
 CONDENSE  
     Isb.hosts file 116, 122  
 CONDENSE\_PENDING\_REASONS  
     Isb.params file 136  
 configurable job ID limit 161  
 CONSUMABLE  
     Isf.shared file 469  
 CONSUMER  
     Isb.serviceclasses file 275  
 CONTROL\_ACTION  
     Isb.serviceclasses file 275  
 CORELIMIT  
     Isb.applications file 45  
     Isb.queues file 191  
 CPU\_TIME\_FACTOR  
     Isb.params file 137  
     Isb.queues file 193  
 CPUFACTOR  
     Isf.shared file 467  
 CPULIMIT  
     Isb.applications file 45  
     Isb.queues file 192  
 Cray checkpointing 112  
 cross-cluster account mapping 289  
 cross-cluster account mapping in MultiCluster 289  
 cross-cluster user account mapping in MultiCluster 289  
 CSA (IRIX Comprehensive System Accounting)  
     configuring and using 381  
 cshrc.Isf file  
     description 9  
     setting the LSF environment 10  
 CUMULATIVE\_RUSAGE  
     LSF HPC extensions parameter 386  
 custom resources  
     reserving 269

## D

daemons  
     automatic shut down 318  
     security 426  
 DATALIMIT  
     Isb.applications file 46  
     Isb.queues file 193  
 dedicated resource. *See* exclusive resource 308  
 DEFAULT\_APPLICATION  
     Isb.params file 137  
 DEFAULT\_EXTSCHED  
     Isb.queues file 193  
 DEFAULT\_HOST\_SPEC  
     Isb.params file 138  
     Isb.queues file 194  
 DEFAULT\_JOBGROUP  
     Isb.params file 138  
 DEFAULT\_PROJECT  
     Isb.params file 139  
 DEFAULT\_QUEUE  
     Isb.params file 139  
 DEFAULT\_SLA\_VELOCITY  
     Isb.params file 140  
 DEFAULT\_USER\_GROUP  
     Isb.params file 140  
 DESCRIPTION  
     Isb.applications file 46

- lsb.queues file 194
- lsb.resources file GuaranteedResourcePool section 260
- lsb.serviceclasses file 276
- lsf.licensescheduler file Project section 464
- lsf.licensescheduler file ProjectGroup section 462
- lsf.shared file 470
- DETECT\_IDLE\_JOB\_AFTER
  - lsb.params file 141
- DIRECTION
  - lsb.users file 289
- DISABLE\_UACCT\_MAP
  - lsb.params file 141
- DISP\_RES\_USAGE\_LIMITS
  - LSF HPC extensions parameter 386
- DISPATCH\_ORDER
  - lsb.queues file 194
- DISPATCH\_WINDOW
  - lsb.hosts file 112
  - lsb.queues file 195
- DISTRIBUTION
  - lsb.resources file GuaranteedResourcePool section 259
  - lsb.resources file HostExport section 262
  - lsb.resources file SharedResourceExport section 264
  - lsf.licensescheduler file Feature section 447
- DISTRIBUTION\_POLICY\_VIOLATION\_ACTION
  - lsf.licensescheduler file Parameters section 437
- DJOB\_COMMFAIL\_ACTION
  - lsb.applications file 47
- DJOB\_DISABLED
  - lsb.applications file 47
- DJOB\_ENV\_SCRIPT
  - lsb.applications file 47
- DJOB\_HB\_INTERVAL
  - lsb.applications file 48
- DJOB\_RU\_INTERVAL
  - lsb.applications file 48
- dual-core CPUs
  - enabling detection 382
- dual-stack hosts
  - setting to IPv6 only 378
- DURATION
  - lsb.resources file GuaranteedResourcePool section 260
- DYNAMIC
  - lsf.licensescheduler file Feature section 457

**E**

- EADMIN\_TRIGGER\_DURATION
  - lsb.params file 141
- EGO\_BINDIR
  - cshrc.lsf and profile.lsf files 14
- EGO\_CONF\_RETRY\_INT parameter in ego.conf 369
- EGO\_CONF\_RETRY\_MAX parameter in ego.conf 370
- EGO\_CONFDIR
  - cshrc.lsf and profile.lsf files 15
- EGO\_DAEMON\_CONTROL
  - install.config file 22, 483
- EGO\_DEBUG\_LIM parameter in ego.conf 375
- EGO\_DHCP\_ENV parameter in ego.conf 376
- EGO\_DYNAMIC\_HOST\_TIMEOUT parameter in ego.conf 379
- EGO\_DYNAMIC\_HOST\_WAIT\_TIME parameter in ego.conf 380
- EGO\_ENABLE\_AUTO\_DAEMON\_SHUTDOWN 318
- EGO\_ENABLE\_DUALCORE parameter in ego.conf 382
- EGO\_ESLIM\_TIMEOUT 318
- EGO\_ESRVDIR
  - cshrc.lsf and profile.lsf files 15
- EGO\_LIBDIR
  - cshrc.lsf and profile.lsf files 16
- EGO\_LIM\_CONNTIMEOUT parameter in ego.conf 365
- EGO\_LIM\_DEBUG parameter in ego.conf 399
- EGO\_LIM\_PORT parameter in ego.conf 400
- EGO\_LIM\_RECVTIMEOUT parameter in ego.conf 365
- EGO\_LOCAL\_CONFDIR
  - cshrc.lsf and profile.lsf files 16
- EGO\_LOCAL\_RESOURCES parameter in ego.conf 402
- EGO\_LOG\_MASK parameter in ego.conf 374, 403
- EGO\_MASTER\_LIST parameter in ego.conf 409
- EGO\_PIM\_INFODIR parameter in ego.conf 416
- EGO\_PIM\_SLEEPTIME parameter in ego.conf 417
- EGO\_PIM\_SLEEPTIME\_UPDATE parameter in ego.conf 417
- EGO\_RES\_REQ
  - lsb.serviceclasses file 276
- EGO\_RSH parameter in ego.conf 423
- EGO\_SERVERDIR
  - cshrc.lsf and profile.lsf files 16
- EGO\_STATIC\_LIM\_TIMEOUT 318
- EGO\_STRIP\_DOMAIN parameter in ego.conf 428
- EGO\_TIME\_LIM parameter in ego.conf 428
- EGO\_TOP
  - cshrc.lsf and profile.lsf files 17
- ego.conf file
  - corresponding lsf.conf parameters 316
  - LSF parameter migration for upgrade 316
  - managing error logs 374

ELIM\_ABORT\_VALUE  
     lsf.cluster file 295  
 ELIM\_ABORT\_VALUE variable 500  
 ELIM\_POLL\_INTERVAL  
     lsf.cluster file 295  
 ELIMARGS  
     lsf.cluster file 296  
 email  
     configuring on UNIX 341  
 ENABLE\_DEFAULT\_EGO\_SLA  
     lsb.params file 142  
 ENABLE\_DYNAMIC\_HOSTS  
     install.config file 22  
 ENABLE\_DYNAMIC\_RUSAGE  
     lsf.licensescheduler file Feature section 457  
 ENABLE\_EGO  
     install.config file 22, 484  
 ENABLE\_EVENT\_STREAM  
     lsb.params file 142  
 ENABLE\_HIST\_RUN\_TIME  
     lsb.params file 143  
     lsb.queues file 195  
 ENABLE\_HOST\_INTERSECTION  
     lsb.params¶ 143  
 ENABLE\_HPC\_INST  
     install.config file 23  
 ENABLE\_INTERACTIVE  
     lsf.licensescheduler file Parameters section 438  
 ENABLE\_MINJOB\_PREEMPTION  
     lsf.licensescheduler file Feature section 458  
 ENABLE\_USER\_RESUME  
     lsb.params file 143  
 ENFORCE\_ONE\_UG\_LIMITS  
     lsb.params file 144  
 ENFORCE\_UG\_TREE  
     lsb.params file 144  
 environment variables  
     BSUB\_BLOCK 496  
     BSUB\_QUIET 496, 497  
     BSUB\_QUIET2 497  
     BSUB\_STDERR 498  
     CLEARCASE\_DRIVE 498  
     CLEARCASE\_MOUNTDIR 499  
     CLEARCASE\_ROOT 499  
     ELIM\_ABORT\_VALUE 500  
     LM\_LICENSE\_FILE 500  
     LS\_EXEC\_T 501  
     LS\_JOBPID 501  
     LS\_LICENSE\_SERVER\_feature 501  
     LS\_SUBCWD 502  
     LSB\_CHKPNT\_DIR 502  
     LSB\_DEBUG 502  
     LSB\_DEBUG\_CMD 503  
     LSB\_DEBUG\_MBD 503  
     LSB\_DEBUG\_NQS 503  
     LSB\_DEBUG\_SBD 503  
     LSB\_DEBUG\_SCH 503  
     LSB\_DEFAULT\_JOBGROUP 503  
     LSB\_DEFAULTPROJECT 504  
     LSB\_DEFAULTQUEUE 505  
     LSB\_DJOB\_COMMFAIL\_ACTION 518  
     LSB\_DJOB\_ENV\_SCRIPT 518  
     LSB\_DJOB\_NUMPROC 505  
     LSB\_ECHKPNT\_KEEP\_OUTPUT 506  
     LSB\_ECHKPNT\_METHOD 506  
     LSB\_ECHKPNT\_METHOD\_DIR 506  
     LSB\_ERESTART\_USRCMD 506  
     LSB\_EXEC\_RUSAGE 506  
     LSB\_EXECHOSTS 507  
     LSB\_EXIT\_IF\_CWD\_NOTEXIST 507  
     LSB\_EXIT\_PRE\_ABORT 507  
     LSB\_EXIT\_REQUEUE 508  
     LSB\_FRAMES 508  
     LSB\_HOSTS 509  
     LSB\_INTERACTIVE 509  
     LSB\_JOB\_INCLUDE\_POSTPROC 510  
     LSB\_JOBEXIT\_INFO 510  
     LSB\_JOBEXIT\_STAT 511  
     LSB\_JOBFILENAME 511  
     LSB\_JOBGROUP 511  
     LSB\_JOBID 512  
     LSB\_JOBINDEX 512  
     LSB\_JOBINDEX\_STEP 513  
     LSB\_JOBNAME 513  
     LSB\_JOBPEND 514  
     LSB\_JOBPGIDS 514  
     LSB\_JOBPIIDS 515  
     LSB\_MAILSIZE 515  
     LSB\_MCPU\_HOSTS 516  
     LSB\_NQS\_PORT 516  
     LSB\_NTRIES 517  
     LSB\_OLD\_JOBID 517  
     LSB\_OUTPUT\_TARGETFAILED 517  
     LSB\_QUEUE 519  
     LSB\_REMOTEINDEX 519  
     LSB\_REMOTEJID 519

- LSB\_RESTART 520
- LSB\_RESTART\_PGID 520
- LSB\_RESTART\_PID 521
- LSB\_RTASK\_GONE\_ACTION 521
- LSB\_SUB\_CLUSTER 522
- LSB\_SUB\_COMMAND\_LINE 522
- LSB\_SUB\_EXTSCHED\_PARAM 522
- LSB\_SUB\_JOB\_ACTION\_WARNING\_TIME 522
- LSB\_SUB\_JOB\_WARNING\_ACTION 523
- LSB\_SUB\_PARM\_FILE 523
- LSB\_SUCCESS\_EXIT\_VALUES 523
- LSB\_SUSP\_REASONS 523
- LSB\_SUSP\_SUBREASONS 524
- LSB\_UNIXGROUP 525
- LSB\_USER\_BIND\_CPU\_LIST 525
- LSB\_USER\_BIND\_JOB 525
- LSF\_CMD\_LOGDIR 525
- LSF\_DEBUG\_CMD 525
- LSF\_DEBUG\_LIM 525
- LSF\_DEBUG\_RES 526
- LSF\_EAUTH\_AUX\_DATA 526
- LSF\_EAUTH\_AUX\_PASS 526
- LSF\_EAUTH\_CLIENT 526
- LSF\_EAUTH\_SERVER 526
- LSF\_EAUTH\_UID 527
- LSF\_EXECUTE\_DOMAIN 527
- LSF\_INTERACTIVE\_STDERR 527
- LSF\_INVOKE\_CMD 527
- LSF\_JOB\_STARTER 528
- LSF\_LD\_LIBRARY\_PATH 529
- LSF\_LIM\_API\_NTRIES 529
- LSF\_LIM\_DEBUG 530
- LSF\_LOGDIR 530
- LSF\_MASTER 530
- LSF\_NIOS\_DEBUG 530
- LSF\_NIOS\_DIE\_CMD 530
- LSF\_NIOS\_ERR\_LOGDIR 530
- LSF\_NIOS\_IGNORE\_SIGWINDOW 531
- LSF\_NIOS\_PEND\_TIMEOUT 531
- LSF\_NIOS\_PORT\_RANGE 532
- LSF\_RESOURCES 532
- LSF\_TS\_LOGON\_TIME 532
- LSF\_USE\_HOSTEQUIV 533
- LSF\_USER\_DOMAIN 533
- EP\_BACKUP
  - install.config file 23
  - slave.config file 484
- EQUIV
  - Isf.cluster file 313
- error logs
  - EGO\_LOG\_MASK parameter 374
- EVENT\_ADRSV\_FINISH record
  - Isb.acct 38
- EVENT\_STREAM\_FILE
  - Isb.params file 145
- EVENT\_UPDATE\_INTERVAL
  - Isb.params file 145
- EXCLUSIVE
  - Isb.queues file 195
- exclusive resource 308
- EXINTERVAL
  - Isf.cluster file 296
- EXIT\_RATE
  - Isb.hosts file 112
- EXIT\_RATE\_TYPE
  - Isb.params file 146
- EXTEND\_JOB\_EXCEPTION\_NOTIFY
  - Isb.params file 146
- external authentication
  - LSF\_AUTH parameter 366
- F**
- FAIRSHARE
  - Isb.queues file 196
- FAIRSHARE\_ADJUSTMENT\_FACTOR
  - Isb.params file 147
  - Isb.queues file 197
- FAIRSHARE\_QUEUES
  - Isb.queues file 197
- Feature section
  - Isf.licensescheduler file description 446
- FeatureGroup
  - Isf.licensescheduler 459
- FILELIMIT
  - Isb.applications file 50
  - Isb.queues file 198
- files
  - adding default system lists 478
  - removing default system lists 478
  - viewing task lists 478
- FLEX\_NAME
  - Isf.licensescheduler file Feature section 447
- FLOAT\_CLIENTS
  - Isf.cluster file 296

FLOAT\_CLIENTS\_ADDR\_RANGE

Isf.cluster file 297

## G

GLOBAL\_EXIT\_RATE

Isb.params file 142, 147

GOALS

Isb.serviceclasses file 277

GROUP

Isf.licensescheduler file Feature section 451

Isf.licensescheduler file ProjectGroup section 460

GROUP\_ADMIN

Isb.hosts file 117

GROUP\_DISTRIBUTION

Isf.licensescheduler file Feature section 451

GROUP\_MEMBER

Isb.hosts file 116

Isb.users file 284

GROUP\_NAME

Isb.hosts file 115

Isb.users file 284

GRP\_ADD record

Isb.events 105

GRP\_MOD record

Isb.events 106

GuaranteedResourcePool section

Isb.resources 257

## H

hierarchical fairshare user groups 286

HIST\_HOURS

Isb.params file 147

Isb.queues file 198

HJOB\_LIMIT

Isb.queues file 199

host groups

EGO enabled 200

host management

daemon clean up 318

host models

automatic detection 467

host types

automatic detection 467

HOST\_CTRL record

Isb.events 86

HOST\_INACTIVITY\_LIMIT

Isf.cluster file 299

HOST\_NAME

Isb.hosts file 111

HOST\_RUSAGE

LSF HPC extensions parameter 386

HostExport section

Isb.resources 261

HOSTNAME

Isf.cluster file 307

hosts

exclusive resource 308

HOSTS

Isb.hosts file 120

Isb.queues file 199

Isb.resources file GuaranteedResourcePool section 259

Isb.resources file Limit section 242

Isb.resources file ResourceReservation section 266

Isf.licensescheduler file Parameters section 438

hosts file 18

HPART\_NAME

Isb.hosts file 120

## I

identd 366

identification daemon authentication

LSF\_AUTH parameter 366

identification daemons 366

IGNORE\_DEADLINE

Isb.queues file 202

IMPT\_JOBKLG

Isb.queues file 202

INCREASING

Isf.shared file 469

install.config file

description 21

INTERACTIVE

Isb.queues file 203

INTERRUPTIBLE\_BACKFILL

Isb.queues file 203

INTERVAL

Isf.shared file 469

io

Isb.hosts file 114

Isb.queues file 209

IPv6

dual-stack hosts 378

enable 383

- example 18, 20
- in FLOAT\_CLIENTS\_ADDR\_RANGE 298
- in LSF\_HOST\_ADDR\_RANGE 302
- IRIX ULDB (User Limits Database)
  - description 430
  - jlimit.in file 430
- it
  - lsb.hosts file 114
  - lsb.queues file 209
- J**
- JL/P
  - lsb.users file 288
- JL/U
  - lsb.hosts file 113
- jlimit.in file
  - IRIX ULDB 430
- job ID
  - limit 161
  - rollover 161
  - sequencing 161
- job migration
  - absolute job priority scheduling 189
- JOB\_ACCEPT record
  - lsb.events 81
- JOB\_ACCEPT\_INTERVAL
  - lsb.params file 148
  - lsb.queues file 204
- JOB\_ACTION\_WARNING\_TIME
  - lsb.queues file 204
- JOB\_ATTA\_DATA record
  - lsb.events 98
- JOB\_ATTA\_DIR
  - lsb.params file 148
- JOB\_CHUNK record
  - lsb.events 99
- JOB\_CLEAN record
  - lsb.events 96
- JOB\_CONTROLS
  - lsb.queues file 205
- JOB\_DEP\_LAST\_SUB
  - lsb.params file 149
- JOB\_EXECUTE record
  - lsb.events 95
- JOB\_EXIT\_RATE\_DURATION
  - lsb.params file 149
- JOB\_EXT\_MSG record
  - lsb.events 98
- JOB\_FINISH record
  - lsb.acct 32
- JOB\_FORCE record
  - lsb.events 104
- JOB\_FORWARD record
  - lsb.events 80
- JOB\_GROUP\_CLEAN
  - lsb.params file 150
- JOB\_IDLE
  - lsb.queues file 206
- JOB\_INCLUDE\_POSTPROC parameter
  - lsb.applications 49
  - lsb.params 150
- JOB\_MODIFY2 record
  - lsb.events 90
- JOB\_MOVE record
  - lsb.events 85
- JOB\_NEW record
  - lsb.events 75
- JOB\_OVERRUN
  - lsb.queues file 207
- JOB\_POSITION\_CONTROL\_BY\_ADMIN
  - lsb.params file 151
- JOB\_POSTPROC\_TIMEOUT parameter
  - lsb.applications 49
  - lsb.params 151
- JOB\_PRIORITY\_OVER\_TIME
  - lsb.params file 152
- JOB\_QUEUE record
  - lsb.events 96
- JOB\_RUNLIMIT\_RATIO
  - lsb.params file 152
- JOB\_SCHEDULING\_INTERVAL
  - lsb.params file 153
- JOB\_SIGACT record
  - lsb.events 88
- JOB\_SIGNAL record
  - lsb.events 94
- JOB\_SPOOL\_DIR
  - lsb.params file
    - description 154
- JOB\_START record
  - lsb.events 81
- JOB\_START\_ACCEPT record
  - lsb.events 82
- JOB\_STARTER
  - lsb.applications file 50

- Isb.queues file 207
- JOB\_STATUS record
  - Isb.events 83
- JOB\_SWITCH record
  - Isb.events 84
- JOB\_TERMINATE\_INTERVAL
  - Isb.params file 155
- JOB\_UNDERRUN
  - Isb.queues file 208
- JOB\_WARNING\_ACTION
  - Isb.queues file 208
- jobs
  - allowing preemption of 218
  - preempting by run time 176
- JOBSS
  - Isb.resources file Limit section 243

## L

- LIB\_RECVTIMEOUT
  - Isf.licensescheduler file Parameters section 439
- LIC\_COLLECTOR
  - Isf.licensescheduler file ServiceDomain section 445
- LIC\_SERVERS
  - Isf.licensescheduler file ServiceDomain section 445
- LICENSE
  - Isb.resources file Limit section 244
- lim.acct file 30
- LIMIT
  - Isf.licensescheduler file ProjectGroup section 461
- limit number of hosts 344
- Limit section
  - Isb.resources 239
- limits
  - enforced with overlapping members 144
  - job ID 161
- live reconfiguration
  - directory 400
  - LSF\_LIVE\_CONFDIR 400
- LM\_LICENSE\_FILE variable 500
- LM\_REMOVE\_INTERVAL
  - Isf.licensescheduler file Features section 457
  - Isf.licensescheduler file Parameters section 439
- LM\_STAT\_INTERVAL
  - Isf.licensescheduler file Parameters section 439
  - Isf.licensescheduler file ServiceDomain section 445
- LMSTAT\_PATH
  - Isf.licensescheduler file Parameters section 439
- load\_index
  - Isb.hosts file 114
  - Isb.queues file 208
- LOAD\_INDEX record
  - Isb.events 88
- LOAN\_POLICIES
  - Isb.resources file GuaranteedResourcePool section 260
- LOCAL
  - Isb.users file 289
- local tasks in task files 479
- LOCAL\_MAX\_PREEXEC\_RETRY
  - Isb.applications file 51
  - Isb.params file 155
  - Isb.queues file 209
- LOCAL\_TO
  - Isf.licensescheduler file Features section 452
- LOCATION
  - Isf.cluster file 311
- log files
  - nios.log.host\_name 411
- ls
  - Isb.hosts file 114
  - Isb.queues file 209
- LS\_ADMIN
  - setup.config file 480
- LS\_DEBUG\_BLD
  - Isf.licensescheduler file Parameters section 440
- LS\_ENABLE\_MAX\_PREEMPT
  - Isf.licensescheduler file Parameters section 440
- LS\_EXEC\_T variable 501
- LS\_FEATURE\_PERCENTAGE
  - Isf.licensescheduler file Features section 454
- LS\_HOSTS
  - setup.config file 481
- LS\_JOBPID variable 501
- LS\_LICENSE\_FILE
  - setup.config file 481
- LS\_LICENSE\_SERVER\_feature variable 501
- LS\_LMSTAT\_PATH
  - setup.config file 482
- LS\_LOG\_MASK
  - Isf.licensescheduler file Parameters section 441
- LS\_MAX\_STREAM\_FILE\_NUMBER
  - Isf.licensescheduler file Parameters section 441
- LS\_MAX\_STREAM\_SIZE
  - Isf.licensescheduler file Parameters section 442
- LS\_MAX\_TASKMAN\_PREEMPTS
  - Isf.licensescheduler file Parameters section 442

LS\_MAX\_TASKMAN\_SESSIONS  
     Isf.licensescheduler file Parameters section 442

LS\_PREEMPT\_PEER  
     Isf.licensescheduler file Parameters section 443

LS\_STREAM\_FILE  
     Isf.licensescheduler file Parameters section 442

LS\_SUBCWD variable 502

LS\_TOP  
     setup.config file 482

LS\_WAIT\_TO\_PREEMPT  
     Isf.licensescheduler file 458

LSB\_API\_CONNTIMEOUT  
     Isf.conf file 319

LSB\_API\_RECVTIMEOUT  
     Isf.conf file 319

LSB\_API\_VERBOSE  
     Isf.conf file 320

LSB\_BJOBS\_CONSISTENT\_EXIT\_CODE  
     Isf.conf file 320

LSB\_BLOCK\_JOBINFO\_TIMEOUT  
     Isf.conf file 322

LSB\_BPEEK\_REMOTE\_OUTPUT  
     Isf.conf file 322

LSB\_CHKPNT\_DIR variable 502

LSB\_CHUNK\_RUSAGE  
     Isf.conf file 323

LSB\_CMD\_LOG\_MASK  
     Isf.conf file 323

LSB\_CMD\_LOGDIR  
     Isf.conf file 324

LSB\_CONFDIR  
     Isf.conf file 325

LSB\_CPUSET\_BESTCPUS  
     Isf.conf file 325

LSB\_CRDIR  
     Isf.conf file 325

LSB\_DEBUG  
     Isf.conf file 326  
     variable 502

LSB\_DEBUG\_CMD  
     Isf.conf file 326  
     variable 503

LSB\_DEBUG\_MBD  
     Isf.conf file 328  
     variable 503

LSB\_DEBUG\_NQS  
     Isf.conf file 329  
     variable 503

LSB\_DEBUG\_SBD  
     Isf.conf file 329  
     variable 503

LSB\_DEBUG\_SCH  
     Isf.conf file 330  
     variable 503

LSB\_DEFAULT\_JOBGROUP variable 503

LSB\_DEFAULTPROJECT variable 504

LSB\_DEFAULTQUEUE variable 505

LSB\_DISABLE\_LIMLOCK\_EXCL  
     Isf.conf file 177, 331

LSB\_DISABLE\_RERUN\_POST\_EXEC  
     Isf.conf file 331

LSB\_DISPLAY\_YEAR  
     Isf.conf file 332

LSB\_DJOB\_COMMFAIL\_ACTION variable 518

LSB\_DJOB\_ENV\_SCRIPT variable 518

LSB\_DJOB\_NUMPROC variable 505

LSB\_ECHKPNT\_KEEP\_OUTPUT  
     Isf.conf file 332

LSB\_ECHKPNT\_KEEP\_OUTPUT variable 506

LSB\_ECHKPNT\_METHOD  
     Isf.conf file 332

LSB\_ECHKPNT\_METHOD variable 506

LSB\_ECHKPNT\_METHOD\_DIR  
     Isf.conf file 333

LSB\_ECHKPNT\_METHOD\_DIR variable 506

LSB\_ERESTART\_USRCMD variable 506

LSB\_ESUB\_METHOD  
     Isf.conf file 332, 334

LSB\_EXEC\_RUSAGE variable 506

LSB\_EXECHOSTS variable 507

LSB\_EXIT\_IF\_CWD\_NOTEXIST variable 507

LSB\_EXIT\_PRE\_ABORT variable 507

LSB\_EXIT\_REQUEUE variable 508

LSB\_FRAMES variable 508

LSB\_HCLOSE\_BY\_RES  
     LSF HPC extensions parameter 387

LSB\_HJOB\_PER\_SESSION  
     Isf.conf file 334

LSB\_HOSTS variable 509

LSB\_INDEX\_BY\_JOB  
     Isf.conf file 335

LSB\_INTERACT\_MSG\_ENH  
     Isf.conf file 335

LSB\_INTERACT\_MSG\_INTVAL  
     Isf.conf file 336

LSB\_INTERACTIVE variable 509

LSB\_JOB\_CPULIMIT  
     Isf.conf file 336  
 LSB\_JOB\_INCLUDE\_POSTPROC variable 510  
 LSB\_JOB\_MEMLIMIT  
     Isf.conf file 337  
 LSB\_JOB\_OUTPUT\_LOGGING  
     Isf.conf file 339  
 LSB\_JOBEXIT\_INFO variable 510  
 LSB\_JOBEXIT\_STAT variable 511  
 LSB\_JOBFILENAME variable 511  
 LSB\_JOBGROUP variable 511  
 LSB\_JOBID variable 512  
 LSB\_JOBID\_DISP\_LENGTH  
     Isf.conf file 339  
 LSB\_JOBINDEX variable 512  
 LSB\_JOBINDEX\_STEP variable 513  
 LSB\_JOBNAME variable 513  
 LSB\_JOBPEND variable 514  
 LSB\_JOBPGIDS variable 514  
 LSB\_JOBPIIDS variable 515  
 LSB\_KEEP\_SYSDEF\_RLIMIT  
     Isf.conf file 340  
 LSB\_LIMLOCK\_EXCLUSIVE parameter 331  
 LSB\_LOAD\_TO\_SERVER\_HOSTS  
     Isf.conf file 340  
 LSB\_LOCALDIR  
     Isf.conf file 340  
 LSB\_MAIL\_FROM\_DOMAIN  
     Isf.conf file 343  
 LSB\_MAILPROG  
     Isf.conf file 341  
 LSB\_MAILSERVER  
     Isf.conf file 342  
 LSB\_MAILSIZE variable 515  
 LSB\_MAILSIZE\_LIMIT  
     Isf.conf file 343  
 LSB\_MAILTO  
     Isf.conf file 344  
 LSB\_MAX\_ASKED\_HOSTS\_NUMBER  
     Isb.params 344  
 LSB\_MAX\_JOB\_DISPATCH\_PER\_SESSION  
     Isf.conf file 345  
 LSB\_MAX\_NQS\_QUEUES  
     Isf.conf file 347  
 LSB\_MAX\_PACK\_JOBS  
     Isf.conf file 346  
 LSB\_MAX\_PROBE\_SBD  
     Isf.conf file 346  
 LSB\_MBD\_BUSY\_MSG  
     Isf.conf file 347  
 LSB\_MBD\_CONNECT\_FAIL\_MSG  
     Isf.conf file 348  
 LSB\_MBD\_DOWN\_MSG  
     Isf.conf file 348  
 LSB\_MBD\_MAX\_SIG\_COUNT 349  
 LSB\_MBD\_PORT  
     Isf.conf file 349, 400  
 LSB\_MC\_CHKPNT\_RERUN  
     Isf.conf file 349  
 LSB\_MC\_INITFAIL\_MAIL  
     Isf.conf file 349  
 LSB\_MC\_INITFAIL\_RETRY  
     Isf.conf file 350  
 LSB\_MCPU\_HOSTS variable 516  
 LSB\_MEMLIMIT\_ENFORCE  
     Isf.conf file 350  
 LSB\_MIG2PEND  
     Isf.conf file 351  
 LSB\_MIXED\_PATH\_DELIMITER  
     Isf.conf 351  
 LSB\_MOD\_ALL\_JOBS  
     Isf.conf file 352  
 LSB\_NCPU\_ENFORCE  
     Isf.conf file 353  
 LSB\_NQS\_PORT  
     Isf.conf file 353  
 LSB\_NQS\_PORT variable 516  
 LSB\_NTRIES environment variable 162  
 LSB\_NTRIES variable 517  
 LSB\_NUM\_NIOS\_CALLBACK\_THREADS  
     Isf.conf file 354  
 LSB\_OLD\_JOBID variable 517  
 LSB\_OUTPUT\_TARGETFAILED variable 517  
 LSB\_PACK\_MESUB  
     Isf.conf file 354  
 LSB\_PACK\_SKIP\_ERROR Isf.conf file 354  
 LSB\_PRE\_POST\_EXEC\_USER  
     Isf.sudoers file 473  
 LSB\_PSET\_BIND\_DEFAULT  
     Isf.conf file 355  
 LSB\_QUERY\_PORT  
     Isf.conf file 355  
 LSB\_QUERY\_PORT parameter in Isf.conf 157  
 LSB\_QUEUE variable 519  
 LSB\_REMOTEINDEX variable 519  
 LSB\_REMOTEJID variable 519

LSB\_REQUEUE\_TO\_BOTTOM  
     Isf.conf file 356  
 LSB\_RESTART variable 520  
 LSB\_RESTART\_PGID variable 520  
 LSB\_RESTART\_PID variable 521  
 LSB\_RLA\_PORT  
     Isf.conf file 357  
 LSB\_RLA\_UPDATE  
     Isf.conf file 357  
 LSB\_RLA\_WORKDIR  
     Isf.conf file 357  
 LSB\_RTASK\_GONE\_ACTION variable 521  
 LSB\_SACCT\_ONE\_UG  
     Isf.conf 358  
 LSB\_SBD\_PORT  
     Isf.conf file 358, 400  
 LSB\_SET\_TMPDIR  
     Isf.conf file 358  
 LSB\_SHAREDIR  
     Isf.conf file 358  
 LSB\_SHORT\_HOSTLIST  
     Isf.conf file 359  
 LSB\_SIGSTOP  
     Isf.conf file 359  
 LSB\_SSH\_XFORWARD\_CMD  
     Isf.conf file 360  
 LSB\_STDOUT\_DIRECT  
     Isf.conf file 360  
 LSB\_STOP\_IGNORE\_IT  
     Isf.conf file 360  
 LSB\_SUB\_CLUSTER variable 522  
 LSB\_SUB\_COMMAND\_LINE variable 522  
 LSB\_SUB\_COMMANDNAME  
     Isf.conf file 361  
 LSB\_SUB\_EXTSCHED\_PARAM variable 522  
 LSB\_SUB\_JOB\_ACTION\_WARNING\_TIME variable 522  
 LSB\_SUB\_JOB\_WARNING\_ACTION variable 523  
 LSB\_SUB\_PARM\_FILE variable 523  
 LSB\_SUBK\_SHOW\_EXEC\_HOST  
     Isf.conf file 361  
 LSB\_SUCCESS\_EXIT\_VALUES variable 523  
 LSB\_SUSP\_REASONS variable 523  
 LSB\_SUSP\_SUBREASONS variable 524  
 LSB\_SYNC\_HOST\_STAT\_LIM  
     Isb.params 156  
 LSB\_TIME\_CMD  
     Isf.conf file 362  
 LSB\_TIME\_MBD  
     Isf.conf file 362  
 LSB\_TIME\_RESERVE\_NUMJOBS  
     Isf.conf file 362  
 LSB\_TIME\_SBD  
     Isf.conf file 363  
 LSB\_TIME\_SCH  
     Isf.conf file 363  
 LSB\_UNIXGROUP  
     variable 525  
 LSB\_USER\_BIND\_CPU\_LIST  
     variable 525  
 LSB\_USER\_BIND\_JOB  
     variable 525  
 LSB\_UTMP  
     Isf.conf file 363  
 Isb.acct file 31  
 Isb.applications  
     JOB\_INCLUDE\_POSTPROC parameter 49  
     JOB\_POSTPROC\_TIMEOUT parameter 49  
     POST\_EXEC parameter 57  
     PRE\_EXEC parameter 59  
 Isb.applications file 42  
     description 41  
 Isb.events file 74  
 Isb.hosts file  
     description 111  
     time-based configuration 125  
     user group administrator 285  
 Isb.modules file 127  
 Isb.params  
     SUB\_TRY\_INTERVAL parameter 162  
 Isb.params file  
     description 132  
     time-based configuration 186  
 Isb.queues  
     POST\_EXEC parameter 216  
     PRE\_EXEC parameter 217  
 Isb.queues file  
     description 188  
     time-based configuration 238  
 Isb.resources file  
     description 239  
     time-based configuration 270  
 Isb.serviceclasses file 272  
 Isb.users file  
     description 283  
     time-based configuration 290  
 LSF\_ADD\_CLIENTS

- install.config file 24
- LSF\_ADD\_SERVERS
  - install.config file 23
- LSF\_ADMINS
  - install.config file 25
  - slave.config file 485
- LSF\_AFS\_CELLNAME
  - lsf.conf file 364
- LSF\_AM\_OPTIONS
  - lsf.conf file 364
- LSF\_API\_CONNTIMEOUT
  - lsf.conf file 365
- LSF\_API\_RECVMTIMEOUT
  - lsf.conf file 365
- LSF\_ASPLUGIN
  - lsf.conf file 366
- LSF\_AUTH
  - lsf.conf file 366
- LSF\_AUTH parameter 366
- LSF\_AUTH\_DAEMONS
  - lsf.conf file 367
- LSF\_BIND\_JOB
  - lsf.conf file 42, 367
- LSF\_BINDIR
  - csfrc.lsf and profile.lsf files 12
  - lsf.conf file 367
- LSF\_BMPLUGIN
  - lsf.conf file 368
- LSF\_CLUSTER\_NAME
  - install.config file 25
- LSF\_CMD\_LOG\_MASK
  - lsf.conf file 368
- LSF\_CMD\_LOGDIR
  - lsf.conf file 368
  - variable 525
- LSF\_CONF\_RETRY\_INT
  - lsf.conf file 369
- LSF\_CONF\_RETRY\_MAX
  - lsf.conf file 370
- LSF\_CONFDIR
  - lsf.conf file 370
- LSF\_CPUSETLIB
  - lsf.conf file 371
- LSF\_CRASH\_LOG
  - lsf.conf file 371
- LSF\_DAEMON\_WRAP
  - lsf.conf file 372
- LSF\_DAEMONS\_CPUS
  - lsb.params file 371
- LSF\_DAEMONS\_CPUS parameter in ego.conf 372
- LSF\_DEBUG\_CMD
  - lsf.conf file 373
- LSF\_DEBUG\_CMD variable 525
- LSF\_DEBUG\_LIM
  - lsf.conf file 374
  - variable 525
- LSF\_DEBUG\_RES
  - lsf.conf file 375
  - variable 526
- LSF\_DHCP\_ENV
  - lsf.conf file 376
- LSF\_DISABLE\_LSRUN
  - lsf.conf file 376
- LSF\_DISPATCHER\_LOGDIR
  - lsf.conf file 377
- LSF\_DUALSTACK\_PREFER\_IPV6
  - lsf.conf file 377
- LSF\_DYNAMIC\_HOST\_TIMEOUT
  - lsf.conf file 378
- LSF\_DYNAMIC\_HOST\_WAIT\_TIME
  - install.config file 26
  - lsf.conf file 379
- LSF\_EAUTH\_AUX\_DATA variable 526
- LSF\_EAUTH\_AUX\_PASS variable 526
- LSF\_EAUTH\_CLIENT variable 526
- LSF\_EAUTH\_KEY
  - lsf.sudoers file 473
- LSF\_EAUTH\_SERVER variable 526
- LSF\_EAUTH\_UID variable 527
- LSF\_EAUTH\_USER
  - lsf.sudoers file 473
- LSF\_EEXEC\_USER
  - lsf.sudoers file 474
- LSF\_EGO\_ADMIN\_PASSWD
  - lsf.sudoers file 474
- LSF\_EGO\_ADMIN\_USER
  - lsf.sudoers file 474
- LSF\_EGO\_DAEMON\_CONTROL
  - lsf.conf file 380
- LSF\_EGO\_ENVDIR
  - lsf.conf file 381
- LSF\_ELIM\_BLOCKTIME
  - lsf.cluster file 300
- LSF\_ELIM\_DEBUG
  - lsf.cluster file 300
- LSF\_ELIM\_RESTARTS

Isf.cluster file 301	LSF_LIC_SCHED_PREEMPT_REQUEUE
LSF_ENABLE_CSA	Isf.conf file 394
Isf.conf file 381	LSF_LIC_SCHED_PREEMPT_SLOT_RELEASE
LSF_ENABLE_DUALCORE	Isf.conf file 394
Isf.conf file 382	LSF_LIC_SCHED_PREEMPT_STOP
LSF_ENABLE_EGO	Isf.conf file 395
Isf.conf file 383	LSF_LIC_SCHED_STRICT_PROJECT_NAME
LSF_ENABLE_EXTSCHEDULER	Isf.conf file 395
Isf.conf file 383	LSF_LICENSE
LSF_ENABLE_SUPPORT_IPV6	install.config file 26
Isf.conf 383	LSF_LICENSE_ACCT_PATH
LSF_ENVDIR	Isf.conf file 396
cshrc.Isf and profile.Isf files 12	LSF_LICENSE_FILE
Isf.conf file 384	Isf.conf file 396
LSF_EVENT_PROGRAM	LSF_LICENSE_MAINTENANCE_INTERVAL
Isf.conf file 384	Isf.conf file 397
LSF_EVENT_RECEIVER	LSF_LICENSE_NOTIFICATION_INTERVAL
Isf.conf file 384	Isf.conf file 398
LSF_EXECUTE_DOMAIN variable 527	LSF_LIM_API_NTRIES
LSF_HOST_ADDR_RANGE	Isf.conf file 398
Isf.cluster file 301	variable 529
LSF_HOST_CACHE_NTTL	LSF_LIM_DEBUG
Isf.conf file 385	Isf.conf file 399
LSF_HOST_CACHE_PTTL	variable 530
Isf.conf file 385	LSF_LIM_IGNORE_CHECKSUM
LSF_HPC_EXTENSIONS	Isf.conf file 399
Isf.conf file 386	LSF_LIM_PORT
LSF_HPC_PJL_LOADENV_TIMEOUT	Isf.conf file 400
Isf.conf file 390	slave.config file 485
LSF_ID_PORT	LSF_LIVE_CONFDIR
Isf.conf file 391	Isf.conf 400
LSF_INCLUDEDIR	LSF_LOAD_PLUGINS
Isf.conf file 391	Isf.sudoers file 475
LSF_INDEP	LSF_LOAD_USER_PROFILE
Isf.conf file 391	Isf.conf 401
LSF_INTERACTIVE_STDERR	LSF_LOCAL_RESOURCES
Isf.conf file 392	Isf.conf file 401
variable 527	LSF_LOG_MASK
LSF_INVOKE_CMD variable 527	Isf.conf file 402, 404
LSF_JOB_STARTER variable 528	LSF_LOGDIR
LSF_LD_LIBRARY_PATH variable 529	Isf.conf file 404, 405
LSF_LD_SECURITY	variable 530
Isf.conf 393	LSF_LSLOGIN_SSH
LSF_LIBDIR	Isf.conf file 406
cshrc.Isf and profile.Isf files 13	LSF_MACHDEP
Isf.conf file 393	Isf.conf file 407
LSF_LIC_SCHED_HOSTS	LSF_MANAGER product name
Isf.conf file 394	Isf.cluster_name.license.acct file 314

LSF\_MANDIR  
     Isf.conf file 407  
 LSF\_MASTER variable 530  
 LSF\_MASTER\_LIST  
     install.config file 27  
     Isf.conf file 408  
 LSF\_MASTER\_NSLOOKUP\_TIMEOUT  
     Isf.conf file 409  
 LSF\_MAX\_TRY\_ADD\_HOST  
     Isf.conf file 409  
 LSF\_MC\_NON\_PRIVILEGED\_PORTS  
     Isf.conf file 410  
 LSF\_MISC  
     Isf.conf file 410  
 LSF\_MONITOR\_LICENSE\_TOOL  
     Isf.conf file 410  
 LSF\_MULTICLUSTER product name  
     Isf.cluster\_name.license.acct file 315  
 LSF\_NIOS\_DEBUG  
     Isf.conf file 411  
     variable 530  
 LSF\_NIOS\_DIE\_CMD variable 530  
 LSF\_NIOS\_ERR\_LOGDIR  
     Isf.conf file 411  
     variable 530  
 LSF\_NIOS\_IGNORE\_SIGWINDOW variable 531  
 LSF\_NIOS\_JOBSTATUS\_INTERVAL  
     Isf.conf file 412  
 LSF\_NIOS\_MAX\_TASKS  
     Isf.conf file 412  
 LSF\_NIOS\_PEND\_TIMEOUT variable 531  
 LSF\_NIOS\_PORT\_RANGE variable 532  
 LSF\_NIOS\_RES\_HEARTBEAT  
     Isf.conf file 413  
 LSF\_NON\_PRIVILEGED\_PORTS  
     Isf.conf file 413  
 LSF\_PAM\_APPL\_CHKPNP 414  
     Isf.conf 414  
 LSF\_PAM\_CLEAN\_JOB\_DELAY  
     Isf.conf file 414  
 LSF\_PAM\_HOSTLIST\_USE  
     Isf.conf file 414  
 LSF\_PAM\_PLUGINDIR  
     Isf.conf file 415  
 LSF\_PAM\_USE\_ASH  
     Isf.conf file 415  
 LSF\_PASSWD\_DIR  
     Isf.conf file 415  
 LSF\_PIM\_INFODIR  
     Isf.conf file 416  
 LSF\_PIM\_LINUX\_ENHANCE  
     Isf.conf file 416  
 LSF\_PIM\_SLEEPTIME  
     Isf.conf file 417  
 LSF\_PIM\_SLEEPTIME\_UPDATE  
     Isf.conf file 417  
 LSF\_POE\_TIMEOUT\_BIND  
     Isf.conf file 418  
 LSF\_POE\_TIMEOUT\_SELECT  
     Isf.conf file 418  
 LSF\_QUIET\_INST  
     install.config file 27  
 LSF\_REMOTE\_COPY\_CMD 418  
 LSF\_RES\_ACCT  
     Isf.conf file 419  
 LSF\_RES\_ACCTDIR  
     Isf.conf file 420  
 LSF\_RES\_ACTIVE\_TIME  
     Isf.conf file 420  
 LSF\_RES\_CLIENT\_TIMEOUT  
     Isf.conf 420  
 LSF\_RES\_CONNECT\_RETRY  
     Isf.conf file 421  
 LSF\_RES\_DEBUG  
     Isf.conf file 421  
 LSF\_RES\_PORT  
     Isf.conf file 400  
 LSF\_RES\_RLIMIT\_UNLIM  
     Isf.conf file 422  
 LSF\_RES\_TIMEOUT  
     Isf.conf file 422  
 LSF\_RESOURCES variable 532  
 LSF\_ROOT\_REX  
     Isf.conf file 422  
 LSF\_RSH  
     Isf.conf file 423  
 LSF\_SECUREDIR  
     Isf.conf file 424  
 LSF\_SERVER\_HOSTS  
     Isf.conf file 424  
     slave.config file 486  
 LSF\_SERVERDIR  
     cshrc.Isf and profile.Isf files 13  
     Isf.conf file 425  
 LSF\_SHELL\_AT\_USERS  
     Isf.conf file 425

LSF\_SHIFT\_JIS\_INPUT  
     Isf.conf file 426  
 LSF\_STARTUP\_PATH  
     Isf.sudoers file 475  
 LSF\_STARTUP\_USERS  
     Isf.sudoers file 475  
 LSF\_STRICT\_CHECKING  
     Isf.conf file 426  
 LSF\_STRICT\_RESREQ  
     Isf.conf file 427  
 LSF\_STRIP\_DOMAIN  
     Isf.conf file 427  
 LSF\_TARDIR  
     install.config file 28  
     slave.config file 486  
 LSF\_TIME\_CMD  
     Isf.conf file 428  
 LSF\_TIME\_LIM  
     Isf.conf file 428  
 LSF\_TIME\_RES  
     Isf.conf file 428  
 LSF\_TMPDIR  
     Isf.conf file 429  
 LSF\_TOP  
     install.config file 28  
     slave.config file 488  
 LSF\_TS\_LOGON\_TIME  
     variable 532  
 LSF\_ULDB\_DOMAIN  
     Isf.conf file 430  
 LSF\_UNIT\_FOR\_LIMITS  
     Isf.conf file 431  
 LSF\_USE\_HOSTEQUIV  
     Isf.conf file 432  
     variable 533  
 LSF\_USER\_DOMAIN  
     Isf.conf file 433  
     variable 533  
 LSF\_USER\_DOMAIN parameter 433  
 LSF\_VPLUGIN  
     Isf.conf file 433  
 Isf.cluster file 294  
 Isf.cluster\_name.license.acct file 314  
 Isf.conf  
     LSB\_LIMLOCK\_EXCLUSIVE parameter 331  
     LSB\_MBD\_MAX\_SIG\_COUNT 349  
     LSB\_SACCT\_ONE\_UG 358  
     LSF\_AUTH parameter 366

    LSF\_USER\_DOMAIN parameter 433  
 Isf.conf file 316  
     corresponding ego.conf parameters 316  
     LSB\_DISPLAY\_YEAR 332  
     LSB\_QUERY\_PORT parameter 157  
     LSB\_SUBK\_SHOW\_EXEC\_HOST 361  
 Isf.licensescheduler file  
     time-based configuration 464  
 Isf.shared file 465  
 Isf.sudoers file 471  
 Isf.task file 477  
 Isf.task.cluster file 477

## M

mail  
     configuring on UNIX 341  
 MANDATORY\_EXTSCHEM  
     Isb.queues file 210  
 MASTER\_INACTIVITY\_LIMIT  
     Isf.cluster file 304  
 MAX\_ACCT\_ARCHIVE\_FILE  
     Isb.params file 156  
 MAX\_CONCURRENT\_JOB\_QUERY  
     Isb.params file 157  
 MAX\_EVENT\_STREAM\_FILE\_NUMBER  
     Isb.params file 157  
 MAX\_EVENT\_STREAM\_SIZE  
     Isb.params file 158  
 MAX\_HOST\_IDLE\_TIME  
     Isb.serviceclasses file 278, 279  
 MAX\_INFO\_DIRS  
     Isb.params file 158  
 MAX\_JOB\_ARRAY\_SIZE  
     Isb.params file 159  
 MAX\_JOB\_ATA\_SIZE  
     Isb.params file 159  
 MAX\_JOB\_MSG\_NUM  
     Isb.params file 159  
 MAX\_JOB\_NUM  
     Isb.params file 160  
 MAX\_JOB\_PREEMPT  
     Isb.applications file 51  
     Isb.params file 160  
     Isb.queues file 210  
 MAX\_JOB\_REQUEUE  
     Isb.applications file 51  
     Isb.params file 160

- lsb.queues file 210
- MAX\_JOBID
  - lsb.params file 161
- MAX\_JOBINFO\_QUERY\_PERIOD
  - lsb.params file 161
- MAX\_JOBS
  - lsb.users file 288
- MAX\_PEND\_JOBS
  - lsb.params file 162
  - lsb.users file 289
- MAX\_PREEXEC\_RETRY
  - lsb.applications file 52
  - lsb.params file 162
  - lsb.queues file 211
- MAX\_RSCHED\_TIME
  - lsb.queues file 211
- MAX\_SBD\_CONNS
  - lsb.params file 163
- MAX\_SBD\_FAIL
  - lsb.params file 163
- MAX\_SLOTS\_IN\_POOL
  - lsb.queues file 212
- MAX\_TOTAL\_TIME\_PREEMPT
  - lsb.applications file 52
  - lsb.params file 164
  - lsb.queues file 212
- MAX\_USER\_PRIORITY
  - lsb.params file 164
- maximum
  - job ID 161
- mbatchd
  - how to fix when busy 349
- MBD\_DIE record
  - lsb.events 87
- MBD\_EGO\_CONNECT\_TIMEOUT
  - lsb.params file 165
- MBD\_EGO\_READ\_TIMEOUT
  - lsb.params file 165
- MBD\_EGO\_TIME2LIVE
  - lsb.params file 165
- MBD\_QUERY\_CPUS
  - lsb.params file 165
- MBD\_REFRESH\_TIME
  - lsb.params file 166
- MBD\_SLEEP\_TIME
  - lsb.params file 167
- MBD\_START record
  - lsb.events 86
- MBD\_USE\_EGO\_MXJ
  - lsb.params file 168
- MC\_PENDING\_REASON\_PKG\_SIZE
  - lsb.params file 168
- MC\_PENDING\_REASON\_UPDATE\_INTERVAL
  - lsb.params file 168
- MC\_PLUGIN\_REMOTE\_RESOURCE
  - lsf.conf file 434
- MC\_PLUGIN\_SCHEDULE\_ENHANCE
  - lsb.params file 169
- MC\_PLUGIN\_UPDATE\_INTERVAL
  - lsb.params file 170
- MC\_RECLAIM\_DELAY
  - lsb.params file 170
- MC\_RUSAGE\_UPDATE\_INTERVAL
  - lsb.params file 170
- mem
  - lsb.hosts file 114
  - lsb.queues file 209
- MEM
  - lsb.resources file HostExport section 263
  - lsb.resources file Limit section 245
- MEMBER
  - lsb.hosts file 122
- MEMLIMIT
  - lsb.applications file 53, 54
  - lsb.queues file 212
  - per parallel task 388
  - per-job limit 337
- METHOD
  - lsb.resources file ReservationUsage section 269
- MIG
  - lsb.hosts file 113
  - lsb.queues file 55, 214
- MIG record
  - lsb.events 89
- migrated jobs
  - absolute job priority scheduling 189
- MIN\_SWITCH\_PERIOD
  - lsb.params file 171
- mixed cluster
  - specifying paths 351
- model
  - lsf.cluster file 307
- MODELNAME
  - lsf.shared file 467
- MXJ
  - lsb.hosts file 114

## N

### NAME

- lsb.hosts file 122
- lsb.resources file GuaranteedResourcePool section 258
- lsb.resources file Limit section 246
- lsb.resources file ResourceReservation section 266
- lsb.resources file SharedResourceExport section 264
- lsb.serviceclasses file 279
- lsf.licensescheduler file Feature section 446
- lsf.licensescheduler file ServiceDomain section 444

### nd

- lsf.cluster file 307

### NEW\_JOB\_SCHED\_DELAY

- lsb.queues file 214

### NHOSTS

- lsb.resources file HostExport section 262

### NICE

- lsb.applications file 55
- lsb.queues file 214

### NINSTANCES

- lsb.resources file SharedResourceExport section 264

### NIOS

- standard message format 393

### nios.log.host\_name

- 411

### NO\_PREEMPT\_FINISH\_TIME

- lsb.applications file 56

### NO\_PREEMPT\_INTERVAL

- lsb.applications file 56
- lsb.params file 172
- lsb.queues file 215

### NO\_PREEMPT\_RUN\_TIME

- lsb.applications file 57
- lsb.params file 173

### NON\_SHARED

- lsf.licensescheduler file ProjectGroup section 461

### NON\_SHARED\_DISTRIBUTION

- lsf.licensescheduler file Feature section 454

### NQS\_QUEUES

- lsb.queues file 215

### NQS\_QUEUES\_FLAGS

- lsb.params file 173

### NQS\_REQUESTS\_FLAGS

- lsb.params file 174

## O

### OK license usage status

- bld.license.acct file 7

- lsf.cluster\_name.license.acct file 315

### OS types

- automatic detection 318

### OVERUSE license usage status

- bld.license.acct file 7
- lsf.cluster\_name.license.acct file 315

### OWNERSHIP

- lsf.licensescheduler file ProjectGroup section 460

## P

### parallel jobs

- optimizing preemption of 177

### PARALLEL\_SCHED\_BY\_SLOT

- 174

### Parameters section

- lsf.licensescheduler file
- description 436

### PATCH\_BACKUP\_DIR

- install.config file 28

### PATCH\_HISTORY\_DIR

- install.config file 29

### PEND\_REASON\_MAX\_JOBS

- 174

### PEND\_REASON\_UPDATE\_INTERVAL

- 175

### PER\_HOST

- lsb.resources file HostExport section 261

- lsb.resources file Limit section 247

### PER\_PROJECT

- lsb.resources file Limit section 247, 248

### PER\_QUEUE

- lsb.resources file Limit section 248

### PER\_USER

- lsb.resources file Limit section 249

### PERSISTENT\_HOST\_ORDER

- lsb.applications 57

### pg

- lsb.hosts file 114

- lsb.queues file 209

### PG\_SUSP\_IT

- lsb.params file 175

### PJOB\_LIMIT

- lsb.queues file 216

### PORT

- lsf.licensescheduler file Parameters section 443

### POST\_EXEC

- lsb.applications file

- bsub -Ep 57

- lsb.queues file

- bsub -Ep 57
- POST\_EXEC parameter
  - lsb.applications 57
  - lsb.queues 216
- PRE\_EXEC
  - lsb.applications file 59
  - lsb.queues file 217
- PRE\_EXEC parameter
  - lsb.applications 59
  - lsb.queues 217
- PRE\_EXEC\_START record
  - lsb.events 103
- pre- and post-execution processing
  - application level
    - POST\_EXEC parameter 57
    - PRE\_EXEC parameter 59
  - JOB\_INCLUDE\_POSTPROC parameter 49, 150
  - JOB\_POSTPROC\_TIMEOUT parameter 49, 151
  - queue level
    - POST\_EXEC parameter 216
    - PRE\_EXEC parameter 217
- PREEMPT\_FINISH\_TIME
  - lsb.params file 172
- PREEMPT\_FOR
  - lsb.params file 176
- PREEMPT\_JOBTYPE
  - lsb.params file 177
- PREEMPT\_LSF
  - lsf.licensescheduler file Feature section 455
- PREEMPT\_RESERVE
  - lsf.licensescheduler file Feature section 456
- preemptable queue
  - defining 218
- PREEMPTABLE\_RESOURCES
  - lsb.params file 178
- preemption
  - jobs by run time 176
  - of parallel jobs 177
- PREEMPTION
  - lsb.queues file 218
- PREEMPTION\_WAIT\_TIME
  - lsb.params file 178
- preemptive queue
  - defining 218
- preemptive scheduling
  - backfill jobs 177
  - enabling 218
  - exclusive jobs 177
  - jobs by run time 176
- LSB\_LIMLOCK\_EXCLUSIVE parameter 331
- per-host job slot limit for users and user groups 176
- per-processor job slot limit for a user 177
- per-processor job slot limit for user groups 176
- PREEMPT\_FOR parameter 176
- PREEMPT\_JOBTYPE parameter 177
- total job slot limit for user groups 176
- PREEXEC\_EXCLUDE\_HOST\_EXIT\_VALUES
  - lsb.params file 175
- priority
  - queues 220
- PRIORITY
  - lsb.queues file 220
  - lsb.serviceclasses file 279
  - lsf.licensescheduler file ProjectGroup section 461
  - lsf.licensescheduler file Projects section 463
- privileged ports authentication
  - LSF\_AUTH parameter 366
- PROBE\_TIMEOUT
  - lsf.cluster file 304
- PROCESSLIMIT
  - lsb.applications file 59
  - lsb.queues file 221
- PROCLIMIT
  - lsb.applications file 60
  - lsb.queues file 221
- PRODUCTS
  - lsf.cluster file 304
- profile.lsf file
  - description 9
  - setting the LSF environment 10
- ProjectGroup section
  - lsf.licensescheduler file
  - description 459
- PROJECTS
  - lsb.resources file Limit section 244, 250
  - lsf.licensescheduler file Projects section 463
- Projects section
  - lsf.licensescheduler file
  - description 462

## Q

- QJOB\_LIMIT
  - lsb.queues file 221
- QUEUE\_CTRL record
  - lsb.events 85

QUEUE\_GROUP  
     lsb.queues file 222  
 QUEUE\_NAME  
     lsb.applications file 55  
     lsb.queues file 222  
 queues  
     making preemptable 218  
     making preemptive 218  
     setting priority of 220  
 QUEUES  
     lsb.resources file GuaranteedResourcePool section 260  
     lsb.resources file Limit section 250

**R**

r15m  
     lsb.hosts file 114  
     lsb.queues file 209  
 r15s  
     lsb.hosts file 114  
     lsb.queues file 209  
 r1m  
     lsb.hosts file 114  
     lsb.queues file 209  
 RB\_PLUGIN  
     lsb.modules file 130  
 RCVJOBS\_FROM  
     lsb.queues file 222  
 RECV\_FROM  
     lsf.cluster file 313  
 RELEASE  
     lsf.shared file 470  
 REMOTE  
     lsb.users file 289  
 remote task list 477  
 remote tasks in task files 479  
 REMOTE\_MAX\_PREEXEC\_RETRY  
     lsb.applications file 60  
     lsb.params file 179  
     lsb.queues file 223  
 REQUEUE\_EXIT\_VALUES  
     lsb.applications file 61  
     lsb.queues file 223  
 RERUNNABLE  
     lsb.applications file 61  
     lsb.queues file 224  
 RES\_REQ  
     lsb.applications file 62  
     lsb.queues file 225  
 RES\_SELECT  
     lsb.resources file GuaranteedResourcePool section 259  
     lsb.resources file HostExport section 261  
 ReservationUsage section  
     lsb.resources 268  
 reserve  
     custom resources 269  
 reserve resources 269  
 RESERVE\_BY\_STARTTIME  
     LSF HPC extensions parameter 387  
 RESOURCE  
     lsb.resources file Limit section 251  
     lsb.resources file ReservationUsage section 269  
 RESOURCE\_RESERVE  
     lsb.queues file 224  
 RESOURCE\_RESERVE\_PER\_SLOT  
     lsb.params file 179  
 RESOURCENAME  
     lsf.cluster file 312  
     lsf.shared file 468  
 ResourceReservation section  
     lsb.resources 265  
 RESOURCES  
     lsf.cluster file 308  
 RESRSV\_LIMIT  
     lsb.queues file 227  
 RESUME\_COND  
     lsb.queues file 228  
 RETRY\_LIMIT  
     lsf.cluster file 305  
 REXPRI  
     lsf.cluster file 309  
 rollover  
     job IDs 161  
 RTASK\_GONE\_ACTION  
     lsb.applications file 67  
 RUN\_JOB\_FACTOR  
     lsb.params file 179  
     lsb.queues file 228  
 RUN\_TIME\_DECAY  
     lsb.params file 180  
     lsb.queues file 229  
 RUN\_TIME\_FACTOR  
     lsb.params file 180  
     lsb.queues file 229  
 RUN\_WINDOW  
     lsb.queues file 230

- RUNLIMIT
  - lsb.applications file 68
  - lsb.queues file 230
- RUNTIME
  - lsb.applications file 69
- RUNWINDOW
  - lsf.cluster file 309
- S**
- SBD\_SLEEP\_TIME
  - lsb.params file 181
- SBD\_UNREPORTED\_STATUS record
  - lsb.events 100
- SCH\_DISABLE\_PHASES
  - lsb.modules file 130
- SCH\_PLUGIN
  - lsb.modules file 128
- schmod\_advrsv scheduler plugin 129
- schmod\_aps scheduler plugin 129
- schmod\_cpuset scheduler plugin 129
- schmod\_default scheduler plugin 128
- schmod\_fairshare scheduler plugin 128
- schmod\_fcfs scheduler plugin 128
- schmod\_jobweight scheduler plugin 129
- schmod\_limit scheduler plugin 128
- schmod\_mc scheduler plugin 129
- schmod\_parallel scheduler plugin 128
- schmod\_preemption scheduler plugin 129
- schmod\_ps scheduler plugin 129
- schmod\_pset scheduler plugin 129
- schmod\_reserve scheduler plugin 129
- security
  - daemons
    - increasing 426
- sendmail program 341
- server
  - lsf.cluster file 309
- Servers
  - lsf.shared file 465
- service class
  - examples 281
- SERVICE\_DOMAINS
  - lsf.licensescheduler file Feature section 456
- ServiceDomain section
  - lsf.licensescheduler file
    - description 444
- setup.config file 480
- seven-digit job ID 161
- SharedResourceExport section
  - lsb.resources 264
- SHARES
  - lsf.licensescheduler file ProjectGroup section 460
- SHORT\_PIDLIST
  - LSF HPC extensions parameter 387
- SLA scheduling
  - service classes
    - examples 281
- SLA\_GUARANTEES\_IGNORE
  - lsb.queues file 231
- SLA\_TIMER
  - lsb.params file 182
- slave.config file 483
- SLOT\_POOL
  - lsb.queues file 231
- SLOT\_RESERVE
  - lsb.queues file 232
- SLOT\_SHARE
  - lsb.queues file 233
- SLOTS
  - lsb.resources file HostExport section 263
  - lsb.resources file Limit section 252
- SLOTS\_PER\_HOST
  - lsb.resources file GuaranteedResourcePool section 260
- SLOTS\_PER\_PROCESSOR
  - lsb.resources file Limit section 253
- SNDJOBS\_TO
  - lsb.queues file 233
- SSCHED\_ACCT\_DIR
  - lsb.params file 182
- SSCHED\_MAX\_RUNLIMIT
  - lsb.params file 182
- SSCHED\_MAX\_TASKS
  - lsb.params file 183
- SSCHED\_REQUEUE\_LIMIT
  - lsb.params file 183
- SSCHED\_RETRY\_LIMIT
  - lsb.params file 184
- SSCHED\_UPDATE\_SUMMARY\_INTERVAL
  - lsb.params file 184
- STACKLIMIT
  - lsb.applications file 69
  - lsb.queues file 233
- STOP\_COND
  - lsb.queues file 234
- STRICT\_UG\_CONTROL
  - lsb.params file 185

STRICT\_UG\_CONTROL parameter  
     lsb.params file 284, 285  
 SUB\_TRY\_INTERVAL  
     lsb.params file 185  
 SUB\_TRY\_INTERVAL parameter in lsb.params 162  
 SUCCESS\_EXIT\_VALUES  
     lsb.applications file 69  
 SUSPEND\_CONTROL  
     lsb.applications file 66, 70, 71  
 SWAP  
     lsb.resources file HostExport section 263  
 SWAPLIMIT  
     lsb.applications file 71  
     lsb.queues file 234  
     per parallel task 388  
 swp  
     lsb.hosts file 114  
     lsb.queues file 209  
 SWP  
     lsb.resources file Limit section 254  
 SYSTEM\_MAPPING\_ACCOUNT  
     lsb.params file 186

**T**

task files  
     description 477  
     format 479  
     permissions 478  
     sections 479  
 task lists  
     files 477  
     remote 477  
     viewing 478  
 TASK\_MEMLIMIT  
     LSF HPC extensions parameter 388  
 TASK\_SWAPLIMIT  
     LSF HPC extensions parameter 388  
 TERMINATE\_WHEN  
     lsb.queues file 234  
 THREADLIMIT  
     lsb.applications file 72  
     lsb.queues file 235  
 time windows  
     syntax 267  
 TIME\_WINDOW  
     lsb.resources file ResourceReservation section 267  
 time-based configuration

    lsb.hosts 125  
     lsb.params 186  
     lsb.queues 238  
     lsb.resources 270  
     lsb.users 290  
     lsf.licensescheduler 464  
 tmp  
     lsb.hosts file 114  
     lsb.queues file 209  
 TMP  
     lsb.resources file Limit section 255  
 troubleshoot  
     host groups 200  
 troubleshooting  
     cluster performance 349  
 type  
     lsf.cluster file 310  
 TYPE  
     lsb.hosts file 123  
     lsb.resources file GuaranteedResourcePool section 258  
     lsb.resources file HostExport section 263  
     lsf.shared file 469  
 TYPENAME  
     lsf.shared file 466

**U**

UJOB\_LIMIT  
     lsb.queues file 236  
 ULDB (IRIX User Limits Database)  
     description 430  
     jlimit.in file 430  
 UNFULFILL record  
     lsb.events 87  
 UNIX/Windows user account mapping  
     LSF\_EXECUTE\_DOMAIN parameter 527  
     LSF\_USER\_DOMAIN parameter 433  
 untrusted environments 426  
 USE\_PRIORITY\_IN\_POOL  
     lsb.queues file 236  
 USE\_SUSP\_SLOTS  
     lsb.params file 186  
 user group administrator 285  
 user groups  
     hierarchical fairshare 286  
 user profiles  
     loading for a job 401  
 User section

- Isb.users file 287
- USER\_NAME
  - Isb.users file 288
- USER\_SHARES
  - Isb.hosts file 121
  - Isb.users file 286
- UserGroup section
  - Isb.users file 283
- UserMap section
  - Isb.users file 289
- users
  - overlapping members 144
- USERS
  - Isb.queues file 237
  - Isb.resources file Limit section 256
  - Isb.resources file ResourceReservation section 268
  - Isb.serviceclasses file 280
- ut
  - Isb.hosts file 114
  - Isb.queues file 209

## V

variables. *See* environment variables

## W

windows

time 267

Windows and UNIX

enable mixed paths 351

WORKLOAD\_DISTRIBUTION

Isf.licensescheduler file Feature section 456

## X

XLSF\_APPDIR

Isf.conf file 434

XLSF\_UIIDIR

cshrc.Isf and profile.Isf files 14

Isf.conf file 435