Important:

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

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

<table>
<thead>
<tr>
<th>Timestamp</th>
<th>Action</th>
<th>Status</th>
<th>Version</th>
<th>License ID</th>
<th>Log ID</th>
<th>License ID</th>
<th>Log ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1107961731</td>
<td>LICENSE_SCHEDULER</td>
<td>7.0 0 0 OK</td>
<td>335a33c2bd9c9428140a61e57bd06da02b623a42</td>
<td>1107961792</td>
<td>LICENSE_SCHEDULER</td>
<td>7.0 2 0 OK</td>
<td>58e45b89f371811e6dfc6eb6f5270059a74ee31a</td>
</tr>
</tbody>
</table>

### See also

- LSF_LOGDIR in `lsf.conf`
- LSF_LICENSE_ACCT_PATH in `lsf.conf`
- `lsf.cluster_name.licenseacct`
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:

- `LSFBINDIR`
- `LSFSERVERDIR`
- `LSFLIBDIR`
- `XLSFUIDDIR`

`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`
  - `LSFSERVERDIR`
  - `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:
  ```bash
  source /usr/lsf/lsf_8/conf/cshrc.lsf
  ```
- For example, in sh, ksh, or bash:
  ```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:
  ```bash
  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/uid
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/uid
...
```

**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

\texttt{LSF\_BINDIR=dir}

Description

Directory where LSF user commands are installed.

Examples

\begin{itemize}
\item Set in csh and tcsh by cshrc.lsf:
\texttt{setenv LSF\_BINDIR /usr/lsf/lsf\_8/8.0/linux2.6-glibc2.3-x86/bin}
\item Set and exported in sh, ksh, or bash by profile.lsf:
\texttt{LSF\_BINDIR=/usr/lsf/lsf\_8/8.0/linux2.6-glibc2.3-x86/bin}
\end{itemize}

Values

\begin{itemize}
\item In cshrc.lsf for csh and tcsh:
\texttt{setenv LSF\_BINDIR $LSF\_TOP/$LSF\_VERSION/$BINARY\_TYPE/bin}
\item Set and exported in profile.lsf for sh, ksh, or bash:
\texttt{LSF\_BINDIR=$LSF\_TOP/$LSF\_VERSION/$BINARY\_TYPE/bin}
\end{itemize}

LSF_ENVDIR

Syntax

\texttt{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

\begin{itemize}
\item Set in csh and tcsh by cshrc.lsf:
\texttt{setenv LSF\_ENVDIR /usr/lsf/lsf\_8/conf}
\end{itemize}
• Set and exported in `sh`, `ksh`, or `bash` by `profile.lsf`:

  ```bash
  LSF_ENVDIR=/usr/lsf/lsf_8/conf
  ```

**Values**

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

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

• Set and exported in `profile.lsf` for `sh`, `ksh`, or `bash`:

  ```bash
  LSF_ENVDIR=/usr/lsf/lsf_8/conf
  ```

**LSF_LIBDIR**

**Syntax**

```bash
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`:

  ```bash
  setenv LSF_LIBDIR /usr/lsf/lsf_8/8.0/linux2.6-glibc2.3-x86/lib
  ```

• Set and exported in `sh`, `ksh`, or `bash` by `profile.lsf`:

  ```bash
  LSF_LIBDIR=/usr/lsf/lsf_8/8.0/linux2.6-glibc2.3-x86/lib
  ```

**Values**

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

  ```bash
  setenv LSF_LIBDIR $LSF_TOP/$LSF_VERSION/$BINARY_TYPE/lib
  ```

• Set and exported in `profile.lsf` for `sh`, `ksh`, or `bash`:

  ```bash
  LSF_LIBDIR=$LSF_TOP/$LSF_VERSION/$BINARY_TYPE/lib
  ```

**LSF_SERVERDIR**

**Syntax**

```bash
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`:

  ```bash
  setenv LSF_SERVERDIR /usr/lsf/lsf_8/8.0/linux2.6-glibc2.3-x86/etc
  ```

• Set and exported in `sh`, `ksh`, or `bash` by `profile.lsf`:

  ```bash
  LSF_SERVERDIR=/usr/lsf/lsf_8/8.0/linux2.6-glibc2.3-x86/etc
  ```
cshrc.lsf and profile.lsf

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/lsf/lsf_8/8.0/linux2.6-glibc2.3-x86/lib/uid
  ```
- Set and exported in sh, ksh, or bash by profile.lsf:
  ```
  XLSF_UIDDIR=/usr/lsf/lsf_8/8.0/linux2.6-glibc2.3-x86/lib/uid
  ```

Values

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

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

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

```bash
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`:
  ```bash
  setenv EGO_LIBDIR /usr/lsf/lsf_8/8.0/linux2.6-glibc2.3-x86/lib
  ```

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

**Values**

- In `cshrc.lsf` for `csh` and `tcsh`:
  ```bash
  setenv EGO_LIBDIR $LSF_LIBDIR
  ```

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

**EGO_LOCAL_CONFDIR**

**Syntax**

```bash
EGO_LOCAL_CONFDIR=dir
```

**Description**

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

**Examples**

- Set in `csh` and `tcsh` by `cshrc.lsf`:
  ```bash
  setenv EGO_LOCAL_CONFDIR /usr/lsf/lsf_8/conf/ego/lsf1.2.3/kernel
  ```

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

**Values**

- In `cshrc.lsf` for `csh` and `tcsh`:
  ```bash
  setenv EGO_LOCAL_CONFDIR /usr/lsf/lsf_8/conf/ego/lsf1.2.3/kernel
  ```

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

**EGO_SERVERDIR**

**Syntax**

```bash
EGO_SERVERDIR=dir
```

16 Platform LSF Configuration Reference
**Description**

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

**Examples**

- *Set in csh and tcsh by cshrc.lsf:*
  ```bash
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:*
  ```bash
  profile.lsf
  EGO_SERVERDIR=/usr/lsf/lsf_8/8.0/linux2.6-glibc2.3-x86/etc
  ```

**Values**

- *In cshrc.lsf for csh and tcsh:*
  ```bash
cshrc.lsf
setenv EGO_SERVERDIR $LSF_SERVERDIR
  ```

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

**EGO_TOP**

**Syntax**

```
EGO_TOP=dir
```

**Description**

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:*
  ```bash
cshrc.lsf
setenv EGO_TOP /usr/lsf/lsf_8
  ```

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

**Values**

- *In cshrc.lsf for csh and tcsh:*
  ```bash
cshrc.lsf
setenv EGO_TOP /usr/lsf/lsf_8
  ```

- *Set and exported in profile.lsf for sh, ksh, or bash:*
  ```bash
  profile.lsf
  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:

```plaintext
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

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 contains 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]
...
```
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_lsfinstall.tar.Z and is located in the lsf8_lsfinstall 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
EGO_DAEMON_CONTROL

Syntax

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

Description

Enables 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. 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_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 \texttt{lsfinstall} to uncomment \texttt{LSF\_EGO\_ENVDIR} and sets \texttt{LSF\_ENABLE\_EGO="Y" in lsf.conf}.

ENABLE_EGO="N" causes \texttt{lsfinstall} to comment out \texttt{LSF\_EGO\_ENVDIR} and sets \texttt{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)

\textbf{ENABLE\_HPC\_CONFIG}

Syntax

\texttt{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.)

\textbf{EP\_BACKUP}

Syntax

\texttt{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)

\textbf{LSF\_ADD\_SERVERS}

Syntax

\texttt{LSF\_ADD\_SERVERS="host\_name [ host\_name...]"}
**LSF_ADD_SERVERS**

**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.
Example

```shell
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**

```shell
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

```shell
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**

```shell
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 lsfinstall cannot find a valid license file in the default location, lsfinstall exits.

Example

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

Default

The parent directory of the current working directory. For example, if lsfinstall is running under /usr/share/lsf_distrib/lsf_lsfinstall 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.
Example

    LSF_QUI ET_INST="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/lsf_distrib"

**Default**

    The parent directory of the current working directory. For example, if `lsfinstall` is running under `/usr/share/lsf_distrib/lsf_install` the LSF_TARDIR default value is `/usr/share/lsf_distrib`.

**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/lsf"

**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_BACKUP_DIR="/usr/share/lsf/patch"
```

Default

```
LSF_TOP/patch
```
The `lim.acct` file is the log file for Load Information Manager (LIM). Produced by `lsmon`, `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 `lsmon` command line. The default list is "r15s r1m r15m ut pg ls it 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.
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
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_msgrcv (%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)
mailUser (%s)
  Name of the user to whom job related mail was sent
projectName (%s)
  LSF project name
exitStatus (%d)
  UNIX exit status of the job
maxNumProcessors (%d)
  Maximum number of processors specified for the job
loginShell (%s)
  Login shell used for the job
timeEvent (%s)
  Time event string for the job - JobScheduler only
idx (%d)
  Job array index
maxRMem (%d)
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)

`rsvId %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 rsvId field output appears last.

`slea (%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
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.

rsvid %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 rsvid 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.
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/configdir`.

### Changing lsb.applications configuration

After making any changes to `lsb.applications`, run `badmin 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 or run time estimate specified by the `-W` or `-We` 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.
**lsb.applications**

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

**Default**

Not defined

**CHKPNT_PERIOD**

**Syntax**

```
CHKPNT_PERIOD=chkpt_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.
**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 `sbatchd` and `mbatchd` and reduces scheduling overhead in `mbsched`.
- 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 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_DURATON is set in lsb.params, chunk jobs are accepted regardless of the value of CPULIMIT, RUNLIMIT or RUNTIME.

**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)). A 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 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 sbatchd 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 blaunch 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 blaunch 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 blaunch 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(DJOB_HB_INTERVAL, 10) + host_factor
```

where

```
host_factor = 0.01 * 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 blaunch distributed application framework.

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

\[
\text{max(DJOB_RU_INTERVAL, 10)} + \text{host\_factor}
\]

where

\[
\text{host\_factor} = 0.01 \times \text{number of hosts allocated for the job}
\]

Default

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

JOB_INCLUDE_POSTPROC

Syntax

JOB_INCLUDE_POSTPROC=\text{Y} | \text{N}

Description

Specifies whether LSF includes the post-execution processing of the job as part of the job. When set to \text{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
- \text{sbatchd} sends both job finish status (DONE or EXIT) and post-execution processing status (POST_DONE or POST_ERR) to \text{mbatchd} at the same time

The variable LSBJOB_INCLUDE_POSTPROC in the user environment overrides the value of JOB_INCLUDE_POSTPROC in an application profile in lsb.applications.

JOB_INCLUDE_POSTPROC in an application profile in lsb.applications overrides the value of JOB_INCLUDE_POSTPROC in lsb.params.

For SGI cpuset, if JOB_INCLUDE_POSTPROC=\text{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

\text{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=\text{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). On UNIX and Linux, it kills the entire process group of the job's pre-execution processes. On Windows, only the parent process of the pre-execution command is killed when the timeout expires, the child processes of the pre-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`). 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:
\texttt{csh -c "myjob arguments; sleep 10"}

**LOCAL_MAX_PREEXEC_RETRY**

**Syntax**

\texttt{LOCAL_MAX_PREEXEC_RETRY=integer}

**Description**

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

**Valid values**

\[0 < \text{MAX_PREEXEC_RETRY} < \text{INFINIT_INT}\]

\texttt{INFINIT_INT} is defined in \texttt{lsf.h}.

**Default**

Not defined. The number of preexec retry times is unlimited.

**MAX_JOB_PREEMPT**

**Syntax**

\texttt{MAX_JOB_PREEMPT=integer}

**Description**

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

**Valid values**

\[0 < \text{MAX_JOB_PREEMPT} < \text{INFINIT_INT}\]

\texttt{INFINIT_INT} is defined in \texttt{lsf.h}.

**Default**

Not defined. The number of preemption times is unlimited.

**MAX_JOB_REQUEUE**

**Syntax**

\texttt{MAX_JOB_REQUEUE=integer}

**Description**

The maximum number of times to requeue a job automatically.
Valid values

\[ 0 < \text{MAX\_JOB\_REQUEUE} < \text{INFINIT\_INT} \]

\text{INFINIT\_INT} is defined in \text{lsf.h}.

Default

Not defined. The number of requeue times is unlimited

**MAX\_PREEXEC\_RETRY**

Syntax

\[ \text{MAX\_PREEXEC\_RETRY} = \text{integer} \]

Description

Use \text{REMOTE\_MAX\_PREEXEC\_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

\[ 0 < \text{MAX\_PREEXEC\_RETRY} < \text{INFINIT\_INT} \]

\text{INFINIT\_INT} is defined in \text{lsf.h}.

Default

5

**MAX\_TOTAL\_TIME\_PREEMPT**

Syntax

\[ \text{MAX\_TOTAL\_TIME\_PREEMPT} = \text{integer} \]

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 \text{lsb.applications} overrides the parameter of the same name in \text{lsb.queues} and \text{lsb.params}.

Valid values

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

Default

Unlimited
MEMLIMIT

Syntax

\[ \text{MEMLIMIT} = \text{integer} \]

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 \text{LSF\_UNIT\_FOR\_LIMITS} in \text{lsf.conf} to specify a larger unit for the 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 \text{RLIMIT\_RSS} for \text{setrlimit()}.

Not supported on:

- Sun Solaris 2.x
- Windows

LSF memory limit enforcement

To enable LSF memory limit enforcement, set \text{LSB\_MEMLIMIT\_ENFORCE} in \text{lsf.conf} to \text{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 \text{LSB\_JOB\_MEMLIMIT} in \text{lsf.conf} to \text{y}. The difference between \text{LSB\_JOB\_MEMLIMIT} set to \text{y} and \text{LSB\_MEMLIMIT\_ENFORCE} set to \text{y} is that with \text{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 \text{LSB\_MEMLIMIT\_ENFORCE} set to \text{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

```yaml
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

```yaml
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 SUSP 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 *nice*(1) manual page for more details.

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

- *nice*>=0 corresponds to a priority class of *IDLE*
- *nice*<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.applications* 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.applications*), or run limit to be specified for the job (*bsub -W*, or **RUNLIMIT** in *lsb.queues*, or **RUNLIMIT** in *lsb.applications*)
NO_PREEMPT_RUN_TIME

Syntax

\[\text{NO\_PREEMPT\_RUN\_TIME} = \text{minutes} \mid \text{percentage}\]

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 (\texttt{bsub -W} or \texttt{RUNTIME} in \texttt{lsb.applications}), or run limit to be specified for the job (\texttt{bsub -W}, or \texttt{RUNLIMIT} in \texttt{lsb.queues}, or \texttt{RUNLIMIT} in \texttt{lsb.applications})

PERSISTENT_HOST_ORDER

Syntax

\[\text{PERSISTENT\_HOST\_ORDER} = \text{Y} \mid \text{yes} \mid \text{N} \mid \text{no}\]

Description

Applies when migrating parallel jobs in a multicluster environment. Setting \texttt{PERSISTENT\_HOST\_ORDER} = \texttt{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

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

POST_EXEC

Syntax

\[\text{POST\_EXEC} = \text{command}\]

Description

Enables post-execution processing at the application level. The \texttt{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 (\texttt{POST\_EXEC} in \texttt{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 (\texttt{POST\_EXEC} in \texttt{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 QUEUE_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. A 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:

\[1 \leq \text{minimum} \leq \text{default} \leq \text{maximum}\]

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_PREEXEC_RETRY**

**Syntax**

```
REMOTE_MAX_PREEXEC_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

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 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 REQUEUE_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 mbatchd 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
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 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:

<table>
<thead>
<tr>
<th>section</th>
<th>compound application and simple queue behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td>select</td>
<td>both levels satisfied; queue requirement applies to all compound terms</td>
</tr>
<tr>
<td>same</td>
<td>queue level ignored</td>
</tr>
<tr>
<td>order</td>
<td>application-level section overwrites queue-level section (if a given level is present); queue requirement (if used) applies to all compound terms</td>
</tr>
<tr>
<td>span</td>
<td></td>
</tr>
</tbody>
</table>

Platform LSF Configuration Reference
**section** | **compound application and simple queue behavior**
---|---

**rusage**
- both levels merge
- queue requirement if a job-based resource is applied to the first compound term, otherwise applies to all compound terms
- if conflicts occur the application-level section overwrites the queue-level section.

For example: if the application-level requirement is \( \text{num1} \times \{\text{rusage[R1]}\} + \text{num2} \times \{\text{rusage[R2]}\} \) and the queue-level requirement is \( \text{rusage[RQ]} \) where \( \text{RQ} \) is a job resource, the merged requirement is \( \text{num1} \times \{\text{rusage[merge(R1, RQ)]}\} + \text{num2} \times \{\text{rusage[R2]}\} \).

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 `bjobs`.

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
resource requirements.

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

For example:

<table>
<thead>
<tr>
<th>Application-level RES_REQ:</th>
<th>RES_REQ=\texttt{rusage[mem=200:lic=1]} ...</th>
</tr>
</thead>
<tbody>
<tr>
<td>For the job submission:</td>
<td>\texttt{bsub -R'\texttt{rusage[mem=100]}} ...</td>
</tr>
<tr>
<td>the resulting requirement for the job is</td>
<td>\texttt{rusage[mem=100:lic=1]}</td>
</tr>
<tr>
<td>where \texttt{mem=100} specified by the job overrides \texttt{mem=200} specified by the application profile. However, \texttt{lic=1} from application profile is kept, since job does not specify it.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Application-level RES_REQ with decay and duration defined:</th>
<th>RES_REQ=\texttt{rusage[mem=200:duration=20:decay=1]} ...</th>
</tr>
</thead>
<tbody>
<tr>
<td>For a job submission with no decay or duration:</td>
<td>\texttt{bsub -R'\texttt{rusage[mem=100]}} ...</td>
</tr>
<tr>
<td>the resulting requirement for the job is:</td>
<td>\texttt{rusage[mem=100:duration=20:decay=1]}</td>
</tr>
<tr>
<td>Application-level duration and decay are merged with the job-level specification, and \texttt{mem=100} for the job overrides \texttt{mem=200} specified by the application profile. However, \texttt{duration=20} and \texttt{decay=1} from application profile are kept, since job does not specify them.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Application-level RES_REQ with multi-phase job-level rusage:</th>
<th>RES_REQ=\texttt{rusage[mem=(200 150):duration=(10 10):decay=(1),swap=100]} ...</th>
</tr>
</thead>
<tbody>
<tr>
<td>For a multi-phase job submission:</td>
<td>\texttt{bsub -app app_name -R'\texttt{rusage[mem=(600 350):duration=(20 10):decay=(0 1)],swap=100]}} ...</td>
</tr>
<tr>
<td>the resulting requirement for the job is:</td>
<td>\texttt{rusage[mem=(600 350):duration=(20 10):decay=(0 1),swap=100]}</td>
</tr>
<tr>
<td>The job-level values for \texttt{mem}, \texttt{duration} and \texttt{decay} override the application-level values. However, \texttt{swap=100} from the application profile is kept, since the job does not specify \texttt{swap}.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Application-level RES_REQ with multi-phase application-level rusage:</th>
<th>RES_REQ=\texttt{rusage[mem=(200 150):duration=(10 10):decay=(1]}} ...</th>
</tr>
</thead>
<tbody>
<tr>
<td>For a job submission:</td>
<td>\texttt{bsub -app app_name -R'\texttt{rusage[mem=200:duration=15:decay=0]}} ...</td>
</tr>
<tr>
<td>the resulting requirement for the job is:</td>
<td>\texttt{rusage[mem=200:duration=15:decay=0]}</td>
</tr>
<tr>
<td>Job-level values override the application-level multi-phase \texttt{rusage} string.</td>
<td></td>
</tr>
</tbody>
</table>

64 Platform LSF Configuration Reference
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 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 bresize 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 bresize 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 = bresume`. 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_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, 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_TASKEXIT] [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_TASKEXIT
  - 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

```bash
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. A 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 lslinfo (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.
**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**

Unlimited

**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 (M B, 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=bstop. 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_JOBPIDS — 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_SBP_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_JOBPIDS; 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 PROCESSLIMIT 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=kill. This causes a deadlock between the signal and the action.
  - CHKPT 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 `limits.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_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)
  Soft file size limit (%d), see getrlimit(2)
  Soft data segment size limit (%d), see getrlimit(2)
  Soft stack segment size limit (%d), see getrlimit(2)
  Soft core file size limit (%d), see getrlimit(2)
  Soft memory size limit (%d), see getrlimit(2)
  Reserved (%d)
  Reserved (%d)
  Reserved (%d)
  Soft run time limit (%d), see getrlimit(2)
  Reserved (%d)
hostSpec (%s)
  Model or host name for normalizing CPU time and run time
hostFactor (%f)
CPU factor of the above host

umask (%d)
File creation mask for this job

queue (%s)
Name of job queue to which the job was submitted

resReq (%s)
Resource requirements

fromHost (%s)
Submission host name

cwd (%s)
Current working directory (up to 4094 characters for UNIX or 255 characters for Windows)

chkpntDir (%s)
Checkpoint 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)

subHomeDir (%s)
Submitter's home directory

jobFile (%s)
Job file name

numAskedHosts (%d)
Number of candidate host names

askedHosts (%s)
List of names of candidate hosts for job dispatching

dependCond (%s)
Job dependency condition

preExecCmd (%s)
Job pre-execution command

jobName (%s)
Job name (up to 4094 characters)

**command** (%s)

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

**nxf** (%d)

Number of files to transfer (%d)

**xf** (%s)

List of file transfer specifications

**mailUser** (%s)

Mail user name

**projectName** (%s)

Project name

**niosPort** (%d)

Callback port if batch interactive job

**maxNumProcessors** (%d)

Maximum number of processors

**schedHostType** (%s)

Execution host type

**loginShell** (%s)

Login shell

**timeEvent** (%d)

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

**userGroup** (%s)

User group

**exceptList** (%s)

Exception handlers for the job

**options2** (%d)

Bit flags for job processing

**idx** (%d)

Job array index

**inFileSpool** (%s)

Spool input file (up to 4094 characters for UNIX or 255 characters for Windows)

**commandSpool** (%s)

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

rsvid %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_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

- **jobld (%d)**
  - Job ID

- **jStatus (%d)**
  - New status, see `<lsf/lsbatch.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 `<lsf/lsbatch.h>`

- **subreasons (%d)**
  - Pending or suspended subreason code, see `<lsf/lsbatch.h>`

- **cpuTime (%f)**
  - CPU time consumed so far

- **endTime (%d)**
  - Job completion time

- **ru (%d)**
  - Resource usage flag

- **lsfRusage (%)s**
  - Resource usage statistics, see `<lsf/lsf.h>`

- **exitStatus (%d)**
  - Exit status of the job, see `<lsf/lsbatch.h>`

- **idx (%d)**
  - Job array index

- **exitInfo (%d)**
Job termination reason, see `<lsf/lsbatch.h>`

duration4PreemptBackfill
How long a backfilled job can run; used for preemption backfill jobs

numHostRusage(%d)
For a `jStatus` ofJOB_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 ls.conf.

hostRusage
For a `jStatus` ofJOB_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 ls.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
queue (%s)
    Target queue name
idx (%d)
    Job array index
userName (%s)
    Name of the job submitter

JOB_MOVE
    A job moved toward the top or bottom of its queue (b bot or b top). 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
position (%d)
    Position number
base (%d)
    Operation code, (TO_TOP or TO_BOTTOM), see <lsf/lsbatch.h>
idx (%d)
    Job array index
userName (%s)
    Name of the job submitter

QUEUE_CTRL
    A job queue has been altered. 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/lsbatch.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 badmin queue control commands qclose, qopen, qact, and qinact

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/lsbatch.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 badmin 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 mbatchd 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 mbatchd

- **ctrlComments (%s)**
  Administrator comment text from the -C option of badmin mbdr restart

**UNFULFILL**
Actions that were not taken because the mbatchd was unable to contact the sbatchd 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 \texttt{mbatchd} has switched the job to a new queue, but the \texttt{sbatchd} has not been informed of the switch

\texttt{sig}\ (%d)

Signal: this signal has not been sent to the job

\texttt{sig1}\ (%d)

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

\texttt{sig1Flags}\ (%d)

Checkpoint flags, see \texttt{<lsf/lsbatch.h>}

\texttt{chkPeriod}\ (%d)

New checkpoint period for job

\texttt{notModified}\ (%s)

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

\texttt{idx}\ (%d)

Job array index

\textbf{LOAD\_INDEX}

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

\texttt{Version number}\ (%s)

The version number

\texttt{Event time}\ (%d)

The time of the event

\texttt{nIdx}\ (%d)

Number of index names

\texttt{name}\ (%s)

List of index names

\textbf{JOB\_SIGACT}

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

\texttt{Version number}\ (%s)

The version number

\texttt{Event time}\ (%d)

The time of the event

\texttt{jobId}\ (%d)

Job ID
period (%d)
  Action period

pid (%d)
  Process ID of the child `sbatchd` that initiated the action

jstatus (%d)
  Job status

reasons (%d)
  Job pending reasons

flags (%d)
  Action flags, see `<lsf/lsbatch.h>`

actStatus (%d)
  Action status:
  1: Action started
  2: One action preempted other actions
  3: Action succeeded
  4: Action Failed

signalSymbol (%s)
  Action name, accompanied by actFlags

idx (%d)
  Job array index

MIG
  A job has been migrated (`bmig`). 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

  numAskedHosts (%d)
    Number of candidate hosts for migration

  askedHosts (%s)
    List of names of candidate hosts

  userId (%d)
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 `mbatchd` 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)
Reserved (%d)

**rLimits**
Soft run time limit (%d), see [getrlimit(2)]

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
cwd (%s)
  Current working directory (up to 4094 characters for UNIX or 255 characters for Windows)

preExecCmd (%s)
  Job pre-execution command

mailUser (%s)
  Mail user name

projectName (%s)
  Project name

niosPort (%d)
  Callback port if batch interactive job

maxNumProcessors (%d)
  Maximum number of processors. The value 2147483646 means the maximum number of processors is undefined.

loginShell (%s)
  Login shell

schedHostType (%s)
  Execution host type

userGroup (%s)
  User group

exceptList (%s)
  Exception handlers for the job

delOptions2 (%d)
  Delete options for the options2 field

inFileSpool (%s)
  Spool input file (up to 4094 characters for UNIX or 255 characters for Windows)

commandSpool (%s)
  Spool command file (up to 4094 characters for UNIX or 255 characters for Windows)

userPriority (%d)
  User priority

rsvId %s
  Advance reservation ID; for example, "user2#0"

extsched (%s)
  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

Event time (%d)
The time of the event

jobId (%d)
Job ID

userId (%d)
UNIX user ID of the user invoking the command

runCount (%d)
Number of runs

signalSymbol (%s)
Signal name

idx (%d)
Job array index

userName (%s)
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:

Version number (%s)
The version number

Event time (%d)
The time of the event

jobId (%d)
Job ID

execUid (%d)
Mapped UNIX user ID on execution host

jobPGid (%d)
Job process group ID

execCwd (%s)
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)
lsb.events

Mapped user name on execution host

jobPid (%d)
Job process ID

idx (%d)
Job array index

additionalInfo (%s)
Placement information of HPC jobs

SLAscaledRunLimit (%d)
Run time limit for the job scaled by the execution host

execRusage
An internal field used by LSF.

Position
An internal field used by LSF.

duration4PreemptBackfill
How long a backfilled job can run; used for preemption backfill jobs

**JOB_REQUEUE**
This is created when a job ended and requeued by mbatchd. 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

**JOB_CLEAN**
This is created when a job is removed from the mbatchd memory. 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

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: missched
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 missched exception specifies when time event ended.

exceptInfo (%d)

Except Info, pending reason for missched or cantrun exception, the exit code of the job for the abend exception, otherwise 0.

idx (%d)
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_ATTA_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

idx (%d)

Job array index

msgIdx (%d)

Index in the list

dataSize (%ld)

Size of the data if is has any, otherwise 0

dataStatus (%d)

Status of the attached data

fileName (%s)

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:

Version number (%s)

The version number

Event time (%d)

The time of the event

membSize (%ld)

Size of array membJobId

membJobId (%ld)

Job IDs of jobs in the chunk

numExHosts (%ld)

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.
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 <lsf/lsbatch.h>

suspreason (%d)
Pending or suspending subreason code, see <lsf/lsbatch.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)

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_outblock (%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_msgrcv (%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)

exitStatus (%d)  
Exit status of the job, see <lsf/lsbatch.h>

execCwd (%s)
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)

Total 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
npids (%d)
Number of currently active process in given process groups. This entry has four sub-fields:

pid (%d)
Process ID of the child `sbatchd` that initiated the action

ppid (%d)
Parent process ID

pgid (%d)
Process group ID

jobId (%d)
Process Job ID

npgids (%d)
Number of currently active process groups

exitInfo (%d)
Job termination reason, see `<lsf/lsbatch.h>`

PRE_EXEC_START
A pre-execution command has been started.
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

**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 (%ld)**
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. For MultiCluster this is the name of the receive queue at the execution cluster.

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.

idx (%d)
Job array index.

notifyId (%d)
Identifier or handle for notification.

resizeNotifyCmdPid (%d)
Resize notification executable process ID. If no resize notification executable is defined, this field will be set to 0.

resizeNotifyCmdPGid (%d)
Resize notification executable process group ID. If no resize notification executable is defined, this field will be set to 0.

status (%d)
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:

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.

status (%d)
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:

Version number (%s)
The version number.

**Event time (%d)**

The time of the event.

**jobId (%d)**

The job ID.

**idx (%d)**

Job array index.

**reqid (%d)**

Request Identifier or handle.

**options (%d)**

Release options.

**userId (%d)**

UNIX user ID of the user invoking the command.

**userName (%s)**

User name of the submitter.

**resizeNotifyCmd (%s)**

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

**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.

**JOB_RESIZE_CANCEL**

LSF logs this event when receiving cancel request from client. 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.

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 badmin 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.
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 lsb.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 lsb.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 host A.

Begin Host

<table>
<thead>
<tr>
<th>HOST_NAME</th>
<th>MXJ</th>
<th>EXIT_RATE</th>
<th>Keywords</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>!</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>hostA</td>
<td>!</td>
<td>10</td>
<td></td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>HOST_NAME</th>
<th>JL/U</th>
</tr>
</thead>
<tbody>
<tr>
<td>hostA</td>
<td>2</td>
</tr>
</tbody>
</table>

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 \texttt{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
```
Example of a Host section

<table>
<thead>
<tr>
<th>HOST_NAME</th>
<th>MXJ</th>
<th>J/L/U</th>
<th>r1m</th>
<th>pg</th>
<th>DISPATCH_WINDOW</th>
</tr>
</thead>
<tbody>
<tr>
<td>hostA</td>
<td>1</td>
<td>-</td>
<td>0.6/1.6</td>
<td>10/20</td>
<td>(5:19:00-1:8:30 20:00-8:30)</td>
</tr>
<tr>
<td>SUNSOL</td>
<td>1</td>
<td>-</td>
<td>0.5/2.5</td>
<td>-</td>
<td>23:00-8:00</td>
</tr>
<tr>
<td>default</td>
<td>2</td>
<td>1</td>
<td>0.6/1.6</td>
<td>20/40</td>
<td>()</td>
</tr>
</tbody>
</table>

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.

Host A runs one batch job at a time. A job will only be started on host A 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. Host A 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)

End HostGroup

This example defines three host groups:

- groupA includes hostA and hostD and can be administered by user1 and user10.
- **groupB** includes hostF and hostK, 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 group executable.

### Example 2

```
Begin HostGroup

<table>
<thead>
<tr>
<th>GROUP_NAME</th>
<th>GROUP_MEMBER</th>
<th>GROUP_ADMIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>groupA</td>
<td>(all)</td>
<td>()</td>
</tr>
<tr>
<td>groupB</td>
<td>(groupA ~hostA ~hostB)</td>
<td>(user11 user14)</td>
</tr>
<tr>
<td>groupC</td>
<td>(hostX hostY hostZ)</td>
<td>()</td>
</tr>
<tr>
<td>groupD</td>
<td>(groupC ~hostX)</td>
<td>usergroupB</td>
</tr>
<tr>
<td>groupE</td>
<td>(all ~groupC ~hostB)</td>
<td>()</td>
</tr>
<tr>
<td>groupF</td>
<td>(hostF groupC hostK)</td>
<td>()</td>
</tr>
</tbody>
</table>

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 hostA and hostB and is administered by user11 and user14.
- **groupC** contains only hostX, hostY, and hostZ and is administered by the cluster administrator.
- **groupD** contains the hosts in groupC except for hostX. Note that hostX must be a member of host 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 hostB and is administered by the cluster administrator.
- **groupF** contains hostF, hostK, and the 3 hosts in groupC and is administered by the cluster administrator.

### Example 3

```
Begin HostGroup

<table>
<thead>
<tr>
<th>GROUP_NAME</th>
<th>CONDENSE</th>
<th>GROUP_MEMBER</th>
<th>GROUP_ADMIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>groupA</td>
<td>N</td>
<td>(all)</td>
<td>()</td>
</tr>
<tr>
<td>groupB</td>
<td>N</td>
<td>(hostA, hostB)</td>
<td>(usergroupC user1)</td>
</tr>
<tr>
<td>groupC</td>
<td>Y</td>
<td>(all)</td>
<td>()</td>
</tr>
</tbody>
</table>

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 hostA and hostB. 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]) ()

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 `hostA` and is administered by the cluster administrator.
- **groupC** shows uncondensed output, and contains all hosts starting with the string `hostB` except for the hosts from `hostB1` to `hostB50` and is administered by the cluster administrator.
- **groupD** shows condensed output, and contains all hosts from `hostC1` to `hostC50` and all hosts from `hostC101` to `hostC150` and is administered by the the members of `usergroupJ`.
- **groupE** shows uncondensed output, and contains all hosts from `hostC51` to `hostC100` and all hosts from `hostC151` to `hostC200` and is administered by the cluster administrator.
- **groupF** shows condensed output, and contains `hostD1, hostD3`, and all hosts from `hostD5` to `hostD10` and is administered by the cluster administrator.
- **groupG** shows uncondensed output, and contains all hosts from `hostD11` to `hostD50` except for `hostD15, hostD20, and hostD25`. **groupG** also includes `hostD2`. 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 `HostPartition` 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 host K and host M.

\[\text{HOSTS} = \text{groupA} \simhostL\]

The partition includes all the hosts in host group groupA except for host L.

**USER_SHARES**

**Syntax**

\[\text{USER_SHARES} = [\text{user}, \text{number_shares}]...\]

**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**

```plaintext
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`, `allremote`, `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:
  
  ... (enclA[1-10]B[1-20] enclC[101-120])

- The following example is correct:
  
  ... (enclA[1-20] enclC[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=enclosure rack cabinet
  
  then a compute unit of type cabinet can contain compute units of type rack, but not of type enclosure.

- The keywords all, allremote, all@cluster, other and default 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: *, !, $, #, & or ~).
- 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 lsb.params entry
  COMPUTE_UNIT_TYPES=enclosure rack cabinet)

Begin ComputeUnit

<table>
<thead>
<tr>
<th>NAME</th>
<th>MEMBER</th>
<th>TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>encl1</td>
<td>(host1 host2)</td>
<td>enclosure</td>
</tr>
<tr>
<td>encl2</td>
<td>(host3 host4)</td>
<td>enclosure</td>
</tr>
<tr>
<td>encl3</td>
<td>(host5 host6)</td>
<td>enclosure</td>
</tr>
<tr>
<td>encl4</td>
<td>(host7 host8)</td>
<td>enclosure</td>
</tr>
<tr>
<td>rack1</td>
<td>(encl1 encl2)</td>
<td>rack</td>
</tr>
<tr>
<td>rack2</td>
<td>(encl3 encl4)</td>
<td>rack</td>
</tr>
<tr>
<td>cbnt1</td>
<td>(rack1 rack2)</td>
<td>cabinet</td>
</tr>
</tbody>
</table>

End ComputeUnit

This example defines seven compute units:

- encl1, encl2, encl3 and encl4 are the finest granularity, and each contain two hosts.
- rack1 is of coarser granularity and contains two levels. At the enclosure level rack1 contains encl1 and encl2. At the lowest level rack1 contains host1, host2, host3, and host4.
- rack2 has the same structure as rack1, and contains encl3 and encl4.
- cbnt1 contains two racks (rack1 and rack2), four enclosures (encl1, encl2, encl3, and encl4) and all eight hosts. Compute unit cbnt1 is the coarsest granularity in this example.
Example 2

(For the lsb.params entry COMPUTE_UNIT_TYPES=enclosure rack cabinet)

Begin ComputeUnit

<table>
<thead>
<tr>
<th>NAME</th>
<th>CONDENSE</th>
<th>MEMBER</th>
<th>TYPE</th>
<th>ADMIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>encl1</td>
<td>Y</td>
<td>(hg123 ~hostA ~hostB)</td>
<td>enclosure</td>
<td>(user11 user14)</td>
</tr>
<tr>
<td>encl2</td>
<td>Y</td>
<td>(hg456)</td>
<td>enclosure</td>
<td>()</td>
</tr>
<tr>
<td>encl3</td>
<td>N</td>
<td>(hostA hostB)</td>
<td>enclosure</td>
<td>usergroupB</td>
</tr>
<tr>
<td>encl4</td>
<td>N</td>
<td>(hgroupX ~hostB)</td>
<td>enclosure</td>
<td>()</td>
</tr>
<tr>
<td>encl5</td>
<td>Y</td>
<td>(hostC* ~hostC[101-150])</td>
<td>enclosure</td>
<td>usergroupJ</td>
</tr>
<tr>
<td>encl6</td>
<td>N</td>
<td>(hostC[101-150])</td>
<td>enclosure</td>
<td>()</td>
</tr>
<tr>
<td>rack1</td>
<td>Y</td>
<td>(encl1 encl2 encl3)</td>
<td>rack</td>
<td>()</td>
</tr>
<tr>
<td>rack2</td>
<td>N</td>
<td>(encl4 encl5)</td>
<td>rack</td>
<td>usergroupJ</td>
</tr>
<tr>
<td>rack3</td>
<td>N</td>
<td>(encl6)</td>
<td>rack</td>
<td>()</td>
</tr>
<tr>
<td>cbnt1</td>
<td>Y</td>
<td>(rack1 rack2)</td>
<td>cabinet</td>
<td>()</td>
</tr>
<tr>
<td>cbnt2</td>
<td>N</td>
<td>(rack3)</td>
<td>cabinet</td>
<td>user14</td>
</tr>
</tbody>
</table>

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.
- encl1 contains all the hosts in host group hg123 except for host A and host B and is administered by user11 and user14. Note that host A and host B must be members of host group hg123 to be excluded from encl1. encl1 shows condensed output.
- encl2 contains host group hg456 and is administered by the cluster administrator. encl2 shows condensed output.
- encl3 contains host A and host B. usergroupB is the administrator for encl3. encl3 shows uncondensed output.
- encl4 contains host group hgroupX except for host B. Since each host can appear in only one enclosure and host B is already in encl3, it cannot be in encl4. encl4 is administered by the cluster administrator. encl4 shows uncondensed output.
- encl5 contains all hosts starting with the string host C except for hosts host C101 to host C150, and is administered by usergroupJ. encl5 shows condensed output.
- rack1 contains encl1, encl2, and encl3. rack1 shows condensed output.
- rack2 contains encl4, and encl5. rack2 shows uncondensed output.
- rack3 contains encl6. rack3 shows uncondensed output.
- cbnt1 contains rack1 and rack2. cbnt1 shows condensed output.
- cbnt2 contains rack3. Even though rack3 only contains encl6, cbnt3 cannot contain encl6 directly because that would mean skipping the level associated with compute unit type rack. cbnt2 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 badmin 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.
Reconfiguration is done in real time without restarting `mbatchd`, 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   r1s   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
host2      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/configdir, where LSB_CONFDIR is defined in lsf.conf.

Changing lsb.modules configuration

After making any changes to lsb.modules, run badmin 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
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 `lsbqueues`.

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`.

Platform LSF Configuration Reference 129
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/


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

<table>
<thead>
<tr>
<th>NAME</th>
<th>RB_PLUGIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>schmod_default</td>
<td>()</td>
</tr>
<tr>
<td>schmod_fairshare</td>
<td>(rb_fairshare)</td>
</tr>
</tbody>
</table>

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:

<table>
<thead>
<tr>
<th>NAME</th>
<th>SCH_DISABLE_PHASES</th>
</tr>
</thead>
<tbody>
<tr>
<td>schmod_default</td>
<td>()</td>
</tr>
<tr>
<td>schmod_custom</td>
<td>(3,4)</td>
</tr>
</tbody>
</table>

**Default**

Undefined
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 `badmin reconfig` to reconfigure `mbatchd`.

### 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 `badmin 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

```bash
# 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
```

### 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:

<table>
<thead>
<tr>
<th>Begin Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEFAULT_QUEUE = normal  #default job queue name</td>
</tr>
<tr>
<td>MBD_SLEEP_TIME = 20 #Time used for calculating parameter values (60 secs is default)</td>
</tr>
<tr>
<td>SBD_SLEEP_TIME = 15 #sbatchd scheduling interval (30 secs is default)</td>
</tr>
<tr>
<td>JOB_ACCEPT_INTERVAL = 1 #interval for any host to accept a job</td>
</tr>
<tr>
<td>#(default is 1 (one-fold of MBD_SLEEP_TIME))</td>
</tr>
</tbody>
</table>

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 or run time estimate specified by the -W or -We option of bsub
- RUNLIMIT queue-level parameter in lsb_queues
- RUNLIMIT application-level parameter in lsb_applications
- RUNTIME parameter in lsb_applications

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 size of lsb.acct)

**ACCT_ARCHIVE_SIZE**

**Syntax**

\[
\text{ACCT\_ARCHIVE\_SIZE} = \text{kilobytes}
\]

**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 lsb.acct)

**ACCT_ARCHIVE_TIME**

**Syntax**

\[
\text{ACCT\_ARCHIVE\_TIME} = \text{hh:mm}
\]

**Description**

Enables automatic archiving of LSF accounting log file lsbacct, 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 lsbacct)
ADVRSV_USER_LIMIT

Syntax

\[
\text{ADVRSV\_USER\_LIMIT} = \text{integer}
\]

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

\[
\text{CHUNK\_JOB\_DURATION} = \text{minutes}
\]

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 b s u b 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:

\begin{itemize}
  \item No CPU limit, run time limit, or run time estimate is specified at any level, or
  \item A CPU limit, run time limit, or run time estimate is greater than the value of CHUNK_JOB_DURATION.
\end{itemize}

The value of CHUNK_JOB_DURATION is displayed by `b p a r a m s - l`.

Examples

\begin{itemize}
  \item CHUNK_JOB_DURATION is not defined:
    \begin{itemize}
      \item Jobs with no CPU limit, run limit, or run time estimate are chunked
      \item Jobs with a CPU limit, run limit, or run time estimate less than or equal to 30 are chunked
      \item Jobs with a CPU limit, run limit, or run time estimate greater than 30 are not chunked
    \end{itemize}
  \item CHUNK_JOB_DURATION=90:
    \begin{itemize}
      \item Jobs with no CPU limit, run limit, or run time estimate are not chunked
      \item Jobs with a CPU limit, run limit, or run time estimate less than or equal to 90 are chunked
      \item Jobs with a CPU limit, run limit, or run time estimate greater than 90 are not chunked
    \end{itemize}
\end{itemize}
**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 `bjobs` 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.

**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, `bjobs -p` or `bjobs
  -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**

```plaintext
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**

```plaintext
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**

```plaintext
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=//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_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

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 all 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 badmin 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

DETECT_IDLE_JOB_AFTER

Syntax

DETECT_IDLE_JOB_AFTER=time_minutes
Description
The minimum job run time before `mbatchd` reports that the job is idle.

Default
20 (mbatchd 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/eadmin` 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`
- `b restart`
- `bswitch`

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:
• 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 `blimits`.

• 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 `blimits`) 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 `badmin reconfig` to reconfigure your cluster.

Default
N (Not defined.)
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/lsb.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_SHAREDIR 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 eadmin 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_SERVERDIR/eadmin` 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 lsb.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 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. 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 (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

1
**JOB_ATTA_DIR**

**Syntax**

```shell
JOB_ATTA_DIR=directory
```

**Description**

The shared directory in which `mbatchd` saves the attached data of messages posted with the `bpost` command.

Use `JOB_ATTA_DIR` if you use `bpost` and `bread` to transfer large data files between jobs and want to avoid using space in `LSB_SHAREDIR`. By default, the `bread` command reads attachment data from the `JOB_ATTA_DIR` directory.

`JOB_ATTA_DIR` should be shared by all hosts in the cluster, so that any potential LSF master host can reach it. Like `LSB_SHAREDIR`, 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_ATTA_DIR/timestamp/jobidmsgs/msg$msgid
```

On UNIX, specify an absolute path. For example:

```bash
JOB_ATTA_DIR=/opt/share/lsf_work
```

On Windows, specify a UNC path or a path with a drive letter. For example:

```bash
JOB_ATTA_DIR=\HostA\temp\lsf_work
```

or

```bash
JOB_ATTA_DIR=D:\temp\lsf_work
```

After adding `JOB_ATTA_DIR` to `lsb.params`, use `badmin reconfig` to reconfigure your cluster.

**Valid values**

`JOB_ATTA_DIR` can be any valid UNIX or Windows path up to a maximum length of 256 characters.

**Default**

Not defined

If `JOB_ATTA_DIR` is not specified, job message attachments are saved in `LSB_SHAREDIR/info`.

**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.

Running jobs are not affected when `JOB_DEP_LAST_SUB` is changed. To reevaluate job dependencies after changing `JOB_DEP_LAST_SUB`, run `badmin mbdrestart`.
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.
\texttt{sbatchd} sends both job finish status (DONE or EXIT) and post-execution processing status (POST_DONE or POST_ERR) to \texttt{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 \texttt{lsb.applications}.
JOB\_INCLUDE\_POSTPROC in an application profile in \texttt{lsb.applications} overrides the value of JOB\_INCLUDE\_POSTPROC in \texttt{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**

\texttt{JOB\_POSITION\_CONTROL\_BY\_ADMIN=\{Y | N\}}

**Description**

Allows LSF administrators to control whether users can use \texttt{btop} and \texttt{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 \texttt{bbot} and \texttt{btop} to move jobs within a queue.

**Default**

N

**See also**

\texttt{bbot, btop}

**JOB\_POSTPROC\_TIMEOUT**

**Syntax**

\texttt{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, \texttt{sbatchd} 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 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 the user environment.

In the MultiCluster job forwarding model, the JOB_POSTPROC_TIMEOUT value in the receiving cluster
applies to the job.

In the 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

\[ \text{JOB\_RUNLIMIT\_RATIO} = \text{integer} \mid 0 \]

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 (`-W e`) 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 -W`). 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 -W`, the run limit is automatically 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 -W` and the run limit is set by the user, the run limit is `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 -W` 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 `mbatchd` 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/lsf_indir`, and `bsub -Zs` cannot write to the default directory `LSB_SHAREDIR/cluster_name/lsf_cmddir`, 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/lsf_spool
```

On Windows, specify a UNC path or a path with a drive letter. For example:

```
JOB_SPOOL_DIR=\HostA\share\spooldir
```

or

```
JOB_SPOOL_DIR=D:\share\spooldir
```

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/lsf_spool | \HostA\share\spooldir
```

**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%\lsbtmp\user_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_PREEXEC_RETRY

Syntax

```
LOCAL_MAX_PREEXEC_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_PREEXEC_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 `mbatchd` 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, `mbatchd` periodically obtains the host status from the master LIM, and then verifies the status by polling each `sbatchd` at an interval defined by the parameters `MBD_SLEEP_TIME` and `LSB_MAX_PROBE_SBD`.

Default

```
N. mbatchd obtains and reports host status, without contacting the master LIM, by polling each sbatchd 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
```

Platform LSF Configuration Reference 157
**Max_acct_archive_file**

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 `mbatchd` 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 `mbatchd` is not using multithreading, the value of `MAX_CONCURRENT_JOB_QUERY` is always the maximum number of job queries in the cluster.
- If `mbatchd` 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 `mbatchd` that is forked by `mbatchd`. When the new child `mbatchd` starts, it handles new queries, but the old child `mbatchd` 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 `mbatchd`.

### Valid values

1-100
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.utc files that mbatchd uses. If the number of lsb.stream.utc files reaches this number, mbatchd logs an 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
```

Platform LSF Configuration Reference 159
**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:

\[
\text{subdirectory} = \text{jobID} \mod \text{MAX_INFO_DIRS}
\]

This formula ensures an even distribution of job files across the subdirectories.

**Important:**

If you are using local duplicate event logging, you must run `badmin 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 job files to the `info` directory)

**Example**

`MAX_INFO_DIRS=10`

`mbatchd` creates ten subdirectories from `LSB_SHAREDIR/cluster_name/logdir/info/0` to `LSB_SHAREDIR/cluster_name/logdir/info/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_ATTA_SIZE

Syntax

\[ \text{MAX_JOB_ATTA_SIZE} = \text{integer} | 0 \]

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_NUM

Syntax

\[ \text{MAX_JOB_NUM} = \text{integer} \]

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

\[ \text{MAX_JOB_PREEMPT} = \text{integer} \]

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 < 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 bjobs) 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.

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_PREEXEC_RETRY

Syntax

MAX_PREEXEC_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 < MAX_PREEXEC_RETRY < 2147483647

Default

5

MAX_SBD_CONNS

Syntax

MAX_SBD_CONNS=integer

Description

The maximum number of file descriptors mbatchd can have open and connected concurrently to sbatchd.

Controls the maximum number of connections that LSF can maintain to sbatchds 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 mbatchd to constantly die because mbatchd allocates all file descriptors to sbatchd connection. This could cause mbatchd to run out of descriptors, which results in an mbatchd 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, sbatchd.
The interval between retries is defined by MBD_SLEEP_TIME. If mbatchd 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, mbatchd obtains the host status from the master LIM
before it polls sbatchd. When the master LIM reports that a host is unavailable (LIM is down) or
unreachable (sbatchd is down) MAX_SBD_FAIL number of times, mbatchd 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 `btop` 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`, `btop`, `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 mbatchd 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 mbatchd child query processes can run (hard CPU affinity). This improves mbatchd scheduling and dispatch performance by binding query processes to specific CPUs so that higher priority mbatchd processes can run more efficiently.

When you define this parameter, LSF runs mbatchd 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 mbatchd 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**

```bash
MBD_REFRESH_TIME=seconds [min_refresh_time]
```

where `min_refresh_time` defines the minimum time (in seconds) that the child `mbatchd` will stay to handle queries.

**Description**

Time interval, in seconds, when `mbatchd` will fork a new `mbatchd` to service query requests to keep information sent back to clients updated. A child `mbatchd` 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 `mbatchd` 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 `< min_refresh_time`, the child `mbatchd` 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` is `> min_refresh_time`:
  - the child `mbatchd` exits at `min_refresh_time` if a job changes status or a new job is submitted before the `min_refresh_time`
  - the child `mbatchd` exits after the `min_refresh_time` when a job changes status or a new job is submitted
- If `MBD_REFRESH_TIME` is `> min_refresh_time` and no job changes status or a new job is submitted, the child `mbatchd` 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 `bjobs` command may not display up-to-date information if two consecutive query commands are issued before a child `mbatchd` expires because child `mbatchd` job information is not updated. If you use the `bjobs` 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
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

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

\[MC\_RECLAIM\_DELAY=\text{minutes}\]

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

\[MC\_RUSAGE\_UPDATE\_INTERVAL=\text{seconds}\]

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

\[MIN\_SWITCH\_PERIOD=\text{seconds}\]

Description

The minimum period in seconds between event log switches.

Works together with MAX_JOB_NUM to control how frequently mbatchd switches the file. mbatchd checks if MAX_JOB_NUM has been reached every MIN_SWITCH_PERIOD seconds. If mbatchd finds that MAX_JOB_NUM has been reached, it switches the events file.

To significantly improve the performance of mbatchd for large clusters, set this parameter to a value equal to or greater than 600. This causes mbatchd to fork a child process that handles event switching, thereby reducing the load on mbatchd. mbatchd 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 mbatchd to get up to date information about new jobs from the parent mbatchd. When set to Y, job queries with bjobs display new jobs submitted after the child mbatchd was created.

If you have enabled multithreaded mbatchd support, the bjobs command may not display up-to-date information if two consecutive query commands are issued before a child mbatchd expires because child mbatchd job information is not updated. Use NEWJOB_REFRESH=Y to enable the parent mbatchd to push new job information to a child mbatchd.

When NEWJOB_REFRESH=Y, as users submit new jobs, the parent mbatchd pushes the new job event to the child mbatchd. The parent mbatchd transfers the following kinds of new jobs to the child mbatchd:

- 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 mbatchd only pushes the new job event to a child mbatchd. The child mbatchd is not aware of status changes of existing jobs. The child mbatchd will not reflect the results of job control commands (bmod, b mig, bswitch, btop, bbot, brequeue, bstop, b resume, and so on) invoked after the child mbatchd is created.

Default

N (Not defined. New jobs are not pushed to the child mbatchd.)

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_RUN_TIME, requires a run time (bsub -W 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.)

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)

**MAX_JOB_MSG_NUM**

**Syntax**

```
MAX_JOB_MSG_NUM=integer | 0
```

**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**

-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 mbsched.

**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

**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.
LSBDISABLELIMLOCKEXCL=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.

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)*

**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 `mbatchd` 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_PREEXEC_RETRY` in `lsb.params` to limit the total number of hosts that are tried. In a multicluster environment, use in combination with `LOCAL_MAX_PREEXEC_RETRY` and `REMOTE_MAX_PREEXEC_RETRY`.

**Default**

None.

**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_PREEXEC_RETRY**

**Syntax**

`REMOTE_MAX_PREEXEC_RETRY=integer`

**Description**

The maximum number of times to attempt the pre-execution command of a job from the remote cluster.
Valid values

\[0 < \text{REMOTE\_MAX\_PREEXEC\_RETRY} < 2147483647\]

Default

5

**RESOURCE\_RESERVE\_PER\_SLOT**

**Syntax**

\[\text{RESOURCE\_RESERVE\_PER\_SLOT}=y \mid Y\]

**Description**

If Y, `mbatchd` reserves resources based on job slots instead of per-host.

By default, `mbatchd` 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_job
```

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_job` must reserve 500 MB of memory for each job slot (4*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]" my_job
```

Default

N (Not defined; reserve resources per-host.)

**RUN\_JOB\_FACTOR**

**Syntax**

\[\text{RUN\_JOB\_FACTOR}=\text{number}\]

**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 `lsb.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 lsb.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 lsb.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.

**SCHED_METRIC_ENABLE**

**Syntax**

```
SCHED_METRIC_ENABLE=Y | N
```

**Description**

Enable scheduler performance metric collection.

Use `badmin perfmon stop` and `badmin perfmon start` to dynamically control performance metric collection.

**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.

Use `badmin perfmon set period` 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.sched.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**

\[
\text{SSCHED\_MAX\_TASKS}=\text{integer}
\]

**Description**

Used by Platform Session Scheduler (\text{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**

\[
\text{SSCHED\_REQUEUE\_LIMIT}=\text{integer}
\]

**Description**

Used by Platform Session Scheduler (\text{ssched}).

Number of times Session Scheduler tries to requeue a task as a result of the \text{REQUEUE\_EXIT\_VALUES} (\text{ssched} -Q) setting. \text{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

**Default**

3 requeue attempts

**SSCHED_RETRY_LIMIT**

**Syntax**

\[
\text{SSCHED\_RETRY\_LIMIT}=\text{integer}
\]

**Description**

Used by Platform Session Scheduler (\text{ssched}).

Number of times Session Scheduler tries to retry a task that fails during dispatch or setup. \text{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 `all` as a member

After adding or changing `STRICT_UG_CONTROL` in `lsb.params`, use `badmin 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.

**Default**

60 seconds

**See also**

MAX_PEND_JOBS

**SYSTEM_MAPPING_ACCOUNT**

**Syntax**

```
SYSTEM_MAPPING_ACCOUNT=user_account
```
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*
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 badmin 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 (value_s), minutes (value_m), or hours (value_h).
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 JPRIORITY factor, 10 seconds for the QPRIORITY factor, and 10 hours (default) for the RSRC factor:

```
GRACE_PERIOD[MEM, 10h] [JPRIORITY, 10m] [QPRIORITY, 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 (bstop) jobs and migrated jobs (bmig) 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, you should use brequeue, not bmig.

### 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 lsf.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

\[ \text{CPULIMIT=}[\text{default\_limit}]\text{ maximum\_limit} \]

where `default_limit` and `maximum_limit` are:
\[ \text{[hour:]minute[\text{/host\_name}|\text{/host\_model}]} \]

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 `sbatchd` 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**

```plaintext
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**

```plaintext
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**

```plaintext
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_HOST_SPEC

Syntax

\[ \text{DEFAULT\_HOST\_SPEC} = \text{host\_name} | \text{host\_model} \]

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

\[ \text{DESCRIPTION} = \text{text} \]

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

\[ \text{DISPATCH\_ORDER} = \text{QUEUE} \]

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)
- 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:
\begin{itemize}
  \item In queue \texttt{normal}: \texttt{FAIRSHARE QUEUES=short license}
  \item In queue \texttt{priority}: \texttt{FAIRSHARE QUEUES=night owners}
\end{itemize}

\underline{Restriction:}

You cannot, however, define \texttt{night}, \texttt{owners}, or \texttt{priority} as slaves in the queue \texttt{normal}; or \texttt{normal}, \texttt{short} and \texttt{license} as slaves in the \texttt{priority} queue; or \texttt{short,license,night}, \texttt{owners} as master queues of their own.

\begin{itemize}
  \item Cross-queue fairshare cannot be used with host partition fairshare. It is part of queue-level fairshare.
  \item Cross-queue fairshare cannot be used with absolute priority scheduling.
\end{itemize}

\textbf{Default}

\texttt{Not defined}

\textbf{FILELIMIT}

\textbf{Syntax}

\texttt{FILELIMIT=integer}

\textbf{Description}

The per-process (hard) file size limit (in KB) for all of the processes belonging to a job from this queue (see \texttt{getrlimit(2)}).

\textbf{Default}

\texttt{Unlimited}

\textbf{HIST_HOURS}

\textbf{Syntax}

\texttt{HIST_HOURS=hours}

\textbf{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 \texttt{HIST_HOURS=0}, CPU time and run time accumulated by running jobs is not decayed.

If undefined, the cluster-wide value from the \texttt{lsb.params} parameter of the same name is used.

\textbf{Default}

\texttt{Not defined}.
**HJOB_LIMIT**

**Syntax**

\[ \text{HJOB\_LIMIT} = \text{integer} \]

**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 host A, host B, and host C:

```
Begin Queue
...
HJOB\_LIMIT = 1
HOSTS=hostA hostB hostC
...
End Queue
```

**Default**

Unlimited

---

**HOSTS**

**Syntax**

\[ \text{HOSTS} = \text{host\_list} | \text{none} \]

- 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
  - allremote
• 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 `allremote 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 allremote 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 allremote 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 allremote.
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_JOBBKLG**

**Syntax**

```
IMPT_JOBBKLG=integer |infinit
```

**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 `infinit` 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

```bash
JOB_ACTION_WARNING_TIME=2
```

Default

Not defined

**JOB_CONTROLS**

**Syntax**

```bash
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[bqueue]`. 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_JOBPIDS: 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\[kill \$LSB_JOBPIDS; command\]
  • 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

\[\text{JOB_UNDERRUN} = \text{run\_time}\]

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

\[\text{JOB_UNDERRUN} = 2\]

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

\[\text{JOB_WARNING_ACTION} = \text{signal}\]

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

\[\text{JOB_WARNING_ACTION} = \text{URG}\]

Default

Not defined

load_index

Syntax

\[\text{load\_index} = \text{loadSched}[/\text{loadStop}]\]
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 lsload -E, which is normalized for multiprocessor hosts. Thresholds for these parameters should be set at appropriate levels for single processor hosts.

**Example**

```plaintext
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_PREEXEC_RETRY**

**Syntax**

```plaintext
LOCAL_MAX_PREEXEC_RETRY=integer
```

**Description**

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

**Valid values**

```plaintext
0 < MAX_PREEXEC_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 < MAX_JOB_PREEMPT < 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 < MAX_JOB_REQUEUE < INFINIT_INT
INFINIT_INT is defined in lsf.h.
Default
Not defined. The number of requeue times is unlimited.
MAX_PREEXEC_RETRY

Syntax

```
MAX_PREEXEC_RETRY=integer
```

Description

Use REMOTE_MAX_PREEXEC_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 < MAX_PREEXEC_RETRY < INFINIT_INT
```

INFINIT_INT is defined in lsf.h.

Default

5

MAX_RSCHED_TIME

Syntax

```
MAX_RSCHED_TIME=integer | infinit
```

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 infinit 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 (renice) 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**

\[ \text{MIG} = \text{minutes} \]

**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 \textit{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 \textit{WAIT} state are removed from the job chunk and put into \textit{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**

\[ \text{NEW_JOB_SCHED_DELAY} = \text{seconds} \]

**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**

\[ \text{NICE} = \text{integer} \]
lsb.queues

**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 `nice(1)` manual page for more details.

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

- \texttt{nice} \geq 0 \text{ corresponds to an priority class of IDLE}
- \texttt{nice} < 0 \text{ corresponds to an priority class of NORMAL}

Platform LSF on Windows does not support \texttt{HIGH} or \texttt{REAL-TIME} priority classes.

This value is overwritten by the NICE setting in `lsb.applications`, if defined.

**Default**

0 (zero)

**NO_PREEMPT_INTERVAL**

**Syntax**

\[
\text{NO_PREEMPT_INTERVAL}=\text{minutes}
\]

**Description**

Prevents preemption of jobs for the specified number of minutes of uninterrupted run time, where minutes is wall-clock time, not normalized time. \texttt{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**

\[
\text{NQS_QUEUES}=\text{NQS\_queue\_name@NQS\_host\_name...}
\]

**Description**

Makes the queue an NQS forward queue.

\texttt{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.

\texttt{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.

\texttt{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 `QUEUE_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:

```bash
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

  ```
  /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 \texttt{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.

\textit{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.

\texttt{+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.

\textit{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.

\textbf{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 \texttt{normal} and \texttt{low} queues, beginning with jobs from \texttt{low}, as indicated by the preference (+1)
- normal
  - Has the second-lowest relative priority, at 5
  - Jobs from this queue can preempt jobs from \texttt{low}, and can be preempted by jobs from both \texttt{high} and \texttt{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 \texttt{PREEMPTABLE} keyword set.

\begin{verbatim}
Begin Queue
QUEUE_NAME=high
PREEMPTION=PREEMPTIVE
PRIORITY=99
End Queue

Begin Queue
QUEUE_NAME=medium
PREEMPTION=PREEMPTIVE[low+1]
PRIORITY=10
End Queue

Begin Queue
QUEUE_NAME=normal
PREEMPTION=PREEMPTIVE[low]
PREEMPTABLE[high medium]
PRIORITY=5
End Queue

Begin Queue
QUEUE_NAME=low
PRIORITY=1
End Queue
\end{verbatim}

## PRIORITY

### Syntax

\texttt{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 \texttt{NICE} parameter is used to set the UNIX time-sharing priority for batch jobs.

\texttt{integer}

Specify a number greater than or equal to 1, where 1 is the lowest priority.

### Default

1
PROCESSLIMIT
Syntax

\[\text{PROCESSLIMIT}=\text{default\_limit} \text{ maximum\_limit}\]

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

\[\text{PROCLIMIT}=\text{minimum\_limit} \text{ [default\_limit]} \text{ maximum\_limit}\]

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 (\text{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 \text{lsb.applications} 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:

\[1 \leq \text{minimum} \leq \text{default} \leq \text{maximum}\]

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

\[\text{QJOB\_LIMIT}=\text{integer}\]
**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_PREEXEC_RETRY

Syntax

REMOTE_MAX_PREEXEC_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 same host. 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 mbatchd 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
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:**

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 queue. However, lic=1 from queue is kept, since job does not specify it.

**Queue-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]

Queue-level duration and decay are merged with the job-level specification, and mem=100 for the job overrides mem=200 specified by the queue. However, duration=20 and decay=1 from queue are kept, since job does not specify them.

**Queue-level RES_REQ with multi-phase job-level rusage:**

RES_REQ=rusage[mem=200:duration=20:decay=1] ...

For a job submission with no decay or duration:

bsub -R'rusage[mem=(300 200 100):duration=(10 10 10)]' ...

the resulting requirement for the job is:

rusage[mem=(300 200 100):duration=(10 10 10)]
Multi-phase rusage values in the job submission override the single phase specified by the queue.

- If RESRSV_LIMIT is defined in lsb.queues and has a maximum memory limit of 300 MB or greater, this job will be accepted.
- If RESRSV_LIMIT is defined in lsb.queues and has a maximum memory limit of less than 300 MB, this job will be rejected.
- If RESRSV_LIMIT is not defined in lsb.queues and the queue-level RES_REQ value of 200 MB acts as a ceiling, this job will be rejected.

For a single phase job submission with no decay or duration:

```bash
bsub -q q_name -R 'rusage[mem=100:swap=150]' ...
```

the resulting requirement for the job is:

```bash
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 bjobs.

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 badmin reconfig. 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

```bash
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**

```bash
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 `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**
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 lslslsinfo(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.
**SLOT_SHARE**

**Syntax**

```plaintext
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**

```plaintext
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**

```plaintext
SNDJOBS_TO = queue2@cluster2 queue3@cluster2 queue3@cluster3
```

---

**STACKLIMIT**

**Syntax**

```plaintext
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 `bjobs`.

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 belong 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-ipo
- linux2.6-glibc2.3-x86
- linux2.6-glibc2.3-x86_64

Overrides MEMLIMIT_TYPE=Process.

Overridden (for CPU limit only) by LSB_JOBS_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
```

Platform LSF Configuration Reference 237
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 (bswitch) 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 `badmin 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

```
Begin Queue

...#if time(8:30-18:30)
INTERACTIVE = ONLY  # interactive only during day shift #endif
...
End Queue
```
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/configdir`, where `LSB_CONFDIR` is defined in `lsf.conf`.

**Changing lsb.resources configuration**

After making any changes to `lsb.resources`, run `badmin reconfig` 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:
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 hostA
- 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.

<table>
<thead>
<tr>
<th>Begin Limit</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME</td>
</tr>
<tr>
<td>limit1</td>
</tr>
<tr>
<td>-</td>
</tr>
<tr>
<td>-</td>
</tr>
<tr>
<td>End Limit</td>
</tr>
</tbody>
</table>

Jobs that do not match these limits; that is, all users except user1 and user3 running jobs on hostA and all users except user2 submitting jobs to queue normal, have no limits.

Example: Horizontal format

All users in user group ugroup1 except user1 using queue1 and queue2 and running jobs on hosts in host group hgroup1 are limited to 2 job slots per processor on each host:

<table>
<thead>
<tr>
<th>Begin Limit</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME</td>
</tr>
<tr>
<td># Resources</td>
</tr>
<tr>
<td>SLOTS_PER_PROCESSOR</td>
</tr>
<tr>
<td>#Consumers</td>
</tr>
<tr>
<td>QUEUES</td>
</tr>
<tr>
<td>USERS</td>
</tr>
<tr>
<td>PER_HOST</td>
</tr>
<tr>
<td>End Limit</td>
</tr>
</tbody>
</table>

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
HOSTS

Syntax

```
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 hostB, hostC, and all hosts in Group1 except for hostA.

**Example 2**

HOSTS=all ~group2 ~hostA

Enforces limits on all hosts in the cluster, except for hostA and the hosts in group2.

**Example 3**

<table>
<thead>
<tr>
<th>HOSTS</th>
<th>SWP (all ~hostK ~hostM)</th>
<th>10</th>
</tr>
</thead>
</table>

Enforces a 10 MB swap limit on all hosts in the cluster, except for hostK and hostM.

**JOBS**

**Syntax**

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

LICENSE=[verilog, 4] [spice, 2]

Begin Limit
LICENSE PER_HOST
([verilog, 1]) (all ~hostA)
([verilog, 1] [spice, 2]) (hostA)
End Limit

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=projA projB

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 (-), blimitis 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_limits
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 HOST, 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=proj1 proj2
```

### 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=proj1 proj2
```

### PER_QUEUE

**Syntax**

```
PER_QUEUE=all [~]queue_name..
```

**Description**

Platform LSF Configuration Reference 249
**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=priority night

**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=projA projB |

QUEUES

Syntax

| QUEUES=all [-]queue_name ...
| QUEUES
| ( [-] | all [-]queue_name ... )
lsb.resources

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
```

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, mbatchd adjusts the maximum number of job slots per host accordingly. Allow the mbatchd 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**

```plaintext
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, `mbatchd` adjusts the maximum number of job slots per host accordingly. Allow the `mbatchd` 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/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 projects.

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 tmp 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 tmp 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 tmp 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 tmp 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.serviceclasses file.

GuaranteedResourcePool section structure


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 SLA 1 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**

```plaintext
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**

```plaintext
DESCRIPTION=description
```

**Description**

A description of the guarantee resource pool.

**Default**

None. `DESCRIPTION` is optional.

**SLOTS_PER_HOST**

**Syntax**

```plaintext
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 M B.
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 M B.
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 of 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#2" -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 ~ggroup2 ~hostA
Advance reservations can be created on all hosts in the cluster, except for hostA and the hosts in ggroup2.

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 ReservationUsage

<table>
<thead>
<tr>
<th>RESOURCE</th>
<th>METHOD</th>
<th>RESERVE</th>
</tr>
</thead>
<tbody>
<tr>
<td>licenseX</td>
<td>PER_JOB</td>
<td>Y</td>
</tr>
<tr>
<td>licenseY</td>
<td>PER_HOST</td>
<td>N</td>
</tr>
<tr>
<td>licenseZ</td>
<td>PER_SLOT</td>
<td>N</td>
</tr>
</tbody>
</table>

End ReservationUsage
```

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.

**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 badmin 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

```bash
# 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 bs1a 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/configdir.

Changing lsb.serviceclasses configuration

After making any changes to lsb.serviceclasses, run badmin reconfig to reconfigure mbatchd.

lsb.serviceclasses structure

Each service class definition begins with the line Begin ServiceClass and ends with the line End ServiceClass.

Syntax

<table>
<thead>
<tr>
<th>Begin ServiceClass</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME = string</td>
</tr>
<tr>
<td>PRIORITY = integer</td>
</tr>
<tr>
<td>GOALS = [throughput</td>
</tr>
<tr>
<td>CONTROL_ACTION = VIOLATION_PERIOD[minutes] CMD [action]</td>
</tr>
<tr>
<td>USER_GROUP = all</td>
</tr>
<tr>
<td>DESCRIPTION = text</td>
</tr>
<tr>
<td>End ServiceClass</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Begin ServiceClass</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME = string</td>
</tr>
<tr>
<td>GOALS = guarantee</td>
</tr>
<tr>
<td>ACCESS CONTROL = [QUEUES[ queue ...]] [USERS[ [user_name] [user_group] ...]] [FAIRSHARE_GROUPS[ user_group ...]] [APPS[app_name ...]] [PROJECTS[proj_name ...]] [LIC_PROJECTS[license_proj ...]]</td>
</tr>
<tr>
<td>AUTO_ATTACH = Y</td>
</tr>
<tr>
<td>DESCRIPTION = text</td>
</tr>
<tr>
<td>End ServiceClass</td>
</tr>
</tbody>
</table>

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

```plaintext
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

```plaintext
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

```plaintext
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/sla_violation.log]
```

**Default**

None

**DESCRIPTION**

**Syntax**

`DESCRIPTION=text`

**Description**

Optional. Description of the service class. Use `bsla` 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] \n[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 bsla 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 bsla 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:

\[ \text{begin} \_ \text{time} - \text{end} \_ \text{time} \]

Times are specified in the format:

\[ [\text{day}:]\text{hour}[:\text{minute}] \]

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:

\[ \text{timeWindow}(8:00-14:00 \ 18:00-22:00) \]

is correct, but

\[ \text{timeWindow}(8:00-14:00 \ 11:00-15:00) \]

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) |
| GOALS=| VELOCITY 10 timeWindow (8:30-16:30) |
| GOALS=| GUARANTEE |

**MAX_HOST_IDLE_TIME**

**Syntax**

\[ \text{MAX\_HOST\_IDLE\_TIME} = \text{seconds} \]
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 \texttt{GOALS=[GUARANTEE]}) cannot have \texttt{PRIORITY} set. If defined, it will be ignored.

**Default**

\texttt{None.}

**USER\_GROUP**

**Syntax**

\texttt{USER\_GROUP=\texttt{all} | [\texttt{user\_name}] \texttt{[user\_group]} \ldots}

**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 \texttt{all} 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 \texttt{lsb.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 \texttt{lsb.users}) or UNIX and Windows user groups.

Guarantee SLAs (with \texttt{GOALS=[GUARANTEE]}) cannot have \texttt{USER\_GROUP} set. If defined, it will be ignored.

**Example**

\texttt{USER\_GROUP=user1 user2 ugroup1}

**Default**

\texttt{all} (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 \texttt{accountingUG} for jobs submitted to the queue \texttt{longjobs}. 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 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
```
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 `badmin 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

<table>
<thead>
<tr>
<th>GROUP_NAME</th>
<th>GROUP_MEMBER</th>
<th>GROUP_ADMIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>groupA</td>
<td>(user1 user2 user3 user4) (user5)</td>
<td></td>
</tr>
<tr>
<td>groupB</td>
<td>(groupA user5)</td>
<td>(groupA)</td>
</tr>
<tr>
<td>groupC</td>
<td>(!)</td>
<td>()</td>
</tr>
</tbody>
</table>

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 user’s 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 user’s 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 **restartup** 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**

\((\text{user}, \text{number shares})\)

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, JLP, 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

<table>
<thead>
<tr>
<th>USER_NAME</th>
<th>MAX_JOBS</th>
<th>JL/P</th>
<th>MAX_PEND_JOBS</th>
</tr>
</thead>
<tbody>
<tr>
<td>user1</td>
<td>10</td>
<td>-</td>
<td>1000</td>
</tr>
<tr>
<td>user2</td>
<td>4</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>user3</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>groupA</td>
<td>10</td>
<td>1</td>
<td>100000</td>
</tr>
<tr>
<td>groupA@</td>
<td>-</td>
<td>1</td>
<td>100</td>
</tr>
<tr>
<td>groupC</td>
<td>-</td>
<td>-</td>
<td>500</td>
</tr>
<tr>
<td>default</td>
<td>6</td>
<td>1</td>
<td>10</td>
</tr>
</tbody>
</table>

End User

USER_NAME

User or user group for which job slot limits are defined.

Use the reserved user name default to specify a job slot limit that applies to each user and user group not explicitly named. Since the limit specified with the keyword default 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 all in the user list, the at sign (@) has no effect. To specify job slot limits for each user in a user group containing all, use the keyword default.

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 all 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 namespace. 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
<table>
<thead>
<tr>
<th>LOCAL</th>
<th>REMOTE</th>
<th>DIRECTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>user1</td>
<td>user2@cluster2</td>
<td>export</td>
</tr>
<tr>
<td>user3</td>
<td>user6@cluster2</td>
<td>export</td>
</tr>
</tbody>
</table>
End UserMap

On cluster2:
Begin UserMap
<table>
<thead>
<tr>
<th>LOCAL</th>
<th>REMOTE</th>
<th>DIRECTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>user2</td>
<td>user1@cluster1</td>
<td>import</td>
</tr>
<tr>
<td>user6</td>
<td>user3@cluster1</td>
<td>import</td>
</tr>
</tbody>
</table>
End UserMap

Cluster1 configures user1 to run jobs as user2 and user3 to run jobs as user6.
Cluster2 configures user1 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 `badmin 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 lsadmin (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_msgrcv (%f)
Number of messages received

ru_nsignals (%f)
Number of signals received  
\texttt{ru\_nvcsw} (%f)  
Number of voluntary context switches  
\texttt{ru\_nivcsw} (%f)  
Number of involuntary context switches  
\texttt{ru\_exutime} (%f)  
Exact user time used (valid only on ConvexOS)
lsf.cluster

- About lsf.cluster
- Parameters section
- ClusterAdmins section
- Host section
- ResourceMap section
- RemoteClusters section

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 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.

Parameters

- `ADJUST_DURATION`
- `ELIM_ABORT_VALUE`
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 lslave(1) and lsloadadj(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 `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_CLIENTS_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_CLIENTS_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:fffe::88bb:*

Expands to 3ffe:fffe:0:0:0:88bb:*.* All IPv6 client hosts belonging to domains starting with 3ffe:fffe::88bb:* are allowed. No IPv4 hosts are allowed.

FLOAT_CLIENTS_ADDR_RANGE = 3ffe-fff: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. All IPv4 client hosts belonging to domains starting with 12.23.45 are allowed.

FLOAT_CLIENTS_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. 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.
**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.
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 elim executables on a host. Defining this parameter prevents an ongoing restart loop in the case of a faulty elim. The MELIM waits the LSF_ELIM_BLOCKTIME to receive a complete load update string before restarting the elim. The MELIM does not restart any elim 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 elim executables.

Valid values

Non-negative integers.

Default

Undefined; the number of elim 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 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, 1080:0:0:8:800:20fc:* 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 l m. 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:fffe:0:0:88bb:* All IPv6 client hosts belonging to domains starting with 3ffe:fffe::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 ff.fe: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_SHAREDIR/cluster_name. If the primary administrator is changed, make sure the owner of the configuration files and the files under LSB_SHAREDIR/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 `lsclusters` 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

Isfadmin

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:

<table>
<thead>
<tr>
<th>BEGIN Host</th>
</tr>
</thead>
<tbody>
<tr>
<td>HOSTNAME</td>
</tr>
<tr>
<td>hostA</td>
</tr>
<tr>
<td>hostD</td>
</tr>
<tr>
<td>hostD</td>
</tr>
<tr>
<td>hostE</td>
</tr>
<tr>
<td>END Host</td>
</tr>
</tbody>
</table>

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_Session_Scheduler` 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 host E:

```
Begin Host
HOSTNAME   model    type   server r1m 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 r1m mem swp RESOURCES #Keywords
hostA ! ! 1 3.5 () () (mg elim res patch rev=3 owner=user1)
hostB ! ! 1 3.5 () () (spec man=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 lslsload -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 `lsload` (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

<table>
<thead>
<tr>
<th>CLUSTERNAME</th>
<th>EQUIV</th>
<th>CACHE_INTERVAL</th>
<th>RECV_FROM</th>
<th>AUTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>cluster1</td>
<td>Y</td>
<td>60</td>
<td>Y</td>
<td>KRB</td>
</tr>
<tr>
<td>cluster2</td>
<td>N</td>
<td>60</td>
<td>Y</td>
<td>-</td>
</tr>
<tr>
<td>cluster4</td>
<td>N</td>
<td>60</td>
<td>N</td>
<td>PKI</td>
</tr>
</tbody>
</table>

End RemoteClusters

CLUSTERNAME

Description

Remote cluster name
Defines the Remote Cluster list. Specify the clusters you want the local cluster to 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

File properties

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_max_avail S s_peak s_max_avail B b_peak 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 (%)**

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 (%)**

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 281288c606a50065ea0e0f3e7161972c56491dc
1128372185 LSF_MANAGER 8.0 E 8 0 S 0 2 B 0 10 OVERUSE fb439ee293821761af9ed0785 1128372185
LSF_MANAGER 8.0 E hostA OVERUSE 2d22a06d65cfd5aba40875c2cb8544444a5
```
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:

- `lsadmin reconfig` to reconfigure LIM
- `badmin mbdrrestart` to restart `mbatchd`
- `badmin 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=number of processors`
- `EGO_DEFINE_NCPUS=cores=number of processors x number of cores`
- `EGO_DEFINE_NCPUS=threads=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 ncpus refers to the number of slots instead of the number of CPUs, however lshosts output will continue to show ncpus 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.cluster file) 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/misc/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:

<table>
<thead>
<tr>
<th>JOBID</th>
<th>USER</th>
<th>STAT</th>
<th>QUEUE</th>
<th>FROM_HOST</th>
<th>EXEC_HOST</th>
<th>JOB_NAME</th>
<th>SUBMIT_TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>111</td>
<td>user1</td>
<td>RUN</td>
<td>normal</td>
<td>hostA</td>
<td>hostB</td>
<td>myjob</td>
<td>Oct 22 09:22</td>
</tr>
</tbody>
</table>

```bash
echo $?
0
```

Job 111 is running, and job 123 does not exist:

```bash
bjobs 111 123
```

<table>
<thead>
<tr>
<th>JOBID</th>
<th>USER</th>
<th>STAT</th>
<th>QUEUE</th>
<th>FROM_HOST</th>
<th>EXEC_HOST</th>
<th>JOB_NAME</th>
<th>SUBMIT_TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>111</td>
<td>user1</td>
<td>RUN</td>
<td>normal</td>
<td>hostA</td>
<td>hostB</td>
<td>myjob</td>
<td>Oct 22 09:22</td>
</tr>
</tbody>
</table>

```bash
Job <123> is not found
echo $?
255
```

Job 111 is finished:

```bash
bjobs 111
```

```bash
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:

```bash
bjobs
```

```bash
No unfinished job found
echo $?
0
```

Job 123 does not exist:

```bash
bjobs 123
```

```bash
Job <123> is not found
echo $?
0
```

Job 111 is running:

```bash
bjobs 111
```

<table>
<thead>
<tr>
<th>JOBID</th>
<th>USER</th>
<th>STAT</th>
<th>QUEUE</th>
<th>FROM_HOST</th>
<th>EXEC_HOST</th>
<th>JOB_NAME</th>
<th>SUBMIT_TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>111</td>
<td>user1</td>
<td>RUN</td>
<td>normal</td>
<td>hostA</td>
<td>hostB</td>
<td>myjob</td>
<td>Oct 22 09:22</td>
</tr>
</tbody>
</table>

```bash
echo $?
0
```
Job 111 is running, and job 123 does not exist:

```
bjobs 111 123
```

<table>
<thead>
<tr>
<th>JOBID</th>
<th>USER</th>
<th>STAT</th>
<th>QUEUE</th>
<th>FROM_HOST</th>
<th>EXEC_HOST</th>
<th>JOB_NAME</th>
<th>SUBMIT_TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>111</td>
<td>user1</td>
<td>RUN</td>
<td>normal</td>
<td>hostA</td>
<td>hostB</td>
<td>myjob</td>
<td>Oct 22 09:22</td>
</tr>
</tbody>
</table>

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, `sbatchd` 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 `LSF_CMD_LOG_MASK`. To specify the logging level of error messages for LSF daemons, use `LSF_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_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/configdir 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

    LSB_CONFDIR/lsbatch

See also

    LSB_CONFDIR, LSF_LIVE_CONFDIR

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_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 checkpoint utility programs that sbatchd 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, LSB_DEBUG_LIM, LSB_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
```

Platform LSF Configuration Reference 327
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. In addition, **LSF_SERVERDIR** (lsf.conf) must be defined.

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_CHKPN T** and **LC2_CHKPN T**: 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** 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_LICENCE** 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_MULT1** and **LC2_MULT1**: 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_PREEM PT and LC2_PREEM PT: 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, LSB_DEBUG_LIM, LSB_DEBUG_RES, LSB_LIM_PORT, LSB_RES_PORT, LSB_MBD_PORT, LSB_SBD_PORT, LSB_LOGDIR, LSB_LIM_DEBUG, LSB_RES_DEBUG

LSB_DEBUG_MBD

Syntax

LSB_DEBUG_MBD=log_class

Description
Sets the debugging log class for mbatchd.

Specifies the log class filtering to be applied to mbatchd. 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 badmin 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 sbatchd.

Specifies the log class filtering to be applied to sbatchd. 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 `badmin sbdd debug` 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, badmin

**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
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 (badmin 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
- lslsd displays the host status `ok`
- bhosts displays the host status `closed`
- Users can run tasks on the host using lsrn or lsgn. 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 timestamp displayed by the commands `bjobs -l`, `bacct -l`, and `bhist -l`.

Default

`N`

LSB_ECHKPNT_KEEP_OUTPUT

Syntax

```
LSB_ECHKPNT_KEEP_OUTPUT=y | Y
```

Description

Saves the standard output and standard error of custom `echkpnt` and `erestart` methods to:

- `checkpoint_dir/$LSB_JOBID/echkpnt.out`
- `checkpoint_dir/$LSB_JOBID/echkpnt.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 `echkpnt` 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 `echkpnt` 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 `echkpnt` and `erestart` programs. You must assign your custom `echkpnt` and `erestart` programs the name `echkpnt.method_name` and `erestart.method_name`. The programs `echkpnt.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 `echkpnt` 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 `echkpnt.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 `echkpnt` applications with the same name such as `echkpnt.my_method` 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 `echkpnt.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 `echkpnt` and `erestart` programs are located.

The checkpoint method directory should be accessible by all users who need to run the custom `echkpnt` and `erestart` programs.

Can also be defined as an environment variable.

**Default**

Not defined. LSF searches in `LSF_SERVERDIR` for custom `echkpnt` 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 names 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

Platform LSF Configuration Reference 335
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**

```plaintext
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 lsb.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**

```plaintext
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 lsb.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 lsb.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`.

The setting of `LSB_JOB_CPULIMIT` has the following effect on how the limit is enforced:

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>Enabled disabled</td>
</tr>
<tr>
<td>n</td>
<td>Disabled enabled</td>
</tr>
<tr>
<td>Not defined</td>
<td>Enabled enabled</td>
</tr>
</tbody>
</table>

Default

Not defined

Notes

To make `LSB_JOB_CPULIMIT` take effect, use the command `badmin hrestart all` to restart all `sbatchd` s 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 Administrating 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 `lsb.queues`.

The setting of `LSB_JOB_MEMLIMIT` has the following effect on how the limit is enforced:

<table>
<thead>
<tr>
<th>When <code>LSB_JOB_MEMLIMIT</code> is</th>
<th>LSF-enforced per-job limit</th>
<th>OS-enforced per-process limit</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>Enabled</td>
<td>Disabled</td>
</tr>
<tr>
<td>n or not defined</td>
<td>Disabled</td>
<td>Enabled</td>
</tr>
</tbody>
</table>

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 `lsb.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 `badmin hrestart all` to restart all `sbatchd` 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 `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_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, lsb.queues, bsub`
- `JOB_TERMINATE_INTERVAL in lsb.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.
**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_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 mbactchd and sbatchd when modifying **LSB_LOCALDIR**.

**Example**

```
LSB_LOCALDIR=/usr/share/lsbatch/loginfo
```

**Default**

Not defined

**See also**

**LSB_SHAREDIR**, **EVENT_UPDATE_INTERVAL** in **lsb.params**
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_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 lsf.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 lsmail.exe mail program,
which supports SMTP and Microsoft Exchange Server protocols on Windows. If lsmail 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

<table>
<thead>
<tr>
<th>LSB_MAILPROG=lsmail.exe</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSB_MAILPROG=/serverA/tools/lsf/bin/unixhost.exe</td>
</tr>
</tbody>
</table>

Default

<table>
<thead>
<tr>
<th>/usr/lib/sendmail (UNIX)</th>
</tr>
</thead>
<tbody>
<tr>
<td>blank (Windows)</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>LSB_MAILSERVER=EXCHANGE:<a href="mailto:Host2@company.com">Host2@company.com</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>LSB_MAILSERVER=SMTP:MailHost</td>
</tr>
</tbody>
</table>

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_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 `sbatchd` 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.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

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 lsb.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 mbatchd if you change the value of LSB_MAX_NQS_QUEUES.

The total number of NQS queues configured by NQS_QUEUES in lsb.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
**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 in the job submission file. However, while `mbatchd` is processing a pack, `mbatchd` is blocked from processing other requests, so increasing pack size can affect `mbatchd` response time for other job submissions.

If you change the configuration of this parameter, you must restart `mbatchd`.

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 `sbatchd` instances can be polled by `mbatchd` in the interval `MBD_SLEEP_TIME/10` (6 seconds by default). Use this parameter in large clusters to reduce the time it takes for `mbatchd` to probe all `sbatchds`.

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 `badmin mbdrestart` to restart `mbatchd` and let the modified value take effect.

If `LSB_MAX_PROBE_SBD` is defined, the value of `MAX_SBD_FAIL` in `lsb.params` can be less than 3.
Valid values
Any positive integer between 0 and 64

Default
20

See also
MAX_SBD_FAIL in lsb.params

LSB_MBD_BUSY_MSG

Syntax

`LSB_MBD_BUSY_MSG="message_string"`

Description
Specifies the message displayed when mbatchd 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 mbatchd 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 mbatchd 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 `mbatchd` 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 `bhosts` command when `mbatchd` 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 `mbatchd` 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_CHKPTNT_RERUN

Syntax

```
LSB_MC_CHKPTNT_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

\texttt{LSB\_MIXED\_PATH\_DELIMITER}=	exttt{"\textpipe\"}

Description

Defines the delimiter between UNIX and Windows paths if \texttt{LSB\_MIXED\_PATH\_ENABLE} = \texttt{y}. For example, /home/tmp/J.out|c:\tmp\J.out.

Default

A pipe \\textpipe is the default delimiter.

See also

\texttt{LSB\_MIXED\_PATH\_ENABLE}

LSB_MIXED_PATH_ENABLE

Syntax

\texttt{LSB\_MIXED\_PATH\_ENABLE}={\texttt{y}} | {\texttt{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 \texttt{unix\_path\_cmd|windows\_path\_cmd}.

Applies to the following options of \texttt{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:

\texttt{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: \texttt{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 (stdout) 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 `badm mbdrrestart` 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`
**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 mbatchd.

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 mbatchd performance when you use the bjobs command.

This may indirectly increase overall mbatchd performance.

The `port_number` is the TCP/IP port number to be used by mbatchd to only service query requests from the LSF system. mbatchd checks the query port during initialization.

If LSB_QUERY_PORT is not defined:

- mbatchd 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, mbatchd forks one child mbatchd to service the request. Each child mbatchd processes one request and then exits.

If LSB_QUERY_PORT is defined:

- mbatchd prepares this port for connection. The default behavior of mbatchd changes, a child mbatchd is forked, and the child mbatchd creates threads to process requests.
- mbatchd responds to requests by forking one child mbatchd. As soon as mbatchd has forked a child mbatchd, the child mbatchd takes over and listens on the port to process more query requests. For each request, the child mbatchd creates a thread to process it.

The interval used by mbatchd for forking new child mbatchds is specified by the parameter MBD_REFRESH_TIME in lsb.params.

The child mbatchd 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 lsb.params has passed (see MBD_REFRESH_TIME in lsb.params for more details). When any of these happens, the parent mbatchd sends a message to the child mbatchd to exit.

LSB_QUERY_PORT must be defined when NEWJOB_REFRESH=Y in lsb.params to enable a child mbatchd to get up to date information about new jobs from the parent mbatchd.

Operating system support

Tip:
See the Online Support area of the Platform Computing Web site at www.platform.com for the latest information about operating systems that support multithreaded mbatchd.

**Default**

Not defined

**See also**

MBD_REFRESH_TIME and NEWJOB_REFRESH in lsb.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 lsb.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 `mbatchd` during fairshare accounting at job submission by limiting the number of share account nodes created on `mbatchd` startup. Most useful when there are a lot of user groups with all members 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 `mbatchd`.

Default

```
N
```

**LSB_SBD_PORT**

See `LSF_LIM_PORT`, `LSF_RES_PORT`, `LSB_MBD_PORT`, `LSB_SBD_PORT`.
LSB_SET_TMPDIR

Syntax

\[ \text{LSB\_SET\_TMPDIR}=y \mid n \]

If \( y \), LSF sets the \text{TMPDIR} environment variable, overwriting the current value with /tmp/job_ID.tmpdir.

Default

\( n \)

LSB_SHAREDIR

Syntax

\[ \text{LSB\_SHAREDIR}=\text{directory} \]

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 LSB_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

\[ \text{LSB\_SHORT\_HOSTLIST}=1 \]

Description

Displays an abbreviated list of hosts in \text{bjobs} and \text{bhist} for a parallel job where multiple processes of a job are running on a host. Multiple processes are displayed in the following format:

\[ \text{processes*hostA} \]

For example, if a parallel job is running 5 processes on hostA, the information is displayed in the following manner:

\[ 5*hostA \]

Setting this parameter may improve \text{mbatchd} restart performance and accelerate event replay.
Default

Not defined

**LSB_SIGSTOP**

Syntax

```bash
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

```bash
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:

```bash
bsub netscape ...
```

Causes `sb` 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**

```bash
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 `sbatchd` on the execution host to make changes take effect.

**Default**

N

**LSB_TIME_MBD**

**Syntax**

```bash
LSB_TIME_MBD=timing_level
```

**Description**

The timing level for checking how long `mbatchd` 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_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_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, LSB_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

```bash
LSB_TIME_SBD=timing_level
```

Description

The timing level for checking how long `sbatch` 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_UTMP**

Syntax

```bash
LSB_UTMP=y | Y
```

Description

If set, enables registration of user and account information for interactive batch jobs submitted with `bsub -Ip` or `bsub -Is`.

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 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 -l` are not associated with a pseudo-terminal, `utmp` file registration is not supported for these jobs.

**Default**

Not defined

**LSF_AFS_CELLNAME**

**Syntax**

```bash
LSF_AFS_CELLNAME=AFS_cell_name
```

**Description**

Must be defined to AFS cell name if the AFS file system is in use.

**Example:**

```bash
LSF_AFS_CELLNAME=xxx.ch
```

**Default**

Not defined

**LSF_AM_OPTIONS**

**Syntax**

```bash
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 AM FIRST

**LSF_API_CONNTIMEOUT**

**Syntax**

```bash
LSF_API_CONNTIMEOUT=time_seconds
```

Platform LSF Configuration Reference 365
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_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

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.

\textbf{eauth}

For site-specific customized external authentication. Provides the highest level of security of all LSF authentication methods.

\textbf{ident}

For authentication using the RFC 931/1413/1414 protocol to verify the identity of the remote client. If you want to use \texttt{ident} authentication, you must download and install the \texttt{ident} protocol, available from the public domain, and register \texttt{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.

\textbf{Restriction:}

To enable privileged ports authentication, LSF_AUTH must not be defined; setuid is not a valid value for LSF_AUTH.

Default

\texttt{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_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_BIND_JOB**

**Syntax**

```bash
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_BINDIR**

**Syntax**

```bash
LSF_BINDIR=directory
```

**Description**

Directory in which all LSF user commands are installed.

Default

`LSF_MACHDEP/bin`

**LSF_BMPLUGIN**

**Syntax**

```bash
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_LOG_MASK**

**Syntax**

```plaintext
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.

In addition, `LSF_SERVERDIR` (lsf.conf) must be defined for successful error message logging.

**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, LSB_LOG_MASK, LSB_LOGDIR, LSB_TIME_CMD`
LSF_CMD_LOGDIR

Syntax

\texttt{LSF\_CMD\_LOGDIR=\textit{path}}

Description

The path to the log files used for debugging LSF commands.
This parameter can also be set from the command line.

Default

\texttt{/tmp}

See also

\texttt{LSB\_CMD\_LOG\_MASK}, \texttt{LSB\_CMD\_LOGDIR}, \texttt{LSB\_DEBUG}, \texttt{LSB\_DEBUG\_CMD}, \texttt{LSB\_TIME\_CMD}, \texttt{LSF\_CMD\_LOG\_MASK}, \texttt{LSF\_LOG\_MASK}, \texttt{LSF\_LOGDIR}, \texttt{LSF\_TIME\_CMD}

LSF_CONF_RETRY_INT

Syntax

\texttt{LSF\_CONF\_RETRY\_INT=\textit{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

\texttt{EGO\_CONF\_RETRY\_INT}

Default

30

See also

\texttt{LSF\_CONF\_RETRY\_MAX}

LSF_CONF_RETRY_MAX

Syntax

\texttt{LSF\_CONF\_RETRY\_MAX=\textit{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, sdb, 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_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, mbatchd, sbatchd, and RES run the executable `daemons.wrap` located in LSF_SERVERDIR.

Default

Not defined. LSF does not run the `daemons.wrap` executable.

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 `mbatchd` 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"
```

then `mbatchd` processes run only on CPU number 1 on the master host, and `mbschd` run on only on CPU number 2.

If you specify

```
LSF_DAEMONS_CPUS="1 2:1 2"
```

both `mbatchd` 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_DEBUG_CMD

Syntax

```
LSF_DEBUG_CMD=log_class
```
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. In addition, `LSF_SERVERDIR` (lsf.conf) must be defined.

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_CHKPT` and `LC2_CHKPT`: 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_LICENCE` 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_LICENCE is also supported for backward compatibility)
LC_LIC_SCHED - 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 badmin 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 lsrun and lsgrun 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 lsmake 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 # Pisa # Venezia # Bologna
15/3 19:4:50 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

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, mbatchd 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_DYNAMIC_HOST_WAIT_TIME** in `lsf.conf`
- **LSF_HOST_ADDR_RANGE** in `lsf.cluster.cluster_name`

**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.

**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/inittab) 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 csabuild 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 csaswitch command to turn on the configuration changes in /etc/csa.conf.

   **Note:**
   See the IRIX resource administration documentation for information about the csaswitch 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_EXTSCHEDULER**

Syntax

```
LSF_ENABLE_EXTSCHEDULER=y | Y
```

Description

If set, enables `mbatchd` external scheduling for LSF HPC features.
**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

**See also**

- **LSF_DUALSTACK_PREFER_IPV6**
- **LSF_ENVDIR**

**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 `badmin mbdrestart` and `badmin 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 usage based on each host’s usage and the total usage being charged to the execution host. This host usage breakdown applies to the laun ch 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 usage 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 usage will not work. If you add or remove this extension, you must restart `mbatchd`, `sbatchd`, and `res` 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 (%)`
  - `execHosts (%)`
• 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 bjobs to omit all but the first process ID (PID) for a job. bjobs displays only the first ID and a count of the process group IDs (PGIDs) and process IDs for the job.
Without SHORT_PIDLIST, bjobs -l displays all the PGIDs and PIDs for the job. With SHORT_PIDLIST set, bjobs -l displays a count of the PGIDs and PIDs.

TASK_MEMLIMIT : Enables enforcement of a memory limit (bsub -M, bmod -M, or MEMLIMIT in lsb.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 lsb.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 lsb.events:

```
"JOB_START" "8.0" 1058998981 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" "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"
"hostB" "hostB" "hostB" "hostB" "hostB" "hostB" "hostB" "hostB" "hostB" "hostB" "hostB" "hostB"
"hostB" "hostB" "hostB" "hostB" "hostB" "hostB" "hostB" "hostB" "hostB" "hostB" "hostB" "hostB"
```

With SHORT_EVENTFILE, a JOB_START event would be logged in lsb.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 lsb.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" "hostA" "hostA" "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" "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" "hostB" "hostB" "hostB" "hostB"
"sleep 100" 0.079999 0.279999 0 0 -1 0 0 0 0 0 -1 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 -1 0 0 0 0 -1
"default" 0 64 ** 0 4304 6024 ** ** ** 0
```

Example bjobs -l output without SHORT_PIDLIST:

```
bjobs -l
```

displays all the PGIDs and PIDs for the job:

```
Job <109>, User <user3>, Project <default>, Status <RUN>, Queue <normal>, Interactive 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>;


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
```

<< 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>, Interactive 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>;
    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   -  -
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

\[ \text{LSF\_ID\_PORT} = \text{port\_number} \]

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

\[ \text{LSF\_INCLUDEDIR} = \text{directory} \]

Description

Directory under which the LSF API header files `lsf.h` and `lsbatch.h` are installed.

Default

\[ \text{LSF\_INDEP/include} \]

See also

LSF_INDEP

LSF_INDEP

Syntax

\[ \text{LSF\_INDEP} = \text{directory} \]

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.

- \[ \text{LSB\_SHAREDIR} = \$\text{LSF\_INDEP/work} \]
- \[ \text{LSF\_CONFDIR} = \$\text{LSF\_INDEP/conf} \]
- \[ \text{LSF\_INCLUDEDIR} = \$\text{LSF\_INDEP/include} \]
- \[ \text{LSF\_MANDIR} = \$\text{LSF\_INDEP/man} \]
- \[ \text{XLSF\_APPDIR} = \$\text{LSF\_INDEP/misc} \]
Default

/usr/share/lsf/mnt

See also

LSF_MACHDEP, LSB_SHARDEDIR, 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 -Ip`
- `bsub -Is`
- `lsrun`
- `lsgrun`
- `lsmake` (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`. If you use `stderr` as a terminal, 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`

LSF_LD_SECURITY

Syntax

```
LSF_LD_SECURITY=y | n
```

Description

`LSF_LD_SECURITY`: When set, jobs submitted using `bsub -Is` or `bsub -Ip` 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 (bld) 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 lsb.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:
    ```bash
    LSF_LICENSE_FILE=/usr/share/lsf/cluster1/conf/license.dat
    ```
  - Windows examples:
    ```bash
    LSF_LICENSE_FILE=C:\licenses\license.dat
    LSF_LICENSE_FILE=\HostA\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:
  ```bash
  LSF_LICENSE_FILE="1700@hostD"
  ```
- For a license with redundant servers, use a comma (,) to separate each `port@host_name` or configure the full path of the license file. The port number must be the same as that specified in the SERVER line of the license file. For example:
  ```bash
  UNIX:
  LSF_LICENSE_FILE="port@hostA, port@hostB, port@hostC"
  ```
  ```bash
  Windows:
  LSF_LICENSE_FILE="C:\licenses\license.dat, C:\licenses\license.dat, C:\licenses\license.dat"
  ```
Windows:

- 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:

```python
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/flexlm/`.

#### LSF_LICENSE_MAINTENANCE_INTERVAL

**Syntax**

```bash
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 limit 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
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, `lsadmin reconfig` 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 log files for non-master hosts because

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 badmin mbdrestart and lsadmin reconfig 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**
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...
```

402 Platform LSF Configuration Reference
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_vemk and p-em 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_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
- pim.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:
  ```plaintext
  HKEY_LOCAL_MACHINE\SOFTWARE\Platform Computing Corporation\LSF\cluster_name\LSF_LOGDIR
  ```

- On Windows XP x64 and Windows 2003 x64:
  ```plaintext
  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 syslogd 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_LOG_DIR, 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
- eeventd.log. host_name
- pim.log. host_name

LSF_LOGDIR_USE_WIN_REG

Syntax

```plaintext
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_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 `lsadmin reconfig` 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_MAST...so that the primary master host is last. For example:

```
LSF_MAST...="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

**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.
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_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/misc`

LSF_MONITOR_LICENSE_TOOL

Syntax

```
LSF_MONITOR_LICENSE_TOOL=y | Y
```

Description

Specify Y to enable data collection by `lim` for the command option `lsadmin lsflic`.

Default

Not defined. `lim` ignores requests from `lsadmin`, closing the channel.
LSF_NIOS_DEBUG

Syntax

\texttt{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 nios.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/nios.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

\texttt{LSF\_NIOS\_ERR\_LOGDIR=directory}

Description

Applies to Windows only.

If LSF\_NIOS\_ERR\_LOGDIR is specified, logs NIOS errors to directory/nios.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\_TMP\_DIR
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

\[ \text{LSF\_NIOS\_JOBSTATUS\_INTERVAL} = \text{time\_minutes} \]

Description

Applies only to interactive batch jobs.
Time interval at which NIOS polls mbatchd 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, mbatchd cannot communicate with NIOS and the return code of a job is not retrieved.
When this parameter is defined, before exiting, NIOS polls mbatchd on the interval defined by LSF_NIOS_JOBSTATUS_INTERVAL to check if a job is still running. NIOS continues to poll mbatchd until it receives an exit code or mbatchd 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 mbatchd on the defined interval to confirm that a job is still pending and this may add load to mbatchd.

See also

Environment variable LSF_NIOS_PEND_TIMEOUT

LSF_NIOS_MAX_TASKS

Syntax

\[ \text{LSF\_NIOS\_MAX\_TASKS} = \text{integer} \]
**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:

```bash
bsub -n 32 -R "span[ptile=4]" pam yourOpenMPJob
```

Where defined

This parameter can alternatively be set as an environment variable. For example:

```bash
setenv LSF_PAM_HOSTLIST_USE unique
```

**LSF_PAM_PLUGINDIR**

**Syntax**

```bash
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**

```bash
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**

```bash
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 `LSFlim` 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**

```bash
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**

```bash
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

\texttt{LSF\_PIM\_SLEEPTIME=\texttt{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 \texttt{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

\texttt{EGO\_PIM\_SLEEPTIME}

Default

30 seconds

LSF_PIM_SLEEPTIME_UPDATE

Syntax

\texttt{LSF\_PIM\_SLEEPTIME\_UPDATE=\texttt{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 \texttt{sbatchd} and PIM on the same host.

When this parameter is not defined or set to \texttt{n}, \texttt{sbatchd} queries PIM as needed for job process information.

When this parameter is defined, \texttt{sbatchd} does not query PIM immediately as it needs information; \texttt{sbatchd} only queries PIM every \texttt{LSF\_PIM\_SLEEPTIME\_UPDATE} seconds.

Limitations

When this parameter is defined:

- \texttt{sbatchd} may be intermittently unable to retrieve process information for jobs whose run time is smaller than \texttt{LSF\_PIM\_SLEEPTIME}.
- It may take longer to view resource usage with \texttt{bjobs -l}.

EGO parameter

\texttt{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, -is, -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 lsadmin command (see lsadmin) 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, \texttt{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**

\texttt{LSF\_RES\_CONNECT\_RETRY=integer | 0}

**Description**

The number of attempts by RES to reconnect to NIOS.

If \texttt{LSF\_RES\_CONNECT\_RETRY} is not defined, the default value is used.

**Default**

0

**See also**

\texttt{LSF\_NIOS\_RES\_HEARTBEAT}

**LSF\_RES\_DEBUG**

**Syntax**

\texttt{LSF\_RES\_DEBUG=1 | 2}

**Description**

Sets RES to debug mode.

If \texttt{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 \texttt{LSF\_RES\_PORT} has been defined.

Specify 1 for this parameter unless you are testing RES.

**Valid values**

- \texttt{LSF\_RES\_DEBUG=1}
- RES runs in the background with no associated control terminal.
- \texttt{LSF\_RES\_DEBUG=2}
- RES runs in the foreground and prints error messages to \texttt{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:
• badmin hstartup
• bpeek
• lsadmin limstartup
• lsadmin resstartup
• lsf restart
• lsf shutdown
• lsf startup
• lsrd
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_config` file or `$HOME/.ssh/config`.

See also

`ssh`, `ssh_config`

**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 llim, res, nios, sbatchd, mbatchd, 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 lstcsh only. Specifies users who are allowed to use @ for host redirection. Users not specified with this parameter cannot use host redirection in lstcsh. 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 lstcsh.

Default
Not defined

**LSF_SHIFT_JIS_INPUT**

**Syntax**

```bash
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**

```bash
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.
LSF_STRICT_RESREQ

Syntax

\[ \text{LSF\_STRICT\_RESREQ=} \{ \text{Y} | \text{N} \} \]

Description

When \( \text{LSF\_STRICT\_RESREQ=} \text{Y} \), the resource requirement selection string must conform to the stricter resource requirement syntax described in \textit{Administering Platform LSF}. The strict resource requirement syntax only applies to the \texttt{select} section. It does not apply to the other resource requirement sections (\texttt{order}, \texttt{rusage}, \texttt{same}, \texttt{span}, or \texttt{cu}).

When \( \text{LSF\_STRICT\_RESREQ=} \text{Y} \) in \texttt{lsf.conf}, LSF rejects resource requirement strings where an \texttt{rusage} section contains a non-consumable resource.

When \( \text{LSF\_STRICT\_RESREQ=} \text{N} \), the default resource requirement selection string evaluation is performed.

Default

\text{N}

LSF_STRIP_DOMAIN

Syntax

\[ \text{LSF\_STRIP\_DOMAIN=} \text{domain\_suffix}[:\text{domain\_suffix} ...] \]

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 \texttt{LSF\_STRIP\_DOMAIN},

\[ \text{LSF\_STRIP\_DOMAIN=} .\text{foo.com} : .\text{bar.com} \]

LSF accepts \texttt{hostA}, \texttt{hostA.\text{foo.com}}, and \texttt{hostA.\text{bar.com}} as names for \texttt{hostA}, and uses the name \texttt{hostA} in all output. The leading period \( . \) is required.

Example:

\[ \text{LSF\_STRIP\_DOMAIN=} .\text{platform.com} : .\text{generic.com} \]

In the above example, LSF accepts \texttt{hostA}, \texttt{hostA.\text{platform.com}}, and \texttt{hostA.\text{generic.com}} as names for \texttt{hostA}, and uses the name \texttt{hostA} in all output.

After adding or changing \texttt{LSF\_STRIP\_DOMAIN}, use \texttt{lsadmin reconfig} and \texttt{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_CMD, LSB_TIME_MBD, LSB_TIME_SBD, 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_LIM, LSF_TIME_RES`

**LSF_TIME_RES**

**Syntax**

`LSF_TIME_RES = timing_level`
lsf.conf

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 badmin 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 \texttt{blaunch}

By default, LSF creates a temporary directory for a job only on the first execution host. If LSF_TM_PDIR is set in \texttt{lsf.conf}, the path of the job temporary directory on the first execution host is set to \texttt{LSF_TMPDIR/job_ID.tmpdir}.

If \texttt{LSB_SET_TMPDIR=Y}, the environment variable \texttt{TMPDIR} will be set equal to the path specified by \texttt{LSF_TMPDIR}.

Tasks launched through the \texttt{blaunch} distributed application framework make use of the LSF temporary directory specified by \texttt{LSF_TM_PDIR}:

\begin{itemize}
  \item When the environment variable \texttt{TMPDIR} is set on the first execution host, the \texttt{blaunch} framework propagates this environment variable to all execution hosts when launching remote tasks.
  \item The job RES or the task RES creates the directory specified by \texttt{TMPDIR} if it does not already exist before starting the job.
  \item The directory created by the job RES or task RES has permission 0700 and is owned by the execution user.
  \item If the \texttt{TMPDIR} directory was created by the task RES, LSF deletes the temporary directory and its contents when the task is complete.
  \item If the \texttt{TMPDIR} directory was created by the job RES, LSF will delete the temporary directory and its contents when the job is done.
  \item If the \texttt{TMPDIR} directory is on a shared file system, it is assumed to be shared by all the hosts allocated to the \texttt{blaunch} job, so LSF does not remove \texttt{TMPDIR} directories created by the job RES or task RES.
\end{itemize}

Default

By default, LSF_TM_PDIR is not enabled. If LSF_TM_PDIR is not specified in \texttt{lsf.conf}, this parameter is defined as follows:

\begin{itemize}
  \item On UNIX: $TMPDIR/job_ID.tmpdir or /tmp/job_ID.tmpdir
  \item On Windows: %TMP%, %TEMP, or %SystemRoot%
\end{itemize}

\texttt{LSF_ULDB_DOMAIN}

Syntax

\texttt{LSF_ULDB_DOMAIN="domain_name ..."}

Description

\texttt{LSF_ULDB_DOMAIN} specifies the name of the LSF domain in the ULDB domain directive. A domain definition of name \texttt{domain_name} must be configured in the SGI IRIX \texttt{jlimit.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 jlimit.in file.

**LSF resource usage limits controlled by ULDB**

- **PROCESSLIMIT**: 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 the resource usage limits:

- **core**
memory
stack
swap

When set, LSF_UNIT_FOR_LIMITS applies cluster-wide to these 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 these limits when modified with bmod, and the display of these resource usage limits in query commands (bacct, bapp, bhist, bhosts, bjobs, bqueues, lsload, 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.

**Note:**

Other limits (such as the file limit) are not affected by setting the parameter LSF_UNIT_FOR_LIMITS.

**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/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

LSF_USER_DOMAIN

Syntax

```bash
LSF_USER_DOMAIN=domain:domain:domain...
```

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

```bash
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**

```plaintext
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/libmpirm.sl`
- 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**

```plaintext
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

\[ \text{XLSF\_APPDIR} = \text{directory} \]

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

\[ \text{LSF\_INDEP/misc} \]

XLSF_UIDDIR

Syntax

\[ \text{XLSF\_UIDDIR} = \text{directory} \]

Description

(UNIX only) Directory in which Motif User Interface Definition files are stored. These files are platform-specific.

Default

\[ \text{LSF\_LIBDIR/uid} \]
lsf.licensescheduler

The lsf.licensescheduler file contains Platform License Scheduler configuration information. All sections except ProjectGroup are required. In cluster mode, the Project section is also not required.

Changing lsf.licensescheduler configuration

After making any changes to lsf.licensescheduler, run the following commands:

• bladmin reconfig to reconfigure bld

If you made the following changes to this file, you may need to restart mbatchd:

• Added or deleted any feature.
• Added or deleted projects in the DISTRIBUTION parameter of the Feature section.
• Changed the Project Group section.
• Changed the feature mode (for example, changed from cluster mode to project mode, or vice versa).

In these cases a message is written to the log file prompting the restart.

If you have added, changed, or deleted any Feature or Projects sections, you may need to restart mbatchd. In this case a message is written to the log file prompting the restart.

If required, run bladmin mbdrestart to restart each LSF cluster.

Parameters section

Description

Required. Defines Platform 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. Mandatory parameters are as follows:

Parameters

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADMIN=lsadmin</td>
</tr>
<tr>
<td>HOSTS=hostA hostB hostC</td>
</tr>
<tr>
<td>LMSTAT_PATH=/etc/flexlm/bin</td>
</tr>
<tr>
<td>LM_STAT_INTERVAL=30</td>
</tr>
<tr>
<td>PORT=9581</td>
</tr>
</tbody>
</table>

Parameters

• ADMIN
• AUTH
• CLUSTER_MODE
• DISTRIBUTION_POLICY_VIOLATION_ACTION
• ENABLE_INTERACTIVE
• HEARTBEAT_INTERVAL
lsf.licensescheduler

- HEARTBEAT_TIMEOUT
- HIST_HOURS
- HOSTS
- INUSE_FROM_RUSAGE
- LIB_RECVTIMEOUT
- LIB_CONNTIMEOUT
- LICENSE_FILE
- LM_REMOVE_INTERVAL
- LM_STAT_INTERVAL
- LM_STAT_TIMEOUT
- LMSTAT_PATH
- LOG_EVENT
- LOG_INTERVAL
- LS_DEBUG_BLC
- LS_DEBUG_BLD
- LS_ENABLE_MAX_PREEMPT
- LS_LOG_MASK
- LS_MAX_STREAM_FILE_NUMBER
- LS_MAX_STREAM_SIZE
- LS_MAX_TASKMAN_PREEMPT
- LS_MAX_TASKMAN_SESSIONS
- LS_STREAM_FILE
- LS_PREEMPT_PEER
- MBDF_HEARTBEAT_INTERVAL
- MBD_REFRESH_INTERVAL
- MERGE_BY_SERVICE_DOMAIN
- PEAK_INUSE_PERIOD
- PORT
- PREEMPT_ACTION
- STANDBY_CONNTIMEOUT
- BLC_HEARTBEAT_FACTOR

**ADMIN**

**Syntax**

`ADMIN=user_name...`

**Description**

Defines the Platform License Scheduler administrator using a valid UNIX user account. You can specify multiple accounts.

Used for both project mode and cluster mode.

**AUTH**

**Syntax**

`AUTH=Y`
Description

Enables Platform License Scheduler user authentication for projects for taskman jobs.

Used for both project mode and cluster mode.

**CLUSTER_MODE**

**Syntax**

```
CLUSTER_MODE=Y
```

**Description**

Enables cluster mode (instead of project mode) in Platform License Scheduler. Setting in individual Feature sections overrides the global setting in the Parameters section.

Cluster mode emphasizes high utilization of license tokens above other considerations such as ownership. License ownership and sharing can still be configured, but within each cluster instead of across multiple clusters. Preemption of jobs (and licenses) also occurs within each cluster instead of across clusters.

Cluster mode was introduced in Platform License Scheduler 8.0. Before cluster mode was introduced, project mode was the only choice available.

**Default**

Undefined (N). Platform License Scheduler runs in project mode.

**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 Platform License Scheduler checks for distribution policy violations.

**reporting_command**

Specify the keyword CMD with the directory path and command that Platform License Scheduler runs when reporting a violation.

**Description**

Optional. Defines how Platform License Scheduler handles distribution policy violations. Distribution policy violations are caused by non-LSF workloads; Platform License Scheduler explicitly follows its distribution policies.

Platform 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 Platform License Scheduler finds a distribution policy violation, it creates or overwrites the LSF_LOGDIR/bld.violation.service_domain_name.log 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. Platform 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. Platform License Scheduler allocates licenses equally to each cluster and does not distribute licenses for interactive tasks.

Used for project mode only.

**HEARTBEAT_INTERVAL**

**Syntax**

```
HEARTBEAT_INTERVAL=seconds
```

**Description**

The time interval between bld heartbeats indicating the bld is still running.
HEARTBEAT_TIMEOUT
Syntax

HEARTBEAT_TIMEOUT=seconds

Description
The time a slave bld waits to hear from the master bld before assuming it has died.

Default
60 seconds

HIST_HOURS
Syntax

HIST_HOURS=hours

Description
Determines the rate of decay the accumulated use value used in fairshare and preemption decisions. When HIST_HOURS=0, accumulated use is not decayed.

Accumulated use is displayed by the blstat command under the heading ACUM_USE.

Used for project mode only.

Default
5 hours. Accumulated use decays to 1/10 of the original value over 5 hours.

HOSTS
Syntax

HOSTS=host_name.domain_name ...

Description
Defines Platform License Scheduler hosts, including Platform License Scheduler candidate hosts.

Specify a fully qualified host name such as hostX.mycompany.com. You can omit the domain name if all your Platform License Scheduler clients run in the same DNS domain.

Used for both project mode and cluster mode.

INUSE_FROM_RUSAGE
Syntax

INUSE_FROM_RUSAGE=Y|N
**LIB_CONNTIMEOUT**

**Syntax**

```
LIB_CONNTIMEOUT=seconds
```

**Description**

Specifies a timeout value in seconds for communication between Platform License Scheduler and Platform LSF APIs. **LIB_CONNTIMEOUT=0** indicates no timeout.

**Default**

5 seconds

**LIB_RECVTIMEOUT**

**Syntax**

```
LIB_RECVTIMEOUT=seconds
```

**Description**

Specifies a timeout value in seconds for communication between Platform License Scheduler and LSF. Used for both project mode and cluster mode.

**Default**

5 seconds

**LICENSE_FILE**

**Syntax**

```
LICENSE_FILE=path
```
Description
Sets the path to the Platform License Scheduler license file containing a valid license for this product. Used for both project mode and cluster mode.

Default
Uses the path set in LSF_LICENSE_FILE in the LSF configuration file lsf.conf.

LM_REMOVE_INTERVAL
Syntax
LM_REMOVE_INTERVAL=seconds

Description
Specifies the minimum time a job must have a license checked out before lmremove can remove the license (using preemption). lmremove causes lmgrd and vendor daemons to close the TCP connection with the application, then retries the license checkout.

License Scheduler only considers preempting a job after this interval has elapsed. LM_REMOVE_INTERVAL overrides the LS_WAIT_TO_PREEMPT value if LM_REMOVE_INTERVAL is larger.

Used for project mode only.

Default
180 seconds

LM_STAT_INTERVAL
Syntax
LM_STAT_INTERVAL=seconds

Description
Defines a time interval between calls that Platform License Scheduler makes to collect license usage information from FlexNet license management.

Used for both project mode and cluster mode.

Default
60 seconds

LM_STAT_TIMEOUT
Syntax
LM_STAT_TIMEOUT=seconds
**Description**

Sets the timeout value passed to the `lmstat` command. The `Parameters` section setting is overwritten by the `ServiceDomain` setting, which is overwritten by the command line setting (`blcollect -t timeout`).

Used for both project mode and cluster mode.

**Default**

180 seconds

**LMSTAT_PATH**

**Syntax**

```plaintext
LMSTAT_PATH=path
```

**Description**

Defines the full path to the location of the FlexNet command `lmstat`.

Used for both project mode and cluster mode.

**LOG_EVENT**

**Syntax**

```plaintext
LOG_EVENT=Y
```

**Description**

Enables logging of Platform License Scheduler events in the `bld.stream` file.

**Default**

Not defined. Information is not logged.

**LOG_INTERVAL**

**Syntax**

```plaintext
LOG_INTERVAL=seconds
```

**Description**

The interval between token allocation data logs in the data directory.

**Default**

60 seconds

**LS_DEBUG_BLC**

**Syntax**

```plaintext
LS_DEBUG_BLC=log_class
```
Description

Sets the debugging log class for the Platform License Scheduler `blcollect` daemon.

Used for both project mode and cluster mode.

Specifies the log class filtering to be applied to `blcollect`. Only messages belonging to the specified log class are recorded.

`LS_DEBUG_BLC` sets the log class and is used in combination with `LS_LOG_MASK`, which sets the log level. For example:

```
LS_LOG_MASK=LOG_DEBUG LS_DEBUG_BLC="LC_TRACE"
```

To specify multiple log classes, use a space-separated list enclosed in quotation marks. For example:

```
LS_DEBUG_BLC="LC_TRACE"
```

You need to restart the `blcollect` daemons after setting `LS_DEBUG_BLC` for your changes to take effect.

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_LICENCE` is also supported for backward compatibility)
- `LC_PERFM` and `LC2_PERFM`: Log performance messages
- `LC_PREEMPT`: Log license preemption policy messages
- `LC_RESREQ` and `LC2_RESREQ`: Log resource requirement messages
- `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.

**LS_DEBUG_BLD**

Syntax

```
LS_DEBUG_BLD=log_class
```

Description

Sets the debugging log class for the Platform License Scheduler `bld` daemon.

Used for both project mode and cluster mode.

Specifies the log class filtering to be applied to `bld`. Messages belonging to the specified log class are recorded. Not all debug messages 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 `bladmin 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_LICENCE` is also supported for backward compatibility)
- `LC_MEMORY`: Log memory use messages
- `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`.

Used for both project mode and cluster mode.

**Default**

`N`

**LS_LOG_MASK**

**Syntax**

```
LS_LOG_MASK=message_log_level
```

446 Platform LSF Configuration Reference
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.

Used for both project mode and cluster mode.

For example:

LS_LOG_MASK=LOG_DEBUG

The log levels in order from highest to lowest are:

- LOG_ERR
- LOG_WARNING
- LOG_INFO
- LOG_DEBUG
- LOG_DEBUG1
- LOG_DEBUG2
- LOG_DEBUG3

The most important Platform 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.

Platform 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.

Used for both project mode and cluster mode.

Default

0 (old bld.stream file is not saved)
LS_MAX_STREAM_SIZE

Syntax

\[ \text{LS_MAX_STREAM_SIZE} = \text{integer} \]

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.

Used for both project mode and cluster mode.

Default

1024

LS_MAX_TASKMAN_PREEMPT

Syntax

\[ \text{LS_MAX_TASKMAN_PREEMPT} = \text{integer} \]

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.

Used for both project mode and cluster mode.

Default

unlimited

LS_MAX_TASKMAN_SESSIONS

Syntax

\[ \text{LS_MAX_TASKMAN_SESSIONS} = \text{integer} \]

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 Platform License Scheduler.

The number taskman sessions must be a positive integer.

The actual maximum number of taskman jobs is affected by the operating system file descriptor limit.

Make sure the operating system file descriptor limit and the maximum concurrent connections are large enough to support all taskman tasks, License Scheduler (bl*) commands, and connections between License Scheduler and LSF.

Used for both project mode and cluster mode.
LS_STREAM_FILE

Syntax

\[ \text{LS_STREAM_FILE} = \text{path} \]

Used for both project mode and cluster mode.

Description

Defines the full path and filename of the bld event log file, bld.stream by default.

\[ \textbf{Note:} \]

In Platform License Scheduler 8.0 the bld.events log file was replaced by the bld.stream log file.

Default

\[ \text{LSF_ROOT/work/db/bld.stream} \]

LS_PREEMPT_PEER

Syntax

\[ \text{LS_PREEMPT_PEER} = \text{Y} \]

Description

Enables bottom-up license token preemption in hierarchical project group configuration. Platform License Scheduler attempts to preempt tokens from the closest projects in the hierarchy first. This balances token ownership from the bottom up.

Used for project mode only.

Default

Not defined. Token preemption in hierarchical project groups is top down.

MBD_HEARTBEAT_INTERVAL

Syntax

\[ \text{MBD_HEARTBEAT_INTERVAL} = \text{seconds} \]

Description

Sets the length of time the cluster license allocation remains unchanged after a cluster has disconnected from bld. After MBD_HEARTBEAT_INTERVAL has passed, the allocation is set to zero and licenses are redistributed to other clusters.

Used for cluster mode only.

Default

900 seconds
**MBD_REFRESH_INTERVAL**

**Syntax**

```plaintext
MBD_REFRESH_INTERVAL=seconds
```

**Description**

`MBD_REFRESH_INTERVAL`: Cluster mode and project mode. This parameter allows the administrator to independently control the minimum interval between load updates from `bld`, and the minimum interval between load updates from LIM. The parameter controls the frequency of scheduling interactive (taskman) jobs. The parameter is read by `mbatchd` on startup. When `MBD_REFRESH_INTERVAL` is set or changed, you must restart `bld`, and restart `mbatchd` in each cluster.

Used for both project mode and cluster mode.

**Default**

15 seconds

**AUTH**

**Syntax**

```plaintext
MERGE_BY_SERVICE_DOMAIN=Y
```

**Description**

Correlates job license checkout with the `lmstat` output across all service domains first before reserving licenses.

This parameter supports the case where application's checkout license number is less than or equal to the job's rusage. If the checked out licenses are greater than the job's rusage, the `ENABLE_DYNAMIC_RUSAGE` parameter is still required.

**Default**

Not defined.

**PEAK_INUSE_PERIOD**

**Syntax**

```plaintext
PEAK_INUSE_PERIOD=seconds
```

**Description**

Defines the interval over which a peak INUSE value is determined for dynamic license allocation in cluster mode for all license features over all service domains.

Used for cluster mode only.

When defined in both the `Parameters` section and the `Features` section, the `Features` section definition is used for that license feature.
Default

300 seconds

PORT
Syntax

PORT=integer

Description

Defines the TCP listening port used by Platform License Scheduler hosts, including candidate Platform License Scheduler hosts. Specify any non-privileged port number.
Used for both project mode and cluster mode.

PREEMPT_ACTION
Syntax

PREEMPT_ACTION=action

Description

Specifies the action used for taskman job preemption.
By default, if PREEMPT_ACTION is not configured, bld sends a TSTP signal to preempt taskman jobs.
You can specify a script using this parameter. For example, PREEMPT_ACTION = /home/user1/preempt.s issues preempt.s when preempting a taskman job.
Used for project mode only.

Default

Not defined. A TSTP signal is used to preempt taskman jobs.

STANDBY_CONNTIMEOUT
Syntax

STANDBY_CONNTIMEOUT=seconds

Description

Sets the connection timeout the standby bld waits when trying to contact each host before assuming the host is unavailable.
Used for both project mode and cluster mode.

Default

5 seconds
BLC_HEARTBEAT_FACTOR

Syntax

\[ \text{BLC\_HEARTBEAT\_FACTOR} = \text{integer} \]

Description

Enables \texttt{bld} to detect \texttt{blcollect} failure. Defines the number of times that \texttt{bld} receives no response from a license collector daemon (\texttt{blcollect}) before \texttt{bld} resets the values for that collector to zero. Each license usage reported to \texttt{bld} by the collector is treated as a heartbeat.

Used for both project mode and cluster mode.

Default

3

Clusters section

Description

Required. Lists the clusters that can use Platform License Scheduler.

When configuring clusters for a WAN, the Clusters section of the master cluster must define its slave clusters.

The Clusters section is the same for both project mode and cluster mode.

Clusters section structure

The Clusters section begins and ends with the lines \texttt{Begin Clusters} and \texttt{End Clusters}. The second line is the column heading, \texttt{CLUSTERS}. Subsequent lines list participating clusters, one name per line:

\begin{verbatim}
Begin Clusters
CLUSTERS
cluster1
cluster2
End Clusters
\end{verbatim}

CLUSTERS

Defines the name of each participating LSF cluster. Specify using one name per line.

ServiceDomain section

Description

Required. Defines Platform License Scheduler service domains as groups of physical license server hosts that serve a specific network.

The ServiceDomain section is the same for both project mode and cluster mode.

ServiceDomain section structure

Define a section for each Platform 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**
- **LM_STAT_TIMEOUT**

#### NAME

Defines the name of the service domain.

Used for both project mode and cluster mode.

#### LIC_SERVERS

**Syntax**

```
LIC_SERVERS=(
    (host_name
        port_number@host_name
    )
    ...
)
```

**Description**

Defines the FlexNet license server hosts that make up the Platform License Scheduler service domain. For each FlexNet license server host, specify the number of the port that FlexNet uses, then the @ 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.

Used for both project mode and cluster mode.

**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 (`blcollect`) to use in each service domain. `blcollect` collects license usage information from FlexNet and passes it to the Platform License Scheduler daemon (`bld`). 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 `blcollect`, you must specify the name of the collector for the service domain. You can use any name you want.

Used for both project mode and cluster mode.

**Default**

Undefined. The Platform 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 Platform 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.

Used for both project mode and cluster mode.

**Default**

Platform License Scheduler applies the global value defined in the Parameters section.

**LM_STAT_TIMEOUT**

**Syntax**

```
LM_STAT_TIMEOUT=seconds
```

**Description**

Sets the timeout value passed to the `lmstat` command. The Parameters section setting is overwritten by the ServiceDomain setting, which is overwritten by the command line setting (`blcollect -t timeout`).

Used for both project mode and cluster mode.

**Default**

180 seconds
Feature section

Description

Required. Defines license distribution policies.

Feature section structure

Define a section for each feature managed by Platform License Scheduler.

```
Begin Feature
NAME=vcs
FLEX_NAME=vcs
...
Distribution policy
Parameters
...
End Feature
```

Parameters

- NAME
- CLUSTER_MODE
- FLEX_NAME
- DISTRIBUTION
- ALLOCATION
- GROUP
- GROUP_DISTRIBUTION
- CLUSTER_DISTRIBUTION
- INUSE_FROM_RUSAGE
- ALLOC_BUFFER
- LOCAL_TO
- LS_ACTIVE_PERCENTAGE
- LS_FEATURE_PERCENTAGE
- NON_SHARED_DISTRIBUTION
- PEAK_INUSE_PERIOD
- PREEMPT_ORDER
- PREEMPT_RESERVE
- RETENTION_FACTOR
- 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 Platform License Scheduler and LSF to identify the license feature.

Used for both project mode and cluster mode.

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.

CLUSTER_MODE

Syntax

CLUSTER_MODE=Y

Description

Enables cluster mode (instead of project mode) for the license feature. Setting in the Feature section overrides the global setting in the Parameters section.

Cluster mode emphasizes high utilization of license tokens above other considerations such as ownership. License ownership and sharing can still be configured, but within each cluster instead of across multiple clusters. Preemption of jobs (and licenses) also occurs within each cluster instead of across clusters.

Cluster mode was introduced in Platform License Scheduler 8.0. Before cluster mode was introduced, project mode was the only choice available.

Default

Undefined (N). Platform License Scheduler runs in project mode.

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 Platform License Scheduler token name is not identical to the FlexNet feature name.

Used for both project mode and cluster mode.

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

<table>
<thead>
<tr>
<th>Begin Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>FLEX_NAME=201-AppZ</td>
</tr>
<tr>
<td>NAME=App201</td>
</tr>
<tr>
<td>DISTRIBUTION=LanServer1(Lp1 1 Lp2 1)</td>
</tr>
<tr>
<td>End Feature</td>
</tr>
</tbody>
</table>
DISTRIBUTION

Syntax

\[
\text{DISTRIBUTION} = \left[ \text{service\_domain\_name} \left[ \left( \text{project\_name} \text{number\_shares} / \text{number\_licenses\_owned} \right) \right] \ldots \left[ \text{default} \right] \right] \ldots
\]

service\_domain\_name

Specify a Platform License Scheduler service domain (described in the ServiceDomain section) that distributes the licenses.

project\_name

Specify a Platform 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. When configured, preemption is enabled and owned licenses are reclaimed using preemption when there is unmet demand.

default

A reserved keyword that represents the default project if the job submission does not specify a project (bsub -Lp), or the specified project is not configured in the Projects section of lsf.licensescheduler. Jobs that belong to projects do not get a share of the tokens when the project is not explicitly defined in DISTRIBUTION.

Description

Used for project mode only.

One of DISTRIBUTION or GROUP_DISTRIBUTION must be defined when using project mode. GROUP_DISTRIBUTION and DISTRIBUTION are mutually exclusive. If defined in the same feature, the Platform License Scheduler daemon returns an error and ignores this feature.

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.)
Examples

DISTRIBUTION=wanserver (Lp1 1 Lp2 1 Lp3 1 Lp4 1)

In this example, the service domain named wanserver shares licenses equally among four 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 or interactive tasks that licenses are to be allocated to.

project_name

Specify a Platform License Scheduler project (described in the Projects section or as default) 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 interactive tasks.

Used for project mode only.

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 tasks in Platform License Scheduler projects, use the LSF Task Manager, taskman. The Task Manager utility is supported by Platform License Scheduler.

Default

If ENABLE_INTERACTIVE is not set, each cluster receives equal share, and interactive tasks receive no shares.

Examples

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=ApplicationX
DISTRIBUTION=LicenseServer1 (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=LicenseServer1 (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=LicenseServer1 (Lp2 1)
ALLOCATIONS=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)
ALLOCATIONS=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. This is different from a ProjectGroup section; groups of projects are not hierarchical.

project_name
Specify a Platform License Scheduler project (described in the \texttt{Projects} section) that is allowed to use the licenses. The project must appear in the \texttt{DISTRIBUTION} and 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. The number of licenses owned by the group is the total number of licenses owned by member projects.

Used for project mode only.

This parameter is ignored if \texttt{GROUP\_distribution} is also defined.

**Example**

For example, without the \texttt{GROUP} configuration shown, \texttt{proj1} owns 4 license tokens and can reclaim them using preemption. After adding the \texttt{GROUP} configuration, \texttt{proj1} and \texttt{proj2} together own 8 license tokens. If \texttt{proj2} is idle, \texttt{proj1} is able to reclaim all 8 license tokens using preemption.

```plaintext
Begin Feature
NAME = AppY
DISTRIBUTION = LanServer1(proj1 1/4 proj2 1/4 proj3 2)
GROUP = GroupA(proj1 proj2)
End Feature
```

**GROUP\_distribution**

**Syntax**

```plaintext
GROUP\_distribution=top\_level\_hierarchy\_name

\texttt{top\_level\_hierarchy\_name}

Specify the name of the top level hierarchical group.
```

**Description**

Defines the name of the hierarchical group containing the distribution policy attached to this feature, where the hierarchical distribution policy is defined in a \texttt{ProjectGroup} section.

One of \texttt{DISTRIBUTION} or \texttt{GROUP\_distribution} must be defined when using project mode. \texttt{GROUP\_distribution} and \texttt{DISTRIBUTION} are mutually exclusive. If defined in the same feature, the Platform License Scheduler daemon returns an error and ignores this feature.

If \texttt{GROUP} is also defined, it is ignored in favor of \texttt{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.

```plaintext
Begin Feature
NAME = myjob2
GROUP_DISTRIBUTION = groups
SERVICE_DOMAINS = LanServer wanServer
End Feature
```

CLUSTER_DISTRIBUTION

Syntax

```plaintext
CLUSTER_DISTRIBUTION=service_domain(cluster shares/min/max ... )...
```

- **service_domain**
  
  Specify a Platform License Scheduler WAN service domain (described in the ServiceDomain section) that distributes licenses to multiple clusters, and the share for each cluster.

  Specify a Platform License Scheduler LAN service domain for a single cluster.

- **cluster**
  
  Specify each LSF cluster that accesses licenses from this service domain.

- **shares**
  
  For each cluster specified for a WAN service domain, specify a positive integer representing the number of shares assigned to the cluster. (Not required for a LAN service domain.)

  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 assigned by the service domain. The total number of shares is the sum of the shares assigned to each cluster.

- **min**
  
  Optionally, specify a positive integer representing the minimum number of license tokens allocated to the cluster when dynamic allocation is enabled for a WAN service domain (when ALLOC_BUFFER is defined for the feature).

  The minimum allocation is allocated exclusively to the cluster, and is similar to the non-shared allocation in project mode.

  Cluster shares take precedence over minimum allocations configured. If the minimum allocation exceeds the cluster's share of the total tokens, a cluster's allocation as given by `bld` may be less than the configured minimum allocation.

- **max**
  
  The CLUSTER_DISTRIBUTION directive is used to distribute tokens to clusters in a WAN service domain or to single clusters in a LAN service domain.
Optionally, specify a positive integer representing the maximum number of license tokens allocated to the cluster when dynamic allocation is enabled for a WAN service domain (when ALLOC_BUFFER is defined for the feature).

### Description

**CLUSTER_DISTRIBUTION** must be defined when using cluster mode.

Defines the cross-cluster 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 cluster name followed by its share assignment. The share assignment determines what fraction of available licenses is assigned to each cluster, in the event of competition between clusters.

### Examples

```bash
CLUSTER_DISTRIBUTION=wanserver(C1 1 C2 1 C3 1 C4 1)
CLUSTER_DISTRIBUTION = SD(C1 1 C2 1) SD1(C3 1 C4 1) SD2(C1 1) SD3(C2 1)
```

In these examples, *wanserver*, *SD*, and *SD1* are WAN service domains, while *SD2* and *SD3* are LAN service domains serving a single cluster.

### INUSE_FROM_RUSAGE

**Syntax**

```
INUSE_FROM_RUSAGE=Y|N
```

**Description**

When not defined or set to *N*, the INUSE value uses *rusage* from *bsub* job submissions merged with license checkout data reported by *blcollect* (as reported by *blstat*).

When **INUSE_FROM_RUSAGE=Y**, the INUSE value uses the *rusage* from *bsub* job submissions instead of waiting for the *blcollect* update. This can result in faster reallocation of tokens when using dynamic allocation (when ALLOC_BUFFER is set).

When for individual license features, the **Feature** section setting overrides the global **Parameters** section setting.

Used for cluster mode only.

**Default**

*N*

### ALLOC_BUFFER

**Syntax**

```
ALLOC_BUFFER = buffer | cluster_name buffer ... default buffer
```

**Description**

Enables dynamic distribution of licenses across clusters in cluster mode.
Cluster names must be the names of clusters defined in the Clusters section of `lsf.licensescheduler`.

Used for cluster mode only.

**ALLOC_BUFFER=buffer** sets one buffer size for all clusters, while **ALLOC_BUFFER=cluster_name buffer** ... sets a different buffer size for each cluster.

The buffer size is used during dynamic redistribution of licenses. Increases in allocation are determined by the **PEAK** value, and increased by **DEMAND** for license tokens to a maximum increase of **BUFFER**, the buffer size configured by **ALLOC_BUFFER**. The licenses allocation can increase in steps as large as the buffer size, but no larger.

Allocation buffers help determine the maximum rate at which tokens can be transferred to a cluster as demand increases in the cluster. The maximum rate of transfer to a cluster is given by the allocation buffer divided by **MBD REFRESH INTERVAL**. Be careful not to set the allocation buffer too large so that licenses are not wasted because they are be allocated to a cluster that cannot use them.

Decreases in license allocation can be larger than the buffer size, but the allocation must remain at **PEAK +BUFFER** licenses. The license allocation includes up to the buffer size of extra licenses, in case demand increases.

Increasing the buffer size allows the license allocation to grow faster, but also increases the number of licenses that may go unused at any given time. The buffer value must be tuned for each license feature and cluster to balance these two objectives.

Detailed license distribution information is shown in the `blstat` output.

Use the keyword **default** to apply a buffer size to all remaining clusters. For example:

```
Begin Feature
NAME = f1
CLUSTER_DISTRIBUTION = WanServers(banff 1 berlin 1 boston 1)
ALLOC_BUFFER = banff 10 default 5
End Feature
```

In this example, dynamic distribution is enabled. The cluster **banff** has a buffer size of 10, and all remaining clusters have a buffer size of 5.

To allow a cluster to be able to use licenses only when another cluster does not need them, you can set the cluster distribution for the cluster to 0, and specify an allocation buffer for the number of tokens that the cluster can request.

For example:

```
Begin Feature
CLUSTER_DISTRIBUTION=Wan(CL1 0 CL2 1)
ALLOC_BUFFER=5
End Feature
```

When no jobs are running, the token allocation for **CL1** is 5. **CL1** can get more than 5 tokens if **CL2** does not require them.

**Default**

Not defined. Static distribution of licenses is used in cluster mode.
LOCAL_TO

Syntax

```
LOCAL_TO=cluster_name | location_name(cluster_name [cluster_name ...])
```

Description

Used for project mode only.

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, Platform 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 Platform 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.

Platform License Scheduler manages features with different localities as 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 when using project mode.

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`.  

Platform LSF Configuration Reference 465
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_ACTIVE_PERCENTAGE**

**Syntax**

\[ \text{LS_ACTIVE_PERCENTAGE}=\text{Y} \mid \text{N} \]

**Description**

Configures license ownership in percentages instead of absolute numbers and adjusts ownership for inactive projects. Sets \text{LS_FEATURE_PERCENTAGE}=\text{Y} automatically.

Settings \text{LS_ACTIVE_PERCENTAGE}=\text{Y} dynamically adjusts ownership based on project activity, setting ownership to zero for inactive projects and restoring the configured ownership setting when projects become active. If the total ownership for the license feature is greater than 100%, each ownership value is scaled appropriately for a total ownership of 100%.

Used for project mode only.

**Default**

\text{N} (Ownership values are not changed based on project activity.)

**LS_FEATURE_PERCENTAGE**

**Syntax**

\[ \text{LS_FEATURE_PERCENTAGE}=\text{Y} \mid \text{N} \]

**Description**

Configures license ownership in percentages instead of absolute numbers. When not combined with hierarchical projects, affects the owned values in \text{DI_ STRI_BUT_ION} and the \text{NON_SHA_RED_DISTRIBUTION} values only.

When using hierarchical projects, percentage is applied to \text{OWNERSHIP}, \text{LIMITS}, and \text{NON_SHARED} values. Used for project mode only.

For example:

```
Begin Feature
LS_FEATURE_PERCENTAGE = Y
DI_ STRI_BUT_ION = LanServer (p1 1 p2 1 p3 1/20)
...
End Feature
```

The service domain LanServer shares licenses equally among three LicenseScheduler 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 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 Platform 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**

<table>
<thead>
<tr>
<th>GROUP</th>
<th>SHARES</th>
<th>OWNERSHIP</th>
<th>LIMITS</th>
<th>NON_SHARED</th>
</tr>
</thead>
<tbody>
<tr>
<td>(R (A p4))</td>
<td>1 1</td>
<td>()</td>
<td>()</td>
<td>()</td>
</tr>
<tr>
<td>(A (B p3))</td>
<td>1 1</td>
<td>(-10)</td>
<td>(-20)</td>
<td>()</td>
</tr>
<tr>
<td>(B (p1 p2))</td>
<td>1 1</td>
<td>(30-)</td>
<td>()</td>
<td>(-5)</td>
</tr>
</tbody>
</table>

**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 Platform License Scheduler service domain (described in the ServiceDomain section) that distributes the licenses.

- **project_name**
  
  Specify a Platform License Scheduler project (described in the 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 privately owned, and are not shared with other license projects. They are available only to one project.
Used for project mode only.

Use blinfo -a to display NON_SHARED_DISTRIBUTION information.

For projects defined with NON_SHARED_DISTRIBUTION, you must assign the project OWNERSHIP an equal or greater number of tokens than the number of non-shared licenses. If the number of owned licenses is less than the number of non-shared licenses, OWNERSHIP is set to the number of non-shared licenses.

Example

```plaintext
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.

PEAK_INUSE_PERIOD

Syntax

```plaintext
PEAK_INUSE_PERIOD=seconds | cluster seconds ...
```

Description

Defines the interval over which a peak INUSE value is determined for dynamic license allocation in cluster mode for this license features and service domain.

Use keyword default to set for all clusters not specified, and the keyword interactive (in place of cluster name) to set for taskman jobs. For example:

```plaintext
PEAK_INUSE_PERIOD = cluster1 1000 cluster2 700 default 300
```

Used for cluster mode only.

When defined in both the Parameters section and the Features section, the Features section definition is used for that license feature.

Default

300 seconds
PREEMPT_ORDER
Syntax
PREEMPT_ORDER=BY_OWNERSHIP
Description
Optional. Sets the preemption order based on configured OWNERSHIP. Used for project mode only.
Default
Not defined.

PREEMPT_RESERVE
Syntax
PREEMPT_RESERVE=Y
Description
Optional. Enables Platform 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. Used for project mode only.
Default
Y. Reserved licenses are preemptable.

RETENTION_FACTOR
Syntax
RETENTION_FACTOR=integer%
Description
Ensures that when tokens are reclaimed from an overfed cluster, the overfed cluster still gets to dispatch additional jobs, but at a reduced rate. Specify the retention factor as a percentage of tokens to be retained by the overfed cluster.

For example:

```
Begin Feature
NAME = f1
CLUSTER_MODE = Y
CLUSTER_DISTRIBUTION = LanServer(LAN1 1 LAN2 1)
ALLOC_BUFFER = 20
RETENTION_FACTOR = 25%
End Feature
```

With RETENTION_FACTOR set, as jobs finish in the overfed cluster and free up tokens, at least 25% of the tokens can be reused by the cluster to dispatch additional jobs. Tokens not held by the cluster are redistributed to other clusters. In general, a higher value means that the process of reclaiming tokens from an overfed cluster takes longer, and an overfed cluster gets to dispatch more jobs while tokens are being reclaimed from it.
Used for cluster mode only.

**Default**
Not defined

**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.

Only a single service domain can be specified when using cluster mode.

**WORKLOAD_DISTRIBUTION**

**Syntax**

```
WORKLOAD_DISTRIBUTION=[service_domain_name(LSF lsf_distribution NON_LSF non_lsf_distribution)] ...
```

**service_domain_name**
Specify a Platform 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.

Used for both project mode and cluster mode. When running in cluster mode, `WORKLOAD_DISTRIBUTION` can only be specified for WAN service domains; if defined for a LAN feature, it is ignored.

Use `blinfo -a` to display `WORKLOAD_DISTRIBUTION` configuration.
Example

Begin Feature
NAME=ApplicationX
DISTRIBUTION=LicenseServer1(Lp1 1 Lp2 2)
WORKLOAD_DISTRIBUTION=LicenseServer1(LSF 8 NON_LSF 2)
End Feature

On the LicenseServer1 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 LicenseServer1 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).

Used for project mode only.

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 Platform License Scheduler to treat the license as a dynamic resource and prevents Platform License Scheduler from scheduling tokens for the feature when they are not available, or reserving license tokens when they should actually be free.

Used for project mode only. Cluster mode does not support rusage duration.

LM_REMOVE_INTERVAL

Syntax

LM_REMOVE_INTERVAL=seconds

Description

Specifies the minimum time a job must have a license checked out before lrmrmove can remove the license. lrmrmove causes lmgrd and vendor daemons to close the TCP connection with the application. They then retry the license checkout.
Used for both project mode and cluster mode.
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: Platform 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, Platform License Scheduler preempts one job that uses 10 or more licenses rather
than 10 jobs that each use one license.

Used for project mode only

Default
Undefined: Platform License Scheduler does not optimize the job list when selecting jobs to preempt.

ACCINUSE_INCLUDES_OWNERSHIP
Syntax

ACCINUSE_INCLUDES_OWNERSHIP=Y

Description
When not defined, accumulated use is incremented each scheduling cycle by (tokens in use) + (tokens
reserved) if this exceeds the number of tokens owned.

When defined, accumulated use is incremented each scheduling cycle by (tokens in use) + (tokens
reserved) regardless of 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.

Accumulated use is displayed by the `blstat` command under the heading `ACUM_USE`.

Used for project mode only. Cluster mode does not track accumulated use.

Default
N, not enabled.

LS_WAIT_TO_PREEMPT
Syntax

LS_WAIT_TO_PREEMPT=seconds
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.

Used for project mode only.

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`.

The FeatureGroup section is supported in both project mode and cluster mode.

**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.

Used for project mode only. When running in cluster mode, any ProjectGroup sections are ignored.

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 Platform License Scheduler.

The keywords GROUP, SHARES, OWNERSHIP, LIMIT, and NON_SHARED are required. The keywords PRIORITY and DESCRIPTION are optional. Empty brackets are allowed only for OWNERSHIP, LIMIT, 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
```

If desired, ProjectGroup sections can be completely independent, without any overlapping projects.

```
Begin ProjectGroup
GROUP               SHARES  OWNERSHIP LIMITS  NON_SHARED
(digital_sim (sim sim_reg)) (40 60)  (100 0) ()  ()
End ProjectGroup
```

```
Begin ProjectGroup
GROUP               SHARES  OWNERSHIP LIMITS  NON_SHARED
(analog_sim (app1 multitoken app1_reg)) (50 10 40) (65 25 0) (- 50 -)
End ProjectGroup
```

Parameters

- GROUP
- SHARES
- OWNERSHIP
- LIMITS
- NON_SHARED
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 INFINITY, 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:

<table>
<thead>
<tr>
<th>GROUP</th>
<th>SHARES</th>
<th>OWNERSHIP</th>
<th>LIMITS</th>
<th>NON_SHARED</th>
</tr>
</thead>
<tbody>
<tr>
<td>(final (G2 G1))</td>
<td>(1 1)</td>
<td>()</td>
<td>()</td>
<td>(2 0)</td>
</tr>
<tr>
<td>(G1 (AP2 AP1))</td>
<td>(1 1)</td>
<td>()</td>
<td>()</td>
<td>(1 1)</td>
</tr>
</tbody>
</table>

**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:

<table>
<thead>
<tr>
<th>GROUP</th>
<th>SHARES OWNERSHIP</th>
<th>LIMITS</th>
<th>NON_SHARED</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>(R (A B))</td>
<td>(1 1)</td>
<td>()</td>
<td>(10 10)</td>
<td>()</td>
</tr>
<tr>
<td>(A (p1 p2))</td>
<td>(1 1)</td>
<td>(40 60)</td>
<td>()</td>
<td>(&quot;p1 desc.&quot; &quot;&quot;)</td>
</tr>
<tr>
<td>(B (p3 p4))</td>
<td>(1 1)</td>
<td>()</td>
<td>()</td>
<td>(&quot;p3 desc.&quot; &quot;p4 desc.&quot;)</td>
</tr>
</tbody>
</table>

Use `blinfo -G` to view hierarchical project group descriptions.

Projects section

Description

Required for project mode only. Ignored in cluster mode. Lists the Platform 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 PRIORITY
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 in-use usage of each project, the projects are preempted according to the specified priority from lowest to highest.

Used for project mode only.

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 blinfo -Lp to view the project description.

Automatic time-based configuration

Variable configuration is used to automatically change Platform 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 bladmin ckconfig command, and restart Platform License Scheduler in the cluster with the bladmin reconfig command.

Used for both project mode and cluster mode.

The expressions are evaluated by Platform License Scheduler every 10 minutes based on the bld start time. When an expression evaluates true, Platform License Scheduler dynamically changes the configuration based on the associated configuration statements, restarting bld automatically.

When Platform LSF determines a feature has been added, removed, or changed, mbatchd no longer restarts automatically. Instead a message indicates that a change has been detected, prompting the user to restart manually with badmin mbdr restart.

This affects automatic time-based configuration in the Feature section of lsf.licensescheduler. When mbatchd detects a change in the Feature configuration, you must restart mbatchd for the change to take effect.
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
- `badmin mbdr restart` 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 TYPE NAME.

Subsequent lines name valid host types.

Example HostType section

```text
Begin HostType
TYPE NAME
SOL64
SOLSPARC
LINUX86 LINUXPPC
LINUX64
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 MODEL NAME, CPU FACTOR, and ARCHITECTURE.

Subsequent lines define a model and its CPU factor.
Example HostModel section

<table>
<thead>
<tr>
<th>MODELNAME</th>
<th>CPUFACTOR</th>
<th>ARCHITECTURE</th>
</tr>
</thead>
<tbody>
<tr>
<td>PC400</td>
<td>13.0</td>
<td>(i86pc_400 i866_400)</td>
</tr>
<tr>
<td>PC450</td>
<td>13.2</td>
<td>(i86pc_450 i866_450)</td>
</tr>
<tr>
<td>SparcSF</td>
<td>3.0</td>
<td>(SUNWSPARCstation5_170_sparc)</td>
</tr>
<tr>
<td>Sparc20</td>
<td>4.7</td>
<td>(SUNWSPARCstation20_151_sparc)</td>
</tr>
<tr>
<td>Ultra5S</td>
<td>10.3</td>
<td>(SUNWUltra5_270_sparcv9 SUNWUltra510_270_sparcv9)</td>
</tr>
</tbody>
</table>

**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:

<table>
<thead>
<tr>
<th>MODELNAME</th>
<th>CPUFACTOR</th>
<th>ARCHITECTURE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sparc5S</td>
<td>5.0</td>
<td>(SUNWUltra5_270_sparcv9)</td>
</tr>
<tr>
<td>PC486</td>
<td>2.0</td>
<td>(i486_33 i486_66)</td>
</tr>
<tr>
<td>PowerPC</td>
<td>3.0</td>
<td>(PowerPC12 PowerPC16 PowerPC31)</td>
</tr>
</tbody>
</table>
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

<table>
<thead>
<tr>
<th>RESOURCENAME</th>
<th>TYPE</th>
<th>INTERVAL</th>
<th>INCREASING</th>
<th>CONSUMABLE</th>
<th>DESCRIPTION</th>
<th># Keywords</th>
</tr>
</thead>
<tbody>
<tr>
<td>patchrev</td>
<td>Numeric</td>
<td>()</td>
<td>Y</td>
<td>()</td>
<td>(Patch revision)</td>
<td></td>
</tr>
<tr>
<td>specman</td>
<td>Numeric</td>
<td>()</td>
<td>N</td>
<td>()</td>
<td>(Specman)</td>
<td></td>
</tr>
<tr>
<td>switch</td>
<td>Numeric</td>
<td>()</td>
<td>Y</td>
<td>N</td>
<td>(Network Switch)</td>
<td></td>
</tr>
<tr>
<td>rack</td>
<td>String</td>
<td>()</td>
<td>()</td>
<td>()</td>
<td>(Server room rack)</td>
<td></td>
</tr>
<tr>
<td>owner</td>
<td>String</td>
<td>()</td>
<td>()</td>
<td>()</td>
<td>(Owner of the host)</td>
<td></td>
</tr>
<tr>
<td>elimres</td>
<td>Numeric</td>
<td>10</td>
<td>Y</td>
<td>()</td>
<td>(elim generated index)</td>
<td></td>
</tr>
<tr>
<td>ostype</td>
<td>String</td>
<td>()</td>
<td>()</td>
<td>()</td>
<td>(Operating system and version)</td>
<td></td>
</tr>
<tr>
<td>lmhostid</td>
<td>String</td>
<td>()</td>
<td>()</td>
<td>()</td>
<td>(FlexLM's lmhostid)</td>
<td></td>
</tr>
<tr>
<td>limversion</td>
<td>String</td>
<td>()</td>
<td>()</td>
<td>()</td>
<td>(Version of LIM binary)</td>
<td></td>
</tr>
</tbody>
</table>

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 indices:**
  - 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).
lsf.shared

- 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 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 badmin reconfig 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 %SYSTEM ROOT% 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 \texttt{lsf.sudoers} on Windows be \textbf{Administrators}. If not, `eauth` may not work.

The permissions on \texttt{lsf.sudoers} for Windows are:

\textbf{Workgroup Environment}
\begin{itemize}
  \item Local Admins (W )
  \item Everyone (R)
\end{itemize}

\textbf{Domain Environment}
\begin{itemize}
  \item Domain Admins (W )
  \item Everyone (R)
\end{itemize}

\textbf{File format}

The format of \texttt{lsf.sudoers} is very similar to that of \texttt{lsf.conf}.

Each entry can have one of the following forms:
\begin{itemize}
  \item \texttt{NAME=VALUE}
  \item \texttt{NAME=}
  \item \texttt{NAME= "STRING1 STRING2 ..."}
\end{itemize}

The equal sign \texttt{=} must follow each \texttt{NAME} even if no value follows and there should be no space beside the equal sign.

\texttt{NAME} describes an authorized operation.

\texttt{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.

\textbf{Example lsf.sudoers File}

\begin{verbatim}
LSB_PRE_POST_EXEC_USER=user100
LSF_STARTUP_PATH=/usr/share/lsf/etc
LSF_STARTUP_USERS="user1 user10 user55"
\end{verbatim}

\textbf{Creating and modifying lsf.sudoers}

You can create and modify \texttt{lsf.sudoers} with a text editor.

After you modify \texttt{lsf.sudoers}, you must run \texttt{badmin hrestart all} to restart all \texttt{sbatchd}s in the cluster with the updated configuration.

\textbf{Parameters}
\begin{itemize}
  \item \texttt{LSB_PRE_POST_EXEC_USER}
  \item \texttt{LSF_EAUTH_KEY}
  \item \texttt{LSF_EAUTH_USER}
  \item \texttt{LSF_EXEEXEC_USER}
  \item \texttt{LSF_EGO_ADMIN_PASSWD}
  \item \texttt{LSF_EGO_ADMIN_USER}
  \item \texttt{LSF_LOAD_PLUGINS}
\end{itemize}
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 exec 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.
**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 the account of the primary LSF administrator.

---

**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.
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_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, sbatchd, and mbatchd) are installed. LSF daemons are usually installed in the path specified by LSF_SERVERDIR defined in the cshrc.lsf, profile.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 admin 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 `lsadmin` and `badmin`, which have been installed as `setuid root`. `LSF_STARTUP_USERS` specifies a list of user accounts that can successfully run the commands `lsadmin` and `badmin` 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 `lsadmin` and `badmin` 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 `lsadmin` and `badmin` 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`
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 my job 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 Remote Tasks. 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 uncompressed and extracted 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_ADMINS="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/ls/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/ls"
```

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
- LSFADMINS
- 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 lsfinstall uncomment LSF_EGO_ENVDIR and sets LSF_ENABLE_EGO="Y" in lsf.conf.

ENABLE_EGO="N" causes lsfinstall 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_distrib"
```

Default

The parent directory of the current working directory. For example, if `lsfinstall` is running under `usr/share/lsf_distrib/lsf_lsfinstall` the LSF_TARDIR default value is `usr/share/lsf_distrib`.

**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

\texttt{LSF\_TOP=\"\texttt{path}\"}

Description

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

\textbf{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 (\texttt{/}).

Example

\texttt{LSF\_TOP=\"/usr/local/lsf\"}

Default

None—required variable
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 `bsub -e`.
- **LSB_JOBID**: Job ID assigned by LSF.
- **LSB_JOBINDEX**: Index of the job that belongs to a job array.
- **LSB_CHKPTNT_DIR**: This variable is set each time a checkpointed job is submitted. The value of the variable is `chkptnt_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_CHKPN_T_DIR
- LSB_CHKPN_T_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

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BSB_BLOCK</td>
<td>BSB_CHK_RESREQ</td>
</tr>
<tr>
<td>BSB_QUIET</td>
<td>BSB_QUIET2</td>
</tr>
<tr>
<td>BSB_STDERR</td>
<td>CLEARCASE_DRIVE  CLEARCASE_MOUNTDIR</td>
</tr>
<tr>
<td>CLEARCASE_ROOT</td>
<td>ELIM_ABORT_VALUE  LM_LICENSE_FILE</td>
</tr>
<tr>
<td>LS_EXEC_T</td>
<td>LS_JOBPID  LS_LICENSE_SERVER_feature</td>
</tr>
<tr>
<td>LS_SUBCWD</td>
<td>LS_CHKPNT_DIR</td>
</tr>
<tr>
<td>LSB_DEBUG</td>
<td>LSB_DEBUG_CMD</td>
</tr>
<tr>
<td>LSB_DEBUG_MBD</td>
<td>LSB_DEBUG_NQS  LSB_DEBUG_SBD</td>
</tr>
<tr>
<td>LSB_DEBUG_SCH</td>
<td>LSB_DEFAULT_JOBGROUP  LSB_DEFAULTPROJECT</td>
</tr>
<tr>
<td>LSB_DEFAULTQUEUE</td>
<td>LSB_DJOB_COMMFAIL_ACTION  LSB_DJOB_ENV_SCRIPT</td>
</tr>
<tr>
<td>LSB_DJOB_NUMPROC</td>
<td>LSB_ECHKPNT_METHOD  LSB_ECHKPNT_METHOD_DIR</td>
</tr>
<tr>
<td>LSB_ECHKPNT_KEEP_OUTPUT</td>
<td>LSB_ERESTART_USRCMD  LSB_EXEC_RUSAGE</td>
</tr>
<tr>
<td>LSB_EXECHOSTS</td>
<td>LSB_EXIT_IF_CWD_NOTEXIST  LSB_EXIT_PRE_ABORT</td>
</tr>
<tr>
<td>LSB_EXIT_REQUEUE</td>
<td>LSB_FRAMES  LSB_HOSTS</td>
</tr>
<tr>
<td>LSB_INTERACTIVE</td>
<td>LSB_JOB_INCLUDE_POSTPROC  LSB_JOBEXIT_INFO</td>
</tr>
<tr>
<td>LSB_JOBEXIT_STAT</td>
<td>LSB_JOBFILENAME  LSB_JOBGROUP</td>
</tr>
<tr>
<td>LSB_JOBID</td>
<td>LSB_JOBINDEX  LSB_JOBINDEX_STEP</td>
</tr>
<tr>
<td>LSB_JOBNAME</td>
<td>LSB_JOBPEND  LSB_JOBPGIDS</td>
</tr>
<tr>
<td>LSB_JOBPIDS</td>
<td>LSB_MAILSIZE  LSB_MCPU_HOSTS</td>
</tr>
<tr>
<td>LSB_NQS_PORT</td>
<td>LSB_NTRIES  LSB_OLD_JOBID</td>
</tr>
<tr>
<td>LSB_OUTPUT_TARGETFAILED</td>
<td>LSB_QUEUE  LSB_REMOTEINDEX</td>
</tr>
<tr>
<td>LSB_REMOTEJID</td>
<td>LSB_RESIZABLE  LSB_RESIZE_NOTIFY_OK</td>
</tr>
<tr>
<td>LSB_RESIZE_NOTIFY_FAIL</td>
<td>LSB_RESTART_PGID  LSB_RESTART</td>
</tr>
<tr>
<td>LSB_RESTART_PID</td>
<td>LSB_RTASK_GONE_ACTION</td>
</tr>
<tr>
<td>LSB_SUB_CLUSTER</td>
<td>LSB_SUB_COMMAND_LINE  LSB_SUB_EXTSCHED_PARAM</td>
</tr>
<tr>
<td>LSB_SUB_JOB_ACTION_WARNING_TIME</td>
<td>LSB_SUB_JOB_WARNING_ACTI O N</td>
</tr>
<tr>
<td>LSB_SUB_PARM_FILE</td>
<td>LSB_SUCCESS_EXIT_VALUES  LSB_SUSP_REASONS</td>
</tr>
<tr>
<td>LSB_SUSP_SUBREASONS</td>
<td>LSB_UNIXGROUP  LSB_USER_BIND_CPU_LIST</td>
</tr>
</tbody>
</table>
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 \texttt{bsub}, which is synchronous execution, then \texttt{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 \texttt{bsub}

BSUB_CHK_RESREQ

Syntax
\texttt{BSUB_CHK_RESREQ=any\_value}

Description
When \texttt{BSUB_CHK_RESREQ} is set, \texttt{bsub} checks the syntax of the resource requirement selection string without actually submitting the job for scheduling and dispatch. Use \texttt{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`. 
Environment variable reference

Default
Not defined

Where defined
From the command line

Example

BSUB_QIET2=1

BSUB_STDERR
g

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
g

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:

```plaintext
CLEARCASE_ROOT=/view/myview
```

In Windows:

```plaintext
CLEARCASE_ROOT=F:\myview
```

**See also**

CLEARCASE_DRIVE, CLEARCASE_MOUNTDIR, LSF_JOB_STARTER

**ELIM_ABORT_VALUE**

**Syntax**

```plaintext
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 $ELIM_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**

```plaintext
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 | CHKPNT | 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
- CHKPNT 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, `sbatchd` 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 `mbatchd` log file if `DEBUG1` and `LC_LICSCHED` 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 `sbatchd`

**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 badmin 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 badmin 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=engineering
```

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_EXECHOHOSTS

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 requeued or executed

Where defined

Set by sbatchd
See also
See PRE_EXEC in lsb.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 lsb.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 lsb.queues

Example

LSB_EXIT_REQUEUE="7 31"

See also
REQUEUE_EXIT_VALUES in lsb.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 sbatchd

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 sbatchd 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 usingbsub -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 sbatchd

**LSB_JOB_INCLUDE_POSTPROC**

Syntax

```plaintext
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.applications.

Default

Not defined

Where defined

From the command line

**LSB_JOBEXIT_INFO**

Syntax

```plaintext
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 lsbqueues.

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

```plaintext
LSB_JOBEXIT_INFO="SIGNAL -1 SIG_CHKPT" 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

\texttt{LSB\_JOBID=job\_ID}

Description
The job ID assigned by sbatchd. This is the ID of the job assigned by LSF, as shown by bjobs.

Valid values
Any positive integer

Where defined
Set by sbatchd, defined by mbatchd

See also
LSB_REMOTEJID

LSB_JOBINDEX
Syntax

\texttt{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
```

**Environment variable reference**

**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 `sbatchd`

**Example**

When you submit a job using the `-J` option of `bsub`, for example:

```
% bsub -J "myjob" job
```

`sbatchd` sets `LSB_JOBNAME` to the job name that you specified:

```
LSB_JOBNAME=myjob
```

**LSB_JOBPEND**

**Description**

Set if the job is requeued.

**Where defined**

Set by `sbatchd` 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 `sbatchd`.

**See also**

`LSB_JOBPIDS`
LSB_JOBPIDS
Description
A list of the current process IDs of the job.

Where defined
The process IDs are assigned by the operating system, and LSB_JOBPIDS is set by sbatchd.

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 sbatchd 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, `sbatchd` 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. `sbatchd` needs to spend a lot of time parsing the string. If the size of LSB_HOSTS exceeds 4096 bytes, LSB_HOSTS is ignored, and `sbatchd` 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 `sbatchd` before starting a job on the execution host

Example

When you submit a job with the -m and -n options of `bsub`, for example,

```
% bsub -m "hostA hostB" -n 6 job
```

`sbatchd` 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**

```bash
LSB_NTRIES=integer
```

**Description**

The number of times that LSF libraries attempt to contact `mbatchd` or perform a concurrent jobs query.

For example, if this parameter is not defined, when you type `bjobs`, LSF keeps displaying "batch system not responding" if `mbatchd` cannot be contacted or if the number of pending jobs exceeds `MAX_PEND_JOBS` specified in `lsb.params` or `lsb.users`.

If this parameter is set to a value, LSF only attempts to contact `mbatchd` the defined number of times and then quits. LSF will wait for a period of time equal to `SUB_TRY_INTERVAL` specified in `lsb.params` before attempting to contact `mbatchd` again.

**Valid values**

Any positive integer

**Default**

`INFINIT_INT` (The default is to continue the attempts to contact `mbatchd`)

---

**LSB_OLD_JOBID**

**Syntax**

```bash
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 `sbatch`, defined by `mbatchd`

**See also**

`LSB_JOBID`

---

**LSB_OUTPUT_TARGETFAILED**

**Syntax**

```bash
LSB_OUTPUT_TARGETFAILED=Y
```

---

Platform LSF Configuration Reference 533
Environment variable reference

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 `sbatchd` 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.applications` when running `bsub -app` for the specified application

See also

`DJOB_COMMFAIL_ACTION` in `lsb.applications`

**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 `setup` before launching a parallel or distributed job, and with argument `cleanup` 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 sbatchd, defined by mbatchd

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 bmig 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 sbatchd 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**

\[
\text{LSB\_RESTART\_PID} = \text{pid}
\]

**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**

\[
\text{LSB\_RTASK\_GONE\_ACTION} = \text{task\_action} \ldots
\]

**Description**

Defines the actions LSF should take if it detects that a remote task of a parallel job is gone. Where task_action is:

\begin{itemize}
  \item **IGNORE_TASKCRASH**
    A remote task crashes. The job RES does nothing.
  \item **KILLJOB_TASKDONE**
    A remote task exits with zero value. The job RES notifies the task RES to terminate all tasks in the job.
  \item **KILLJOB_TASKEXIT**
    A remote task exits with non-zero value. The job RES notifies the task RES to terminate all tasks in the job.
\end{itemize}
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 `lsrun` or `lsgrun` 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.applications`.

**LSB_SUSP_REASONS**

**Syntax**
```
LSB_SUSP_REASONS=integer
```

**Description**
An integer representing suspend reasons. Suspend reasons are defined in `lsbatch.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 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.

Load index values are defined in lsf.h.

<table>
<thead>
<tr>
<th>Load Index</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>R15S</td>
<td>0</td>
</tr>
<tr>
<td>R1M</td>
<td>1</td>
</tr>
<tr>
<td>R15M</td>
<td>2</td>
</tr>
<tr>
<td>UT</td>
<td>3</td>
</tr>
<tr>
<td>PG</td>
<td>4</td>
</tr>
<tr>
<td>IO</td>
<td>5</td>
</tr>
<tr>
<td>LS</td>
<td>6</td>
</tr>
<tr>
<td>IT</td>
<td>7</td>
</tr>
<tr>
<td>TMP</td>
<td>8</td>
</tr>
<tr>
<td>SWP</td>
<td>9</td>
</tr>
<tr>
<td>MEM</td>
<td>10</td>
</tr>
</tbody>
</table>

**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.applications. 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.applications. 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.
Environment variable reference

**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

\texttt{LSF\_EAUTH\_UID=}user\_ID

Description

Specifies the user account under which \texttt{eauth -s} runs. Set by the LSF daemon that executes \texttt{eauth}. Set by the LSF daemon that executes \texttt{eauth}.

LSF_EXECUTE_DOMAIN

Syntax

\texttt{LSF\_EXECUTE\_DOMAIN=}domain\_name

setenv \texttt{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 (\texttt{.cshrc} or \texttt{.profile}) or from the command line. Specify only one domain.

Use this parameter in conjunction with the \texttt{bsub}, \texttt{lsrun}, and \texttt{lsgrun} 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 \texttt{lsf.conf}.

See LSF\_INTERACTIVE\_STDERR in \texttt{lsf.conf} for more details.

LSF_INVOKE_CMD

Syntax

\texttt{LSF\_INVOKE\_CMD=}invoking\_command\_name

Description

Indicates the name of the last LSF command that invoked an external executable (for example, \texttt{esub} or \texttt{eexec}).

External executables get called by different LSF commands, such as \texttt{bsub}, \texttt{bmod}, or \texttt{lsrun}.

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 lsr, lsg, or ch.

If you define LSF_JOB_STARTER as follows:

```
setenv LSF_JOB_STARTER "/bin/csh -c"
```

and run a simple C-shell job:

```
ls run "'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`. 
Environment variable reference

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 (M ELIM).

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
**LSF_NIOS_DIE_CMD**

**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**

```plaintext
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**

```plaintext
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**

```bash
LSF_NIOS_PORT_RANGE=min_port_number-max_port_number
```

**Description**

Defines a range of listening ports for NIOS to use.

**Example**

```bash
LSF_NIOS_PORT_RANGE=5000-6000
```

**Default**

Not defined. LSF randomly assigns a NIOS port number.

**LSF_RESOURCES**

**Syntax**

```bash
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**

```bash
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

.A lsftask file 493

A

ABS_RUNLIMIT
  lsb.params file 41, 133
ACCESS_CONTROL
  lsb.serviceclasses file 274
ACCINUSE_INCLUDES_OWNERSHIP
  lsf.licensescheduler file Parameters section 473
ACCT_ARCHIVE_AGE
  lsb.params file 133
ACCT_ARCHIVE_SIZE
  lsb.params file 134
ACCT_ARCHIVE_TIME
  lsb.params file 134
ADJUST_DURATION
  lsf.cluster file 296
ADMIN
  lsb.hosts file 123
  lsf.licensescheduler file Parameters section 438
administrator
  user group 286
ADMINISTRATORS
  lsb.queues file 189
  lsf.cluster file 306
ADVRSV_USER_LIMIT
  lsb.params file 135
ALLOC_BUFFER
  lsf.licensescheduler file Features section 463
ALLOCATION
  lsf.licensescheduler file Feature section 458
APS_PRIORITY
  lsb.queues file 189
ARCHITECTURE
  lsf.shared file 483
AUTH
  lsf.licensescheduler file Parameters section 438
AUTO_ATTACH
  lsb.serviceclasses file 275

automatic time-based configuration
  lsb.hosts 125
  lsb.params 132
  lsb.queues 239
  lsb.resources 271
  lsb.users 291
  lsf.licensescheduler 479

B

BACKFILL
  lsb.queues file 190
bconf
  enable 401
BIND_JOB 42
BLC_HEARTBEAT_FACTOR
  lsf.licensescheduler file Parameters section 452
bld
  License Scheduler daemon 454
bld.license.acct file 7
bld.stream file 447–449
BSUB_BLOCK variable 512
BSUB_QUIET variable 512, 513
BSUB_QUIET2 variable 513
BSUB_STDERR variable 514

C

CACHE_INTERVAL
  lsf.cluster file 314
CHKPNT
  lsb.hosts file 112
  lsb.queues file 190
CHKPNT_DIR
  lsb.applications file 42
CHKPNT_INITPERIOD
  lsb.applications file 43
CHKPNT_METHOD
  lsb.applications file 44

Platform LSF Configuration Reference 551
CHKPNT_PERIOD
  lsb.applications file 43
chunk jobs
  CHKPNT parameter in lsb.queues 191
  MIG parameter in lsb.queues 55, 215
  rerunnable 62, 225
CHUNK_JOB_DURATION
  lsb.params file 135
CHUNK_JOB_SIZE
  lsb.applications file 44
  lsb.queues file 191
CLEAN_PERIOD
  lsb.params file 136
CLEARCASE_DRIVE variable 514
CLEARCASE_MOUNTDIR variable 515
CLEARCASE_ROOT variable 515
CLOSE_ON_DEMAND
  lsb.resources file GuaranteedResourcePool section 261
cluster mode
  CLUSTER_MODE parameter 439, 456
CLUSTER_DISTRIBUTION
  lsf.licensescheduler file Feature section 462
CLUSTER_MODE
  lsf.licensescheduler file Parameters section 439, 456
ClusterName
  lsf.shared file 481
CLUSTERNAME
  lsf.cluster file 313
CLUSTERS
  lsf.licensescheduler file Clusters section 452
Clusters section
  lsf.licensescheduler file
  description 452
COMMITTED_RUN_TIME_FACTOR
  lsb.params file 136
  lsb.queues file 192
CPU_TIME_FACTOR
  lsb.params file 138
  lsb.queues file 194
CPUFACTOR
  lsf.shared file 483
CPULIMIT
  lsb.applications file 45
  lsb.queues file 193
Cray checkpointing 112
cross-cluster account mapping 290
cross-cluster account mapping in MultiCluster 290
cross-cluster user account mapping in MultiCluster 290
CSA (IRIX Comprehensive System Accounting)
  configuring and using 382
cshrc.lsf file
  description 9
  setting the LSF environment 10
CUMULATIVE_RUSAGE
  LSF HPC extensions parameter 387
custom resources
  reserving 270

D

daemons
  automatic shut down 319
  security 427
DATALIMIT
  lsb.applications file 46
  lsb.queues file 194
dedicated resource. See exclusive resource 309
DEFAULT_APPLICATION
  lsb.params file 138
DEFAULT_EXTSCHED
  lsb.queues file 194
DEFAULT_HOST_SPEC
  lsb.params file 138
  lsb.queues file 195
DEFAULT_JOBGROUP
  lsb.params file 139
DEFAULT_PROJECT
  lsb.params file 140
DEFAULT_QUEUE
  lsb.params file 140

CONTROL_ACTION
DEFAULT_SLA_VELOCITY
  lsb.params file 140
DEFAULT_USER_GROUP
  lsb.params file 141
DESCRIPTION
  lsb.applications file 46
  lsb.queues file 195
  lsb.resources file GuaranteedResourcePool section 261
  lsb.serviceclasses file 277
  lsf.licensescheduler file Project section 479
  lsf.licensescheduler file ProjectGroup section 477
  lsf.shared file 486
DETECT_IDLE_JOB_AFTER
  lsb.params file 141
DIRECTION
  lsb.users file 290
DISABLE_UACCT_MAP
  lsb.params file 142
DISP_RES_USAGE_LIMITS
  LSF HPC extensions parameter 387
DISPATCH_ORDER
  lsb.queues file 195
DISPATCH_WINDOW
  lsb.hosts file 112
  lsb.queues file 196
DISTRIBUTION
  lsb.resources file GuaranteedResourcePool section 260
  lsb.resources file HostExport section 263
  lsb.resources file SharedResourceExport section 265
  lsf.licensescheduler file Feature section 457
DISTRIBUTION_POLICY_VIOLATION_ACTION
  lsf.licensescheduler file Parameters section 439
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 383
dual-stack hosts
  setting to IPv6 only 378
DURATION
  lsb.resources file GuaranteedResourcePool section 261
DYNAMIC
  lsf.licensescheduler file Feature section 472

E
EADMIN_TRIGGER_DURATION
  lsb.params file 142
EGO_BINDIR
  cshrc.lsf and profile.lsf files 14
EGO_CONF_RETRY_INT parameter in ego.conf 370
EGO_CONF_RETRY_MAX parameter in ego.conf 371
EGO_CONFDIR
  cshrc.lsf and profile.lsf files 15
EGO_DAEMON_CONTROL
  install.config file 22, 499
EGO_DEBUG_LIM parameter in ego.conf 376
EGO_DHCP_ENV parameter in ego.conf 377
EGO_DYNAMIC_HOST_TIMEOUT parameter in ego.conf
  380
EGO_ENABLE_AUTO_DAEMON_SHUTDOWN 319
EGO_ENABLE_DUALCORE parameter in ego.conf 383
EGO_ESLIM_TIMEOUT 319
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 366
EGO_LIM_DEBUG parameter in ego.conf 400
EGO_LIM_PORT parameter in ego.conf 401
EGO_LIM_RECVTIMEOUT parameter in ego.conf 366
EGO_LOCAL_CONFDIR
  cshrc.lsf and profile.lsf files 16
EGO_LOCAL_RESOURCES parameter in ego.conf 403
EGO_LOG_MASK parameter in ego.conf 375, 404
EGO_MASTER_LIST parameter in ego.conf 410
EGO_PIM_INFODIR parameter in ego.conf 417
EGO_PIM_SLEEPTIME parameter in ego.conf 418
EGO_PIM_SLEEPTIME_UPDATE parameter in ego.conf 418
EGO_RES_REQ
  lsb.serviceclasses file 277
EGO_RSH parameter in ego.conf 424
EGO_SERVERDIR
  cshrc.lsf and profile.lsf files 16
EGO_STATIC_LIM_TIMEOUT 319
EGO_STRIP_DOMAIN parameter in ego.conf 429
EGO_TIME_LIM parameter in ego.conf 429
EGO_TOP
  cshrc.lsf and profile.lsf files 17
ego.conf file
    corresponding lsf.conf parameters 317
    LSF parameter migration for upgrade 317
    managing error logs 375

ELIM_ABORT_VALUE
    lsf.cluster file 296
ELIM_ABORT_VALUE variable 516
ELIM_POLL_INTERVAL
    lsf.cluster file 296
ELIMARGS
    lsf.cluster file 297
email
    configuring on UNIX 342
ENABLE_DEFAULT_EGO_SLA
    lsb.params file 142
ENABLE_DYNAMIC_HOSTS
    install.config file 22
ENABLE_DYNAMIC_RUSAGE
    lsf.licensescheduler file Feature section 472
ENABLE_EGO
    install.config file 22, 500
ENABLE_EVENT_STREAM
    lsb.params file 143
ENABLE_HIST_RUN_TIME
    lsb.params file 144
    lsb.queues file 196
ENABLE_HOST_INTERSECTION
    lsb.params file 144
ENABLE_HPC_INST
    install.config file 23
ENABLE_INTERACTIVE
    lsf.licensescheduler file Parameters section 440
ENABLE_MINJOB_PREEMPTION
    lsf.licensescheduler file Feature section 473
ENABLE_USER_RESUME
    lsb.params file 144
ENFORCE_ONE_UG_LIMITS
    lsb.params file 145
ENFORCE_UG_TREE
    lsb.params file 145
environment variables
    BSBATCH_BLOCK 512
    BSBATCH_QUIET 512, 513
    BSBATCH_QUIET2 513
    BSBATCH_STDERR 514
    CLEARCASE_DRIVE 514
    CLEARCASE_MOUNTDIR 515
    CLEARCASE_ROOT 515
Platform LSF Configuration Reference 555
FLEX_NAME
  lsf.licensescheduler file Feature section 456
FLOAT_CLIENTS
  lsf.cluster file 297
FLOAT_CLIENTS_ADDR_RANGE
  lsf.cluster file 298

G
GLOBAL_EXIT_RATE
  lsb.params file 143, 148
GOALS
  lsb.serviceclasses file 278
GROUP
  lsf.licensescheduler file Feature section 460
  lsf.licensescheduler file ProjectGroup section 476
GROUP_ADMIN
  lsb.hosts file 117
GROUP_DISTRIBUTION
  lsf.licensescheduler file Feature section 461
GROUP_MEMBER
  lsb.hosts file 116
  lsb.users file 285
GROUP_NAME
  lsb.hosts file 115
  lsb.users file 285
GRP_ADD record
  lsb.events 105
GRP_MOD record
  lsb.events 106
GuaranteedResourcePool section
  lsb.resources 258

H
hierarchical fairshare user groups 287
HIST_HOURS
  lsb.params file 148
  lsb queues file 199
  lsf.licensescheduler file Parameters section 440, 441
HJOB_LIMIT
  lsb.queues file 200
host groups
  EGO enabled 201
host management
daemon clean up 319
host models
  automatic detection 483
host types
  automatic detection 483

HOST_CTRL record
  lsb.events 86
HOST_INACTIVITY_LIMIT
  lsf.cluster file 300
HOST_NAME
  lsb.hosts file 111
HOST_RUSAGE
  LSF HPC extensions parameter 387
HostExport section
  lsb.resources 262
HOSTNAME
  lsf.cluster file 308
hosts
  exclusive resource 309
HOSTS
  lsb.hosts file 120
  lsb.queues file 200
  lsb.resources file GuaranteedResourcePool section 260
  lsb.resources file Limit section 243
  lsb.resources file ResourceReservation section 267
  lsf.licensescheduler file Parameters section 441
hosts file 18
HPART_NAME
  lsb.hosts file 120

I
identd 367
identification daemon authentication
  LSF_AUTH parameter 367
identification daemons 367
IGNORE_DEADLINE
  lsb.queues file 203
IMPT_JOBBKLG
  lsb.queues file 203
INCREASING
  lsf.shared file 485
install.config file
description 21
INTERACTIVE
  lsb.queues file 204
INTERRUPTIBLE_BACKFILL
  lsb.queues file 204
INTERVAL
  lsf.shared file 485
INUSE_FROM_RUSAGE
  lsf.licensescheduler file Features section 463
  lsf.licensescheduler file Parameters section 441
io
  lsb.hosts file 114
  lsb.queues file 210
IPv6
  dual-stack hosts 378
  enable 384
  example 18, 20
  in FLOAT_CLIENTS_ADDR_RANGE 299
  in LSF_HOST_ADDR_RANGE 303
IRIX ULDB (User Limits Database)
  description 431
  jlimit.in file 431
it
  lsb.hosts file 114
  lsb.queues file 210
J
JL/P
  lsb.users file 289
JL/U
  lsb.hosts file 113
  jlimit.in file
  IRIX ULDB 431
job ID
  limit 162
  rollover 162
  sequencing 162
job migration
  absolute job priority scheduling 190
JOB_ACCEPT record
  lsb.events 81
JOB_ACCEPT_INTERVAL
  lsb.params file 149
  lsb.queues file 205
JOB_ACTION_WARNING_TIME
  lsb.queues file 205
JOB_ATTA_DATA record
  lsb.events 98
JOB_ATTA_DIR
  lsb.params file 150
JOB_CHUNK record
  lsb.events 99
JOB_CLEAN record
  lsb.events 96
JOB_CONTROLS
  lsb.queues file 206
JOB_DEP_LAST_SUB
JOB_EXECUTE record
  lsb.events 95
JOB_EXIT_RATE_DURATION
  lsb.params file 151
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 151
JOB_IDLE
  lsb.queues file 207
JOB_INCLUDE_POSTPROC parameter
  lsb.applications 49
  lsb.params 151
JOB_MODIFY2 record
  lsb.events 90
JOB_MOVE record
  lsb.events 85
JOB_NEW record
  lsb.events 75
JOB_OVERRUN
  lsb.queues file 208
JOB_POSITION_CONTROL_BY_ADMIN
  lsb.params file 152
JOB_POSTPROC_TIMEOUT parameter
  lsb.applications 49
  lsb.params 152
JOB_PRIORITY_OVER_TIME
  lsb.params file 153
JOB_REQUEUE record
  lsb.events 96
JOB_RUNLIMIT_RATIO
  lsb.params file 154
JOB_SCHEDULING_INTERVAL
  lsb.params file 155
JOB_SIGACT record
  lsb.events 88
JOB_SIGNAL record
  lsb.events 94
JOB_SPOOL_DIR
  description 155
JOB_START record
  lsb.events 81
JOB_START_ACCEPT record
  lsb.events 82
JOB_STARTER
  lsb.applications file 50
  lsb.queues file 208
JOB_STATUS record
  lsb.events 83
JOB_SWITCH record
  lsb.events 84
JOB_TERMINATE_INTERVAL
  lsb.params file 156
JOB_UNDERRUN
  lsb.queues file 209
JOB_WARNING_ACTION
  lsb.queues file 209
jobs
  allowing preemption of 219
  preempting by run time 178
JOBSS
  lsb.resources file Limit section 244

L

LIB_CONN_TIMEOUT
  lsf.licensescheduler file Parameters section 442
LIB_RECV_TIMEOUT
  lsf.licensescheduler file Parameters section 442
LIC_COLLECTOR
  lsf.licensescheduler file ServiceDomain section 453
LIC_SERVERS
  lsf.licensescheduler file ServiceDomain section 453
LICENSE
  lsb.resources file Limit section 245
  lim.acct file 30
LIMIT
  lsf.licensescheduler file ProjectGroup section 476
  limit number of hosts 345
Limit section
  lsb.resources 240
limits
  enforced with overlapping members 145
  job ID 162
live reconfiguration
directory 401
  LSF_LIVE_CONFDIR 401
LM_LICENSE_FILE variable 516
LM_REMOVE_INTERVAL
  lsf.licensescheduler file Features section 472
  lsf.licensescheduler file Parameters section 443
LM_STAT_INTERVAL
  lsf.licensescheduler file Parameters section 443
  lsf.licensescheduler file ServiceDomain section 454
LM_STAT_TIMEOUT
  lsf.licensescheduler file Parameters section 443
  lsf.licensescheduler file ServiceDomain section 454
LMSTAT_PATH
  lsf.licensescheduler file Parameters section 444
load_index
  lsb.hosts file 114
  lsb.queues file 209
LOAD_INDEX record
  lsb.events 88
LOAD_POLICIES
  lsb.resources file GuaranteedResourcePool section 261
LOCAL
  lsb.users file 290
  local tasks in task files 495
LOCAL_MAX_PREEXEC_RETRY
  lsb.applications file 51
  lsb.params file 157
  lsb.queues file 210
LOCAL_TO
  lsf.licensescheduler file Features section 465
LOCATION
  lsf.cluster file 312
log files
  nios.log.host_name 412
LOG_EVENT
  lsf.licensescheduler file Parameters section 444
ls
  lsb.hosts file 114
  lsb.queues file 210
LS_ACTIVE_PERCENTAGE
  lsf.licensescheduler file Features section 467
LS_ADMIN
  setup.config file 496
LS_DEBUG_BLC
  lsf.licensescheduler file Parameters section 444
LS_DEBUG_BLD
  lsf.licensescheduler file Parameters section 445
LS_ENABLE_MAX_PREEMPT
  lsf.licensescheduler file Parameters section 446
LS_EXEC_T
  variable 517
LS_FEATURE_PERCENTAGE
lsf.licensescheduler file Features section 467
LS_HOSTS
    setup.config file 497
LS_JOBPID variable 517
LS_LICENSE_FILE
    setup.config file 497
LS_LICENSE_SERVER_feature variable 517
LS_LMSTAT_PATH
    setup.config file 498
LS_LOG_MASK
    lsf.licensescheduler file Parameters section 446
LS_MAX_STREAM_FILE_NUMBER
    lsf.licensescheduler file Parameters section 447
LS_MAX_STREAM_SIZE
    lsf.licensescheduler file Parameters section 448
LS_MAX_TASKMAN_PREEMPTS
    lsf.licensescheduler file Parameters section 448
LS_MAX_TASKMAN_SESSIONS
    lsf.licensescheduler file Parameters section 448
LS_PREEMPT_PEER
    lsf.licensescheduler file Parameters section 449
LS_STREAM_FILE
    lsf.licensescheduler file Parameters section 449
LS_SUBCWD variable 518
LS_TOP
    setup.config file 498
LS_WAIT_TO_PREEMPT
    lsf.licensescheduler file 473
LSB_API_CONNTIMEOUT
    lsf.conf file 320
LSB_API_RECVTIMEOUT
    lsf.conf file 320
LSB_API_VERBOSE
    lsf.conf file 321
LSB_BJOBS_CONSISTENT_EXIT_CODE
    lsf.conf file 321
LSB_BLOCK_JOBINFO_TIMEOUT
    lsf.conf file 321
LSB_BPEEK_REMOTE_OUTPUT
    lsf.conf file 323
LSB_CHKPNT_DIR variable 518
LSB_CHUNK_RUSAGE
    lsf.conf file 324
LSB_CMD_LOG_MASK
    lsf.conf file 324
LSB_CMD_LOGDIR
    lsf.conf file 325
LSB_CONFDIR
LSB_CPUSER_BESTCPUS
    lsf.conf file 326
LSB_CRDIR
    lsf.conf file 326
LSB_DEBUG
    lsf.conf file 327
variable 518
LSB_DEBUG_CMD
    lsf.conf file 327
variable 519
LSB_DEBUG_MBD
    lsf.conf file 329
variable 519
LSB_DEBUG_NQS
    lsf.conf file 330
variable 519
LSB_DEBUG_SBD
    lsf.conf file 330
variable 519
LSB_DEBUG_SCH
    lsf.conf file 331
variable 519
LSB_DEFAULT_JOBGROUP variable 519
LSB_DEFAULTPROJECT variable 520
LSB_DEFAULTQUEUE variable 521
LSB_DISABLE_LIMLOCK_EXCL
    lsf.conf file 179, 332
LSB_DISABLE_RERUN_POST_EXEC
    lsf.conf file 332
LSB_DISPLAY_YEAR
    lsf.conf file 333
LSB_DJOB_COMMFAIL_ACTION variable 534
LSB_DJOB_ENV_SCRIPT variable 534
LSB_DJOB_NUMPROC variable 521
LSB_ECHKPNT_KEEP_OUTPUT
    lsf.conf file 333
LSB_ECHKPNT_KEEP_OUTPUT variable 522
LSB_ECHKPNT_METHOD
    lsf.conf file 333
LSB_ECHKPNT_METHOD_DIR
    lsf.conf file 333
LSB_ERESTART_USRCMD variable 522
LSB_ESUB_METHOD
    lsf.conf file 333
LSB_EXEC_RUSAGE variable 522
LSB_EXECHOSTS variable 523
LSB_EXIT_IF_CWD_NOTEXIST variable 523
LSB_EXIT_PRE_ABORT variable 523
LSB_EXIT_REQUEUE variable 524
LSB_FRAMES variable 524
LSB_HCLOSE_BY_RES
LSF HPC extensions parameter 387
LSB_HJOB_PER_SESSION
lsf.conf file 335
LSB_HOSTS variable 525
LSB_INDEX_BY_JOB
lsf.conf file 336
LSB_INTERACT_MSG_ENH
lsf.conf file 336
LSB_INTERACT_MSG_INTVAL
lsf.conf file 337
LSB_INTERACTIVE variable 525
LSB_JOB_CPULIMIT
lsf.conf file 337
LSB_JOB_INCLUDE_POSTPROC variable 526
LSB_JOB_MEMLIMIT
lsf.conf file 338
LSB_JOB_OUTPUT_LOGGING
lsf.conf file 340
LSB_JOB_EXIT_INFO variable 526
LSB_JOB_EXIT_STAT variable 527
LSB_JOBFILENAME variable 527
LSB_JOBGROUP variable 527
LSB_JOBID variable 528
LSB_JOBID_DISP_LENGTH
lsf.conf file 340
LSB_JOBINDEX variable 528
LSB_JOBINDEX_STEP variable 529
LSB_JOBNAME variable 529
LSB_JOBPEND variable 530
LSB_JOBPGIDS variable 530
LSB_JOBPIDS variable 531
LSB_KEEP_SYSDEF_RLIMIT
lsf.conf file 341
LSB_LIMLOCK_EXCLUSIVE parameter 332
LSB_LOCALDIR
lsf.conf file 341
LSB_MAIL_FROM_DOMAIN
lsf.conf file 342
LSB_MAILPROG
lsf.conf file 342
LSB_MAILSERVER
lsf.conf file 343
LSB_MAILSIZE variable 531
LSB_MAILSIZE_LIMIT
lsf.conf file 344
LSB_MAILTO
lsf.conf file 344
LSB_MAX_ASKED_HOSTS_NUMBER
lsb.params 345
LSB_MAX_JOB_DISPATCH_PER_SESSION
lsf.conf file 346
LSB_MAX_NQS_QUEUES
lsf.conf file 346
LSB_MAX_PACK_JOBS
lsf.conf file 347
LSB_MAX_PROBE_SBD
lsf.conf file 347
LSB_MAX_QS_DOWN_MSG
lsf.conf file 348
LSB_MAX_QS_BUSY_SHORT_MSG
lsf.conf file 348
LSB_MAX_QS_MAX_SIG_COUNT 349
LSB_MAX_QS_MAX_SIG_COUNT 349
LSB_MAX_QS_MAX_SIG_COUNT 349
LSB_MAX_QS_PORT
lsf.conf file 349, 401
LSB_MC_CHKPT_RERUN
lsf.conf file 350
LSB_MC_INITFAIL_MAIL
lsf.conf file 350
LSB_MC_INITFAIL_RETRY
lsf.conf file 350
LSB_MC_CHKPT_RERUN
lsf.conf file 351
LSB_MIG2PEND
lsf.conf file 351
LSB_MIXED_PATH_DELIMITER
lsf.conf 352
LSB_MOD_ALL_JOBS
lsf.conf file 353
LSB_NCPU_ENFORCE
lsf.conf file 353
LSB_NQS_PORT
lsf.conf file 354
LSB_NTRIES environment variable 163
LSB_NTRIES variable 532
LSB_NUM_NIOS_CALLBACK_THREADS
lsf.conf file 354
LSB_OLD_JOBID variable 533
LSB_OUTPUT_TARGETFAILED variable 533
LSB_PACK_MESUB
  lsf.conf file 354
LSB_PACK_SKIP_ERROR lsf.conf file 355
LSB_PRE_POST_EXEC USER
  lsf.sudoers file 489
LSB_PSET_BIND_DEFAULT
  lsf.conf file 355
LSB_QUERY_PORT
  lsf.conf file 356
LSB_QUERY_PORT parameter in lsf.conf 158
LSB_QUEUE variable 535
LSB_REMOTEINDEX variable 535
LSB_REMOTEJID variable 535
LSB_REQUEUE_TO_BOTTOM
  lsf.conf file 357
LSB_RESTART variable 536
LSB_RESTART_RPID variable 536
LSB_RESTART_RPID variable 537
LSB_R_LA_PORT
  lsf.conf file 357
LSB_R_LA_UPDATE
  lsf.conf file 357
LSB_R_LA_WORKDIR
  lsf.conf file 358
LSB_RTASK_GONE_ACTION variable 537
LSB_SACCT_ONE_UG
  lsf.conf 358
LSB_SBD_PORT
  lsf.conf file 358, 401
LSB_SET_TMPDIR
  lsf.conf file 359
LSB_SHAREDIR
  lsf.conf file 359
LSB_SHORT_HOSTLIST
  lsf.conf file 359
LSB_SIGSTOP
  lsf.conf file 360
LSB_SSH_XFORWARD_CMD
  lsf.conf file 360
LSB_STDOUT_DIRECT
  lsf.conf file 361
LSB_STOP_IGNORE_IT
  lsf.conf file 361
LSB_SUB_CLUSTER variable 538
LSB_SUB_COMMAND_LINE variable 538
LSB_SUB_COMMANDNAME
  lsf.conf file 361
LSB_SUB_EXTSCHED_PARAM variable 538
LSB_SUB_JOB_ACTION_WARNING_TIME variable 538
LSB_SUB_JOB_WARNING_ACTION variable 539
LSB_SUB_PARM_FILE variable 539
LSB_SUBK_SHOW_EXEC_HOST
  lsf.conf file 362
LSB_SUCCESS_EXIT_VALUES variable 539
LSB_SUSP_REASONS variable 539
LSB_SUSP_SUBREASONS variable 540
LSB_SYNC_HOSTSTAT_LIM
  lsb.params 157
LSB_TIME_CMD
  lsf.conf file 363
LSB_TIME_MB
  lsf.conf file 363
LSB_TIME_RESERVE_NUMJOBS
  lsf.conf file 363
LSB_TIME_SBD
  lsf.conf file 364
LSB_TIME_SCH
  lsf.conf file 363
LSB_UNIXGROUP
  variable 541
LSB_USER_BIND_CPU_LIST
  variable 541
LSB_USER_BIND_JOB
  variable 541
LSB_UTMP
  lsf.conf file 364
lsb.acct file 31
lsb.applications
  description 41
lsb.events file 74
lsb.hosts file
  description 111
time-based configuration 125
user group administrator 286
lsb.modules file 127
lsb.params
  description 132
  SUB_TRY_INTERVAL parameter 163
lsb.params file
  description 132
time-based configuration 132
lsb.queues
  POST_EXEC parameter 217
  PRE_EXEC parameter 218
lsb.queues file
  description 189
  time-based configuration 239
lsb.resources file
  description 240
  time-based configuration 271
lsb.serviceclasses file 273
lsb.users file
  description 284
  time-based configuration 291
LSF_ADD_CLIENTS
  install.config file 24
LSF_ADD_SERVERS
  install.config file 23
LSF_ADMINS
  install.config file 25
  slave.config file 501
LSF_AFS_CELLNAME
  lsf.conf file 365
LSF_AM_OPTIONS
  lsf.conf file 365
LSF_API_CONNTIMEOUT
  lsf.conf file 365
LSF_API_RECVTIMEOUT
  lsf.conf file 366
LSF_BMPLUGIN
  lsf.conf file 366
LSF_BIND_JOB
  lsf.conf file 42, 368
LSF_BINDIR
  cshrc.lsf and profile.lsf files 12
  lsf.conf file 368
LSF_BMPLUGIN
  lsf.conf file 368
LSF_CLUSTER_NAME
  install.config file 25
LSF_CMD_LOG_MASK
  lsf.conf file 369
LSF_CMD_LOGDIR
  lsf.conf file 370
  variable 541
LSF_CONF_RETRY_INT
  lsf.conf file 370
LSF_CONF_RETRY_MAX
  lsf.conf file 370
LSF_CONFDIR
  lsf.conf file 371
LSF_CPUSSETLIB
  lsf.conf file 371
LSF_CRASH_LOG
  lsf.conf file 372
LSF_DAEMON_WRAP
  lsf.conf file 372
LSF_DAEMONS_CPUS
  lsb.params file 372
LSF_DAEMONS_CPUS parameter in ego.conf 373
LSF_DEBUG_CMD
  lsf.conf file 373
LSF_DEBUG_CMD variable 541
LSF_DEBUG_LIM
  lsf.conf file 375
  variable 541
LSF_DEBUG_RES
  lsf.conf file 376
  variable 542
LSF_DHCP_ENV
  lsf.conf file 377
LSF_DISABLE_LSRUN
  lsf.conf file 377
LSF_DISPATCHER_LOGDIR
  lsf.conf file 378
LSF_DUALSTACK_PREFER_IPV6
  lsf.conf file 378
LSF_DYNAMIC_HOST_TIMEOUT
  lsf.conf file 379
LSF_DYNAMIC_HOST_WAIT_TIME
  install.config file 26
  lsf.conf file 380
LSF_EAUTH_AUX_DATA variable 542
LSF_EAUTH_AUX_PASS variable 542
LSF_EAUTH_CLIENT variable 542
LSF_EAUTH_KEY
  lsf.sudoers file 489
LSF_EAUTH_SERVER variable 542
LSF_EAUTH_UID variable 543
LSF_EAUTH_USER
  lsf.sudoers file 489
<table>
<thead>
<tr>
<th>Environment Variable</th>
<th>File Reference</th>
<th>Line</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSF_EEXEC_USER</td>
<td>lsf.sudoers file 490</td>
<td>490</td>
</tr>
<tr>
<td>LSF_EGO_ADMIN_PASSWD</td>
<td>lsf.sudoers file 490</td>
<td>490</td>
</tr>
<tr>
<td>LSF_EGO_ADMIN_USER</td>
<td>lsf.sudoers file 490</td>
<td>490</td>
</tr>
<tr>
<td>LSF_EGO_DAEMON_CONTROL</td>
<td>lsf.conf file 380</td>
<td>380</td>
</tr>
<tr>
<td>LSF_EGO_ENVDIR</td>
<td>lsf.conf file 381</td>
<td>381</td>
</tr>
<tr>
<td>LSF_ELIM_BLOCKTIME</td>
<td>lsf.cluster file 301</td>
<td>301</td>
</tr>
<tr>
<td>LSF_ELIM_DEBUG</td>
<td>lsf.cluster file 301</td>
<td>301</td>
</tr>
<tr>
<td>LSF_ELIM_RESTARTS</td>
<td>lsf.cluster file 302</td>
<td>302</td>
</tr>
<tr>
<td>LSF_ENABLE_CSA</td>
<td>lsf.conf file 381</td>
<td>381</td>
</tr>
<tr>
<td>LSF_ENABLE_DUALCORE</td>
<td>lsf.conf file 382</td>
<td>382</td>
</tr>
<tr>
<td>LSF_ENABLE_EGO</td>
<td>lsf.conf file 383</td>
<td>383</td>
</tr>
<tr>
<td>LSF_ENABLE_EXTSCHEDULER</td>
<td>lsf.conf file 383</td>
<td>383</td>
</tr>
<tr>
<td>LSF_ENABLE_SUPPORT_IPV6</td>
<td>lsf.conf file 384</td>
<td>384</td>
</tr>
<tr>
<td>LSF_ENVDIR</td>
<td>cshrc.lsf and profile.lsf files 12</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>lsf.conf file 384</td>
<td>384</td>
</tr>
<tr>
<td>LSF_EVENT_PROGRAM</td>
<td>lsf.conf file 384</td>
<td>384</td>
</tr>
<tr>
<td>LSF_EVENT_RECEIVER</td>
<td>lsf.conf file 385</td>
<td>385</td>
</tr>
<tr>
<td>LSF_EXECUTE_DOMAIN</td>
<td>variable 543</td>
<td>543</td>
</tr>
<tr>
<td>LSF_HOST_ADDR_RANGE</td>
<td>lsf.cluster file 302</td>
<td>302</td>
</tr>
<tr>
<td>LSF_HOST_CACHE_NTTTL</td>
<td>lsf.conf file 385</td>
<td>385</td>
</tr>
<tr>
<td>LSF_HOST_CACHE_PTTL</td>
<td>lsf.conf file 386</td>
<td>386</td>
</tr>
<tr>
<td>LSF_HPC_EXTENSIONS</td>
<td>lsf.conf file 386</td>
<td>386</td>
</tr>
<tr>
<td>LSF_HPC_PJL_LOADENV_TIMEOUT</td>
<td>lsf.conf file 391</td>
<td>391</td>
</tr>
<tr>
<td>LSF_ID_PORT</td>
<td>lsf.conf file 392</td>
<td>392</td>
</tr>
<tr>
<td>LSF_INCLUDEDIR</td>
<td>lsf.conf file 392</td>
<td>392</td>
</tr>
<tr>
<td>LSF_INDEP</td>
<td>lsf.conf file 392</td>
<td>392</td>
</tr>
<tr>
<td>LSF_INTERACTIVE_STDERR</td>
<td>lsf.conf file 393</td>
<td>393</td>
</tr>
<tr>
<td></td>
<td>variable 543</td>
<td>543</td>
</tr>
<tr>
<td>LSF_INVOKE_CMD</td>
<td>variable 543</td>
<td>543</td>
</tr>
<tr>
<td>LSF_JOB_STARTER</td>
<td>variable 544</td>
<td>544</td>
</tr>
<tr>
<td>LSF_LD_LIBRARY_PATH</td>
<td>variable 545</td>
<td>545</td>
</tr>
<tr>
<td>LSF_LD_SECURITY</td>
<td>lsf.conf file 394</td>
<td>394</td>
</tr>
<tr>
<td>LSF_LICENSE</td>
<td>install.config file 26</td>
<td>26</td>
</tr>
<tr>
<td>LSF_LICENSE_ACCT_PATH</td>
<td>lsf.conf file 397</td>
<td>397</td>
</tr>
<tr>
<td>LSF_LICENSEFILE</td>
<td>lsf.conf file 397</td>
<td>397</td>
</tr>
<tr>
<td>LSF_LICENSE_MAINTENANCE_INTERVAL</td>
<td>lsf.conf file 398</td>
<td>398</td>
</tr>
<tr>
<td>LSF_LICENSE_NOTIFICATION_INTERVAL</td>
<td>lsf.conf file 399</td>
<td>399</td>
</tr>
<tr>
<td>LSF_LIM_API_NTRIES</td>
<td>variable 545</td>
<td>545</td>
</tr>
<tr>
<td>LSF_LIM_DEBUG</td>
<td>variable 546</td>
<td>546</td>
</tr>
<tr>
<td>LSF_LIM_IGNORE_CHECKSUM</td>
<td>lsf.conf file 400</td>
<td>400</td>
</tr>
<tr>
<td>LSF_LIM_PORT</td>
<td>slave.config file 501</td>
<td>501</td>
</tr>
<tr>
<td>LSF_LIVE_CONFDIR</td>
<td>lsf.conf file 401</td>
<td>401</td>
</tr>
<tr>
<td>LSF_LOAD_PLUGINS</td>
<td>lsf.sudoers file 491</td>
<td>491</td>
</tr>
</tbody>
</table>
LSF_LOAD_USER_PROFILE
  lsf.conf file 402
LSF_LOCAL_RESOURCES
  lsf.conf file 402
LSF_LOG_MASK
  lsf.conf file 403, 405
LSF_LOGDIR
  lsf.conf file 405, 406
variable 546
LSF_LSLOGIN_SSH
  lsf.conf file 407
LSF_MACHDEP
  lsf.conf file 408
LSF_MANAGER product name
  lsf.cluster_name.license.acct file 315
LSF_MANDIR
  lsf.conf file 408
LSF_MASTER variable 546
LSF_MASTER_LIST
  install.config file 27
  lsf.conf file 409
LSF_MASTER_NSLOOKUP_TIMEOUT
  lsf.conf file 410
LSF_MAX_TRY_ADD_HOST
  lsf.conf file 410
LSF_MC_NON_PRIVILEGED_PORTS
  lsf.conf file 411
LSF_MISC
  lsf.conf file 411
LSF_MONITOR_LICENSE_TOOL
  lsf.conf file 411
LSF_MULTICLUSTER product name
  lsf.cluster_name.license.acct file 316
LSF_NIOS_DEBUG
  lsf.conf file 412
  variable 546
LSF_NIOS_DIE_CMD variable 546
LSF_NIOS_ERR_LOGDIR
  lsf.conf file 412
  variable 546
LSF_NIOS_IGNORE_SIGWINDOW variable 547
LSF_NIOS_JOBSTATUS_INTERVAL
  lsf.conf file 413
LSF_NIOS_MAX_TASKS
  lsf.conf file 413
LSF_NIOS_PEND_TIMEOUT variable 547
LSF_NIOS_PORT_RANGE variable 548
LSF_NIOS_RES_heartbeat
  lsf.conf file 414
LSF_NON_PRIVILEGED_PORTS
  lsf.conf file 414
LSF_PAM_APPL_CHKPT 415
  lsf.conf file 415
LSF_PAM_CLEAN_JOB_DELAY
  lsf.conf file 415
LSF_PAM_HOSTLIST_USE
  lsf.conf file 415
LSF_PAM_PLUGINDIR
  lsf.conf file 416
LSF_PAM_USE_ASH
  lsf.conf file 416
LSF_PASSWD_DIR
  lsf.conf file 416
LSF_PIM_INFODIR
  lsf.conf file 417
LSF_PIM_LINUX_ENHANCE
  lsf.conf file 417
LSF_PIM_SLEEPTIME
  lsf.conf file 418
LSF_PIM_SLEEPTIME_UPDATE
  lsf.conf file 418
LSF_POE_TIMEOUT_BIND
  lsf.conf file 419
LSF_POE_TIMEOUT_SELECT
  lsf.conf file 419
LSF_QUIET_INST
  install.config file 27
LSF_REMOTE_COPY_CMD 419
LSF_RES_ACCT
  lsf.conf file 420
LSF_RES_ACCTDIR
  lsf.conf file 421
LSF_RES_ACTIVE_TIME
  lsf.conf file 421
LSF_RES_CLIENT_TIMEOUT
  lsf.conf 421
LSF_RES_CONNECT_RETRY
  lsf.conf file 422
LSF_RES_DEBUG
  lsf.conf file 422
LSF_RES_PORT
  lsf.conf file 401
LSF_RES_RLIMIT_UNLIM
  lsf.conf file 423
LSF_RES_TIMEOUT
  lsf.conf file 423
LSF_RESOURCES variable 548
LSF_ROOT_Rex
  lsf.conf file 423
LSF_RSH
  lsf.conf file 424
LSF_SECUREDIR
  lsf.conf file 425
slave.config file 502
LSF_SERVERDIR
  cshrc.lsf and profile.lsf files 13
  lsf.conf file 426
LSF_SHELL_AT_USERS
  lsf.conf file 426
LSF_SHIFT_JIS_INPUT
  lsf.conf file 427
LSF_STARTUP_PATH
  lsf.sudoers file 491
LSF_STARTUP_USERS
  lsf.sudoers file 491
LSF_STRICT_CHECKING
  lsf.conf file 427
LSF_STRICT_RESREQ
  lsf.conf file 428
LSF_STRIP_DOMAIN
  lsf.conf file 428
LSF_TARDIR
  install.config file 28
  slave.config file 502
LSF_TIME_CMD
  lsf.conf file 429
LSF_TIME_LIM
  lsf.conf file 429
LSF_TIME_RES
  lsf.conf file 429
LSF_TMPDIR
  lsf.conf file 430
LSF_TOP
  install.config file 28
  slave.config file 504
LSF_TS_LOGON_TIME
  variable 548
LSF_ULDB_DOMAIN
  lsf.conf file 431
LSF_UNIT_FOR_LIMITS
  lsf.conf file 432
LSF_USE_HOSTEQUIV
  lsf.conf file 433
  variable 549
LSF_USER_DOMAIN
  lsf.conf file 434
  variable 549
LSF_USER_DOMAIN parameter 434
LSF_VPLUGIN
  lsf.conf file 435
lsf.cluster file 295
lsf.cluster_name.license.acct file 315
lsf.conf
  LSB_LIMLOCK_EXCLUSIVE parameter 332
  LSB_MBD_MAX_SIG_COUNT 349
  LSB_SACCT_ONE_UG 358
  LSB_AUTH parameter 367
  LSB_USER_DOMAIN parameter 434
lsf.conf file 317
  corresponding ego.conf parameters 317
  LSB_DISPLAY_YEAR 333
  LSB_QUERY_PORT parameter 158
  LSB_SUBK_SHOW_EXEC_HOST 362
lsf.licensescheduler file
  time-based configuration 479
lsf.shared file 481
lsf.sudoers file 487
lsf.task file 493
lsf.task.cluster file 493

d M

mail
  configuring on UNIX 342
MANDATORY_EXTSCHEd
  lsb.queues file 211
MASTER_INACTIVITY_LIMIT
  lsf.cluster file 305
MAX_ACCT_ARCHIVE_FILE
  lsb.params file 157
MAX_CONCURRENT_JOB_QUERY
  lsb.params file 158
MAX_EVENT_STREAM_FILE_NUMBER
  lsb.params file 159
MAX_EVENT_STREAM_SIZE
  lsb.params file 159
MAX_HOST_IDLE_TIME
  lsb.serviceclasses file 279, 280
MAX_INFO_DIRS
  lsb.params file 159
MAX_JOB_ARRAY_SIZE
  lsb.params file 160
MAX_JOB_ATTA_SIZE
  lsb.params file 161
MAX_JOB_MSG_NUM
  lsb.params file 175
MAX_JOB_NUM
  lsb.params file 161
MAX_JOB_PREEMPT
  lsb.applications file 51
  lsb.params file 161
  lsbqueues file 211
MAX_JOB_REQUEUE
  lsb.applications file 51
  lsb.params file 161
  lsbqueues file 211
MAX_JOBID
  lsb.params file 162
MAX_JOBINFO_QUERY_PERIOD
  lsb.params file 163
MAX_JOBS
  lsb.users file 289
MAX_PEND_JOBS
  lsb.params file 163
  lsb.users file 290
MAX_PREEXEC_RETRY
  lsb.applications file 52
  lsb.params file 164
  lsbqueues file 212
MAX_RSCHED_TIME
  lsbqueues file 212
MAX_SBD_CONNS
  lsb.params file 164
MAX_SBD_FAIL
  lsb.params file 165
MAX_SLOTS_IN_POOL
  lsbqueues file 213
MAX_TOTAL_TIME_PREEMPT
  lsb.applications file 52
  lsb.params file 165
  lsbqueues file 213
MAX_USER_PRIORITY
  lsb.params file 165
maximum
  job ID 162
mbatchd
  how to fix when busy 349
MBD_DIE record
  lsb.events 87
MBD_EGO_CONNECT_TIMEOUT
  lsb.params file 166
MBD_EGO_READ_TIMEOUT
  lsb.params file 166
MBD_EGO_TIME2LIVE
  lsb.params file 167
MBD_HEARTBEAT_INTERVAL
  lsfl.licensescheduler file Parameters section 449
MBD_QUERY_CPUS
  lsb.params file 167
MBD_REFRESH_INTERVAL
  lsfl.licensescheduler file Parameters section 450
MBD_REFRESH_TIME
  lsb.params file 168
MBD_SLEEP_TIME
  lsb.params file 169
MBD_START record
  lsb.events 86
MBD_USE_EGO_MJ
  lsb.params file 169
MC_PENDING_REASON_PKG_SIZE
  lsb.params file 170
MC_PENDING_REASON_UPDATE_INTERVAL
  lsb.params file 170
MC_PLUGIN_REMOTE_RESOURCE
  lsf.conf file 435
MC_PLUGIN_SCHEDULE_ENHANCE
  lsb.params file 170
MC_PLUGIN_UPDATE_INTERVAL
  lsb.params file 171
MC_RECLAIM_DELAY
  lsb.params file 172
MC_RUSAGE_UPDATE_INTERVAL
  lsb.params file 172
mem
  lsb.hosts file 114
  lsbqueues file 210
MEM
  lsb.resources file HostExport section 264
  lsb.resources file Limit section 246
MEMBER
  lsb.hosts file 122
MEMLIMIT
  lsb.applications file 53, 54
  lsbqueues file 213
  per parallel task 388
  per-job limit 338
MERGE_BY_SERVICE_DOMAIN
   lsf.licensescheduler file Parameters section 450

METHOD
   lsb.resources file ReservationUsage section 270

MIG
   lsb.hosts file 113
   lsb.queues file 55, 215

MIG record
   lsb.events 89

migrated jobs
   absolute job priority scheduling 190

MIN_SWITCH_PERIOD
   lsb.params file 172

mixed cluster
   specifying paths 352

model
   lsf.cluster file 308

MODELNAME
   lsf.shared file 483

MXJ
   lsb.hosts file 114

N

NAME
   lsb.hosts file 122
   lsb.resources file GuaranteedResourcePool section 259
   lsb.resources file Limit section 247
   lsb.resources file ResourceReservation section 267
   lsb.resources file SharedResourceExport section 265
   lsb.serviceclasses file 280
   lsf.licensescheduler file Feature section 456
   lsf.licensescheduler file ServiceDomain section 453

nd
   lsf.cluster file 308

NEW_JOB_SCHED_DELAY
   lsb.queues file 215

NHOSTS
   lsb.resources file HostExport section 263

NICE
   lsb.applications file 55
   lsb.queues file 215

NINSTANCES
   lsb.resources file SharedResourceExport section 265

NIOS
   standard message format 394
   nios.log.host_name 412

NO_PREEMPT_FINISH_TIME
   lsb.applications file 56
   NO_PREEMPT_INTERVAL
   lsb.applications file 56
   lsb.params file 174
   lsb.queues file 216

NO_PREEMPT_RUN_TIME
   lsb.applications file 57
   lsb.params file 174

NON_SHARED
   lsf.licensescheduler file ProjectGroup section 476

NON_SHARED_DISTRIBUTION
   lsf.licensescheduler file Feature section 468

NQS_QUEUES
   lsb.queues file 216

NQS_QUEUES_FLAGS
   lsb.params file 175

NQS_REQUESTS_FLAGS
   lsb.params file 175

O

OK license usage status
   bld.license.acct file 7
   lsf.cluster_name.license.acct file 316

OS types
   automatic detection 319

OVERUSE license usage status
   bld.license.acct file 7
   lsf.cluster_name.license.acct file 316

OWNERSHIP
   lsf.licensescheduler file ProjectGroup section 476

P

parallel jobs
   optimizing preemption of 178

PARALLEL_SCHED_BY SLOT 176

Parameters section
   lsf.licensescheduler file description 437

PATCH_BACKUP_DIR
   install.config file 28

PATCH_HISTORY_DIR
   install.config file 29

PEAK_INUSE_PERIOD
   lsf.licensescheduler file Feature section 469
   lsf.licensescheduler file Parameters section 450

PEND_REASON_MAX_JOBS 176

PEND_REASON_UPDATE_INTERVAL 176

Platform LSF Configuration Reference 567
PER_HOST
lsb.resources file HostExport section 262
lsb.resources file Limit section 248
PER_PROJECT
lsb.resources file Limit section 248, 249
PER_QUEUE
lsb.resources file Limit section 249
PER_USER
lsb.resources file Limit section 250
PERSISTENT_HOST_ORDER
lsb.applications 57
pg
lsb.hosts file 114
lsb.queues file 210
PG_SUSP_IT
lsb.params file 177
PJOb_LIMIT
lsb.queues file 217
PORT
lsf.licensescheduler file Parameters section 451
POST_EXEC
lsb.applications file
bsub -Ep 57
lsb.queues file
bsub -Ep 57
POST_EXEC parameter
lsb.applications 57
lsb.queues 217
PRE_EXEC
lsb.applications file 59
lsb.queues file 218
PRE_EXEC parameter
lsb.applications 59
lsb.queues 218
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, 151
JOB_POSTPROC_TIMEOUT parameter 49, 152
queue level
POST_EXEC parameter 217
PRE_EXEC parameter 218
PREEMPT_ACTION
lsf.licensescheduler file Parameters section 451
PREEMPT_FINISH_TIME
lsb.params file 174
PREEMPT_FOR
lsb.params file 177
PREEMPT_JOBTYPE
lsb.params file 179
PREEMPT_RESERVE
lsf.licensescheduler file Feature section 470
preemptable queue
defining 219
PREEMPTABLE_RESOURCES
lsb.params file 179
preemption
jobs by run time 178
of parallel jobs 178
PREEMPTION
lsb.queues file 219
PREEMPTION_WAIT_TIME
lsb.params file 179
preemptive queue
defining 219
preemptive scheduling
backfill jobs 179
enabling 219
exclusive jobs 179
jobs by run time 178
LSB_LIMLOCK_EXCLUSIVE parameter 332
per-host job slot limit for users and user groups 178
per-processor job slot limit for a user 178
per-processor job slot limit for user groups 177
PREEMPT_FOR parameter 177
PREEMPT_JOBTYPE parameter 179
total job slot limit for user groups 178
PREEXEC_EXCLUDE_HOST_EXIT_VALUES
lsb.params file 180
priority
queues 221
PRIORITY
lsb.queues file 221
lsb.serviceclasses file 280
lsf.licensescheduler file ProjectGroup section 477
lsf.licensescheduler file Projects section 479
privileged ports authentication
LSF_AUTH parameter 367
PROBE_TIMEOUT
lsf.cluster file 305
PROCESSLIMIT
lsb.applications file 59
lsb.queues file 222
PROCLIMIT
lsb.applications file 60
lsb.queues file 222

PRODUCTS
lsf.cluster file 305
profile.lsf file
description 9
setting the LSF environment 10

Project mode
CLUSTER_MODE parameter 439, 456

ProjectGroup section
lsf.licensescheduler file
description 475

Projects
PROJECTS
lsb.resources file Limit section 245, 251
lsf.licensescheduler file Projects section 479

Projects section
lsf.licensescheduler file
description 478

Q
QJOB_LIMIT
lsb.queues file 222

QUEUE_CTRL record
lsb.events 85

QUEUE_GROUP
lsb.queues file 223

QUEUE_NAME
lsb.applications file 55
lsb.queues file 223

queues
making preemptable 219
making preemptive 219
setting priority of 221

QUEUES
lsb.resources file GuaranteedResourcePool section 261
lsb.resources file Limit section 251

R
r15m
lsb.hosts file 114
lsb.queues file 210

r15s
lsb.hosts file 114
lsb.queues file 210

r1m
lsb.hosts file 114
lsb.queues file 210

RB PLUGIN
lsb.modules file 130

RCVJOBS_FROM
lsb.queues file 223

RECV_FROM
lsf.cluster file 314

RELEASE
lsf.shared file 486

REMOTE
lsb.users file 290
remote task list 493
remote tasks in task files 495

REMOTE_MAX_PREEXEC_RETRY
lsb.applications file 60
lsb.params file 180
lsb.queues file 224

REQUEUE_EXIT_VALUES
lsb.applications file 61
lsb.queues file 224

RERUNNABLE
lsb.applications file 61
lsb.queues file 225

RES_REQ
lsb.applications file 62
lsb.queues file 226

RES_SELECT
lsb.resources file GuaranteedResourcePool section 260
lsb.resources file HostExport section 262

ReservationUsage section
lsb.resources 269
reserve
custom resources 270
reserve resources 270

RESERVE_BY_STARTTIME
LSF HPC extensions parameter 387

RESOURCE
lsb.resources file Limit section 252
lsb.resources file ReservationUsage section 270

RESOURCE_RESERVE
lsb.queues file 225

RESOURCE_RESERVE_PER_SLOT
lsb.params file 181

RESOURCENAME
lsf.cluster file 313
lsf.shared file 484

ResourceReservation section
lsb.resources 266

RESOURCES
RESRSV_LIMIT
lsb.queues file 228
RESUME_COND
lsb.queues file 229
RETENTION_FACTOR
lsf.licensescheduler file Feature section 470
RETRY_LIMIT
lsf.cluster file 306
REXPRI
lsf.cluster file 310
rollover
job IDs 162
RTASK_GONE_ACTION
lsb.applications file 67
RUN_JOB_FACTOR
lsb.params file 181
lsb.queues file 229
RUN_TIME_DECAY
lsb.params file 182
lsb.queues file 230
RUN_TIME_FACTOR
lsb.params file 182
lsb.queues file 230
RUN_WINDOW
lsb.queues file 231
RUNLIMIT
lsb.applications file 68
lsb.queues file 231
RUNTIME
lsb.applications file 69
RUNWINDOW
lsf.cluster file 310

S
SBD_SLEEP_TIME
lsb.params file 182, 183
SBD_UNREPORTED_STATUS record
lsb.events 100
SCH_DISABLE_PHASES
lsb.modules file 130
SCH_PLUGIN
lsb.modules file 128
schmod_adsrsv 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 427
sendmail program 342
server
lsf.cluster file 310
Servers
lsf.shared file 481
service class
examples 282
SERVICE_DOMAINS
lsf.licensescheduler file Feature section 471
ServiceDomain section
lsf.licensescheduler file
description 452
setup.config file 496
seven-digit job ID 162
SharedResourceExport section
lsb.resources 265
SHARES
lsf.licensescheduler file ProjectGroup section 476
SHORT_PIDLIST
LSF HPC extensions parameter 388
SLA scheduling
service classes
examples 282
SLA_GUARANTEES_IGNORE
lsb.queues file 232
SLA_TIMER
lsb.params file 183
slave.config file 499
SLOT_POOL
lsb.queues file 232
SLOT_reserve
lsb.queues file 233
SLOT_SHARE
lsb.queues file 234
SLOTS
lsb.resources file HostExport section 264
lsb.resources file Limit section 253
SLOTS_PER_HOST
  lsb.resources file GuaranteedResourcePool section 261
SLOTS_PER_PROCESSOR
  lsb.resources file Limit section 254
SNDJobs_TO
  lsb.queues file 234
SSCHED_ACCT_DIR
  lsb.params file 184
SSCHED_MAX_RUNLIMIT
  lsb.params file 184
SSCHED_MAX_TASKS
  lsb.params file 185
SSCHED_REQUEUE_LIMIT
  lsb.params file 185
SSCHED_RETRY_LIMIT
  lsb.params file 185
SSCHED_UPDATE_SUMMARY_INTERVAL
  lsb.params file 186
STACKLIMIT
  lsb.applications file 69
  lsb.queues file 234
STANDBY_CONN_TIMEOUT
  lsf.licensescheduler file Parameters section 451
STOP_COND
  lsb.queues file 235
STRIC_T_UG_CONTROL
  lsb.params file 187
STRIC_T_UG_CONTROL parameter
  lsb.params file 285, 286
SUB_TRY_INTERVAL
  lsb.params file 187
SUB_TRY_INTERVAL parameter in lsb.params 163
SUCCESS_EXIT_VALUES
  lsb.applications file 69
SUSPEND_CONTROL
  lsb.applications file 66, 70, 71
SWAP
  lsb.resources file HostExport section 264
SWAPLIMIT
  lsb.applications file 71
  lsb.queues file 235
  per parallel task 388
swp
  lsb.hosts file 114
  lsb.queues file 210
SWP
  lsb.resources file Limit section 255
SYSTEM_MAPPING_ACCOUNT
  lsb.params file 188
T
  task files
    description 493
    format 495
    permissions 494
    sections 495
  task lists
    files 493
    remote 493
    viewing 494
TASK_MEMLIMIT
  LSF HPC extensions parameter 388
TASK_SWAPLIMIT
  LSF HPC extensions parameter 388
TERMINATE_WHEN
  lsb.queues file 235
THREADLIMIT
  lsb.applications file 72
  lsb.queues file 236
time windows
  syntax 268
TIME_WINDOW
  lsb.resources file ResourceReservation section 268
time-based configuration
  lsb.hosts 125
  lsb.params 132
  lsb.queues 239
  lsb.resources 271
  lsb.users 291
  lsf.licensescheduler 479
tmp
  lsb.hosts file 114
  lsb.queues file 210
TMP
  lsb.resources file Limit section 256
troubleshoot
  host groups 201
troubleshooting
  cluster performance 349
type
  lsf.cluster file 311
TYPE
  lsb.hosts file 123
  lsb.resources file GuaranteedResourcePool section 259
lsb.resources file HostExport section 264
lsf.shared file 485

TYPENAME
lsf.shared file 482

U

JOB_LIMIT
lsb.queues file 237

ULDB (IRIX User Limits Database)
description 431
jlimit.in file 431

UNFULFILL record
lsb.events 87

UNIX/Windows user account mapping
   LSF_EXECUTE_DOMAIN parameter 543
   LSF_USER_DOMAIN parameter 434

untrusted environments 427

USE_PRIORITY_IN_POOL
lsb.queues file 237

USE_SUSP_SLOTS
lsb.params file 188
user group administrator 286
user groups
   hierarchical fairshare 287
user profiles
   loading for a job 402

User section
lsb.users file 288

USER_NAME
lsb.users file 289

USER_SHARES
lsb.hosts file 121
lsb.users file 287

UserGroup section

lsb.users file 284
UserMap section
lsb.users file 290

users
   overlapping members 145

USERS
lsb.queues file 238
lsb.resources file Limit section 257
lsb.resources file ResourceReservation section 269
lsb.serviceclasses file 281

t
lsb.hosts file 114
lsb.queues file 210

V

variables. See environment variables

W

windows
time 268
Windows and UNIX
   enable mixed paths 352
WORKLOAD_DISTRIBUTION
   lsf.licensescheduler file Feature section 471

X

XLSF_APPDIR
lsf.conf file 436
XLSF_UIDDIR
cshrc.lsf and profile.lsf files 14
lsf.conf file 436