Platform LSF™ Desktop Support™ Administrator's Guide

Version 7 Update 2
Release date: November 2007
Last modified: December 4 2007
Support: support@platform.com
Comments to: doc@platform.com

Platform

Copyright © 1994-2011, Platform Computing Inc.

We'd like to hear from you

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

Your comments should pertain only to Platform documentation. For product support, contact support@platform.com.

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

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

Document redistribution and translation

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

Internal redistribution

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

Trademarks

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

PLATFORM COMPUTING, PLATFORM SYMPHONY, PLATFORM JOBSCHEDULER, PLATFORM ENTERPRISE GRID ORCHESTRATOR, PLATFORM EGO, PLATFORM VM ORCHESTRATOR, PLATFORM VMO, ACCELERATING INTELLIGENCE, and the PLATFORM and PLATFORM LSF logos are trademarks of Platform Computing Inc. in the United States and in other jurisdictions.

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

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

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

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

Macrovision, Globetrotter, and FLEX*lm* are registered trademarks or trademarks of Macrovision Corporation in the United States of America and/or other countries.

Topspin is a registered trademark of Topspin Communications, Inc.

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

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

Third Party License Agreements

http://www.platform.com/legal-notices/third-party-license-agreements

Contents

CHAPTER 1

Introducing Platform LSF Desktop Support

This chapter describes LSF desktop support, previously known as ActiveCluser, and explains how it works.

Contents

About This Guide

This guide is your starting point for learning how to use and manage Platform LSF desktop support. It provides an overview of LSF desktop support concepts, licensing information, how to install and configure LSF desktop support, some basic commands, and some troubleshooting tips.

Who should use this guide

This guide is written for LSF desktop support administrators who want to familiarize themselves with the fundamentals of managing and using an LSF desktop support environment.

What you should already know

This guide assumes that you are familiar with basic LSF administration, and that you are familiar with basic LSF concepts such as clusters, jobs, servers, and hosts.

How this guide is organized

Chapter 1 "Introducing Platform LSF Desktop Support" introduces Platform LSF desktop support, the various components, architecture and configuration, and basic data flow.

Chapter 2 "Planning the Installation" describes issues you need to consider when planning your Platform LSF desktop support installation.

Chapter 3 "Installing LSF Desktop Support" guides you through the installation process.

Chapter 4 "Configuring the Components in LSF Desktop Support" describes the configuration parameters and the ways you use them to configure your environment.

Chapter 5 "Managing LSF Desktop Support" describes those procedures you may use periodically to maintain your environment.

Chapter 6 "Troubleshooting" provides some troubleshooting tips.

Overview

Platform LSF desktop support enables you to take advantage of unused computing cycles on standard Microsoft Windows 2000, Windows 2003 Server, and Windows XP desktop client computers. LSF desktop support replaces the standard LSF job execution mechanism with its own: running large, compute-intensive jobs across Windows desktop clients.

Security and reliability

LSF desktop support maintains the integrity and security of the Windows desktop client—it has no authority on the desktop client, so it cannot access data besides its own files. LSF desktop support provides automatic failure recovery—it automatically restarts failed or canceled jobs on desktop clients.

Scalability

LSF desktop support has undergone extensive testing in a 16-server environment with over 75,000 desktop clients across the Internet.

Using a desktop server with a fast processor and 2 GB of memory, you can run 2000 half-hour jobs per hour on 1000 desktop clients.

Resource-aware scheduling

LSF desktop support uses resource-aware scheduling to match jobs to desktop clients with the appropriate resources. LSF desktop support has built-in resource definitions and also provides support for user-defined resources.

Automatic upgrade

Desktop clients are automatically upgraded from a central desktop server, reducing maintenance time and effort.

Web interface

LSF desktop support provides Web browser interfaces for job submission and to see both job progress and environment status.

Laptop support

LSF desktop support uses ACPI-compliant power management features on laptops, so it does not run a battery down when the laptop is not plugged in.

Applications and data

LSF desktop support can be used to run any application, provided that application can be made available on the desktop client. Applications and data do not need to reside on the desktop client—they can be transferred with the job that runs them. However, data files are cached so that they are not transferred multiple times, tying up valuable network bandwidth.

For security purposes, you can specify that applications running on the desktop client should not write to the cache.

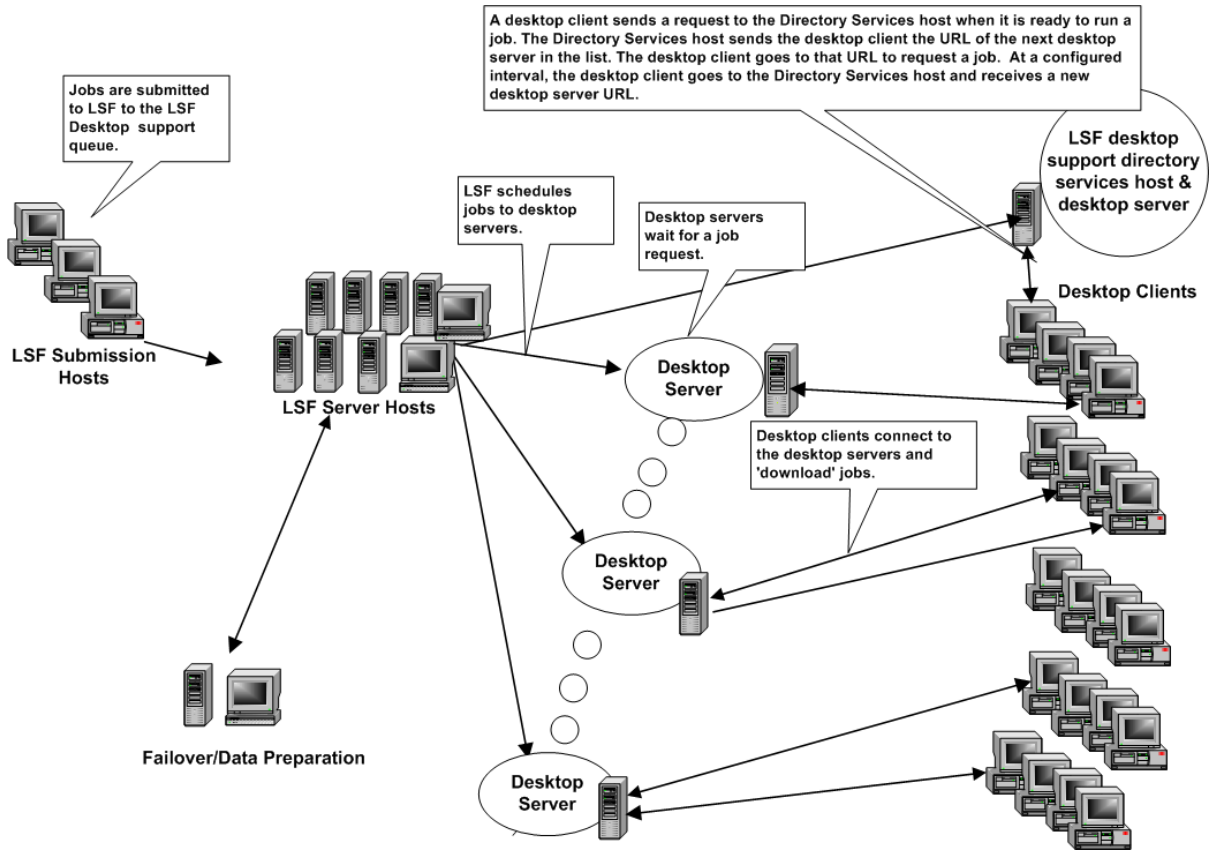
Components

An LSF desktop support environment consists of the following components:

- LSF master host
- Desktop servers
- Desktop clients
- Directory Services host—when load balancing between desktop servers is used

When load balancing of desktop servers is used

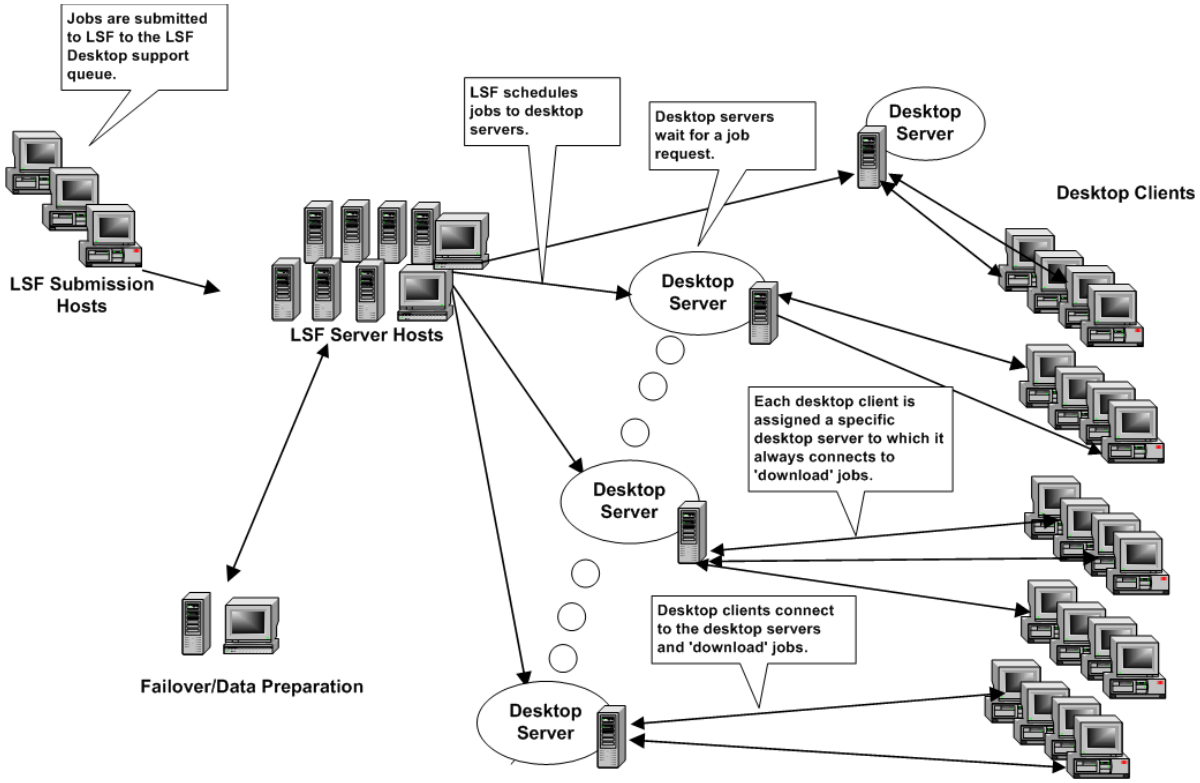
The following diagram illustrates the components in an LSF desktop support environment that uses load balancing between desktop servers via a Directory Services host:



For more information about load balancing, see “Load Balancing and the Directory Services Host” on page 18.

When load balancing of desktop servers is not used

The following diagram illustrates the components in an LSF desktop support environment when load balancing between desktop servers is not used:



LSF master host

LSF desktop support uses the LSF master host for scheduling. Users submit jobs to an LSF desktop support queue, and the LSF master host selects the appropriate desktop server, and dispatches the job to it.

Directory Services host

If load balancing of desktop servers is enabled, Directory Services hosts are configured, which provides load balancing within the LSF desktop support environment. When a desktop client is ready for its first job, it goes to the Directory Services host, where it receives the URL for the next desktop server. The desktop client then goes to that URL to request a job. When the job is completed, the desktop client returns its results to the desktop server. At a configured interval (the default is every 24 hours), the desktop client goes back to the Directory Services host, where it receives a new desktop server URL and any updates to its binary files.

Desktop server

The desktop server is part of an LSF cluster, and it is sometimes referred to as the *MED*. It receives jobs to dispatch from the LSF master host, and awaits a request for a job from a desktop client. A desktop client goes to the desktop server and downloads a job and any related files. When the job completes, the desktop client passes job status and job results to the desktop server, which in turn passes it to the LSF master host.

The desktop server uses an Apache Web server to serve files and a Tomcat Java application server to make the jobs available to the desktop clients and to provide environment status to you and your users.

The desktop server supports a set of Web pages that provide status information about both activity within the LSF desktop support environment and individual job status.

The desktop server is automatically monitored and restarted in the event of a failure. It automatically obtains its configuration parameters when restarted, so it is easily reconfigured.

Reduced network traffic

LSF dispatches groups of jobs at a time using job-level compression, to minimize network traffic. The desktop server avoids network traffic by:

- Caching files that are common to many jobs
- Supporting staging of files during non-busy network periods
- Maintaining minimal communication between desktop server and desktop clients
- Performing no polling—desktop clients initiate all communication with the desktop server

The desktop clients cache reusable input and reference files.

Desktop client

The desktop client resides on Windows machines and is sometimes referred to as an *SED*. If load balancing is used, the desktop client obtains the URL of the desktop server it should connect to from the Directory Services host. This URL is updated at pre-configured intervals. The default is 24 hours. If load balancing is not used, the desktop client is assigned a URL for a desktop server at installation time.

The desktop client polls the desktop server at preset intervals, requesting a job. The desktop client reports its available resources. If a job is waiting, and the desktop client has sufficient resources to run the job, the desktop client downloads the job from the Tomcat server and any files required for the job from the Apache Web server. It runs the job and, at its next polling interval, it returns job status and results to the Tomcat server. If the job has completed, it requests another job.

The desktop client can run jobs in the background at all times. It runs jobs at the lowest priority to minimize interference with the desktop client user.

Generally, the user is not aware of the jobs running, unless the machine has limited memory available, or if a job requires transfer of one or more large files. However, if a user feels that a job is impacting the performance of the machine, that user can temporarily remove the desktop client from the LSF desktop support environment. The desktop client remains out of the LSF desktop support environment for a time period configured by the administrator, or until the user explicitly resumes the desktop client processing.

Controlling desktop client processing

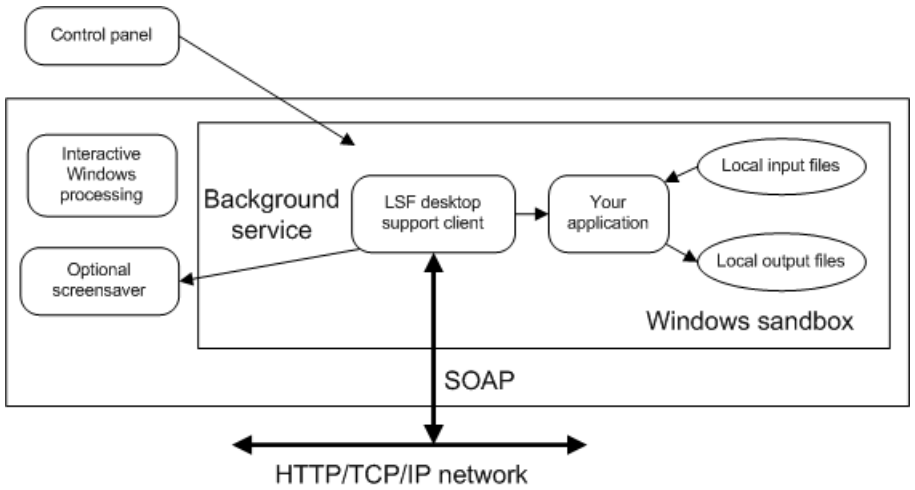
The desktop client can run in one of four modes:

- Always mode, where the desktop client runs jobs in the background at all times. The desktop client runs jobs at the lowest priority to minimize interference with the desktop client user.
- Screen saver mode, where the desktop client runs jobs only when a screen saver is active (default).

- Logon mode, where the desktop client runs jobs only when no user is logged on to the desktop client. A user is logged on during an interactive session initiated from a Windows logon dialog box.
- Idle mode, where the desktop client runs jobs only when the desktop client has been idle for a configurable period.

When running in any mode where the desktop client is allowed control, the user can pause the desktop client if required. At any time, the user can then resume desktop client processing.

Under the LSF desktop support hood



The desktop client automatically restarts after a system failure or reboot, and automatically obtains its configuration parameters from the Web server.

User interface

The desktop client can provide an interface for the desktop client user, accessed from an LSF desktop support system tray icon:

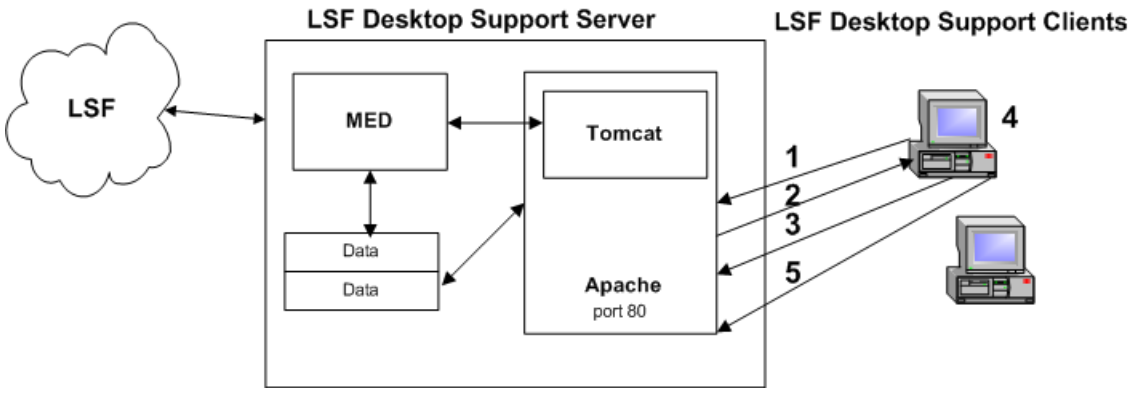
- Right-clicking the system tray icon opens the menu, where the user can pause or resume desktop client processing, go to the **Details** dialog, or hide the system tray icon.
- Double-clicking the system tray icon opens the **Details** dialog, where the user can specify the desktop client mode, as described above.

Desktop client user ID

The desktop client runs jobs under a user ID specific to LSF desktop support. By default, this user ID is `seduser`, but you can change the user ID as part of a silent installation. The desktop client runs using the SED service. The desktop client installer modifies local security settings to grant the **Log on as batch job privilege** to the LSF desktop support user ID. If the target machine is in a Windows 2000 domain, and the Domain Group Policy settings overwrite this privilege, the desktop client may be unable to work.

Data Flow

The data flow process discussed here occurs in an LSF desktop support environment, whether or not load balancing is performed. It occurs after the desktop client receives the URL designating from which desktop server to request a job.



The LSF desktop support job cycle is as follows:

1. The desktop client reports to Apache that it is idle and provides its available resources (memory, disk, number and type of processors, processor speed).
2. Apache forwards the desktop client resource information to Tomcat, and Tomcat returns an appropriate job to the desktop client through Apache.
3. The desktop client gets files from Apache.
4. The desktop client runs a job.
5. The desktop client returns results and job status to Apache, and Apache forwards the job status to Tomcat.

Submitting a job

1. The user submits a job to an LSF desktop support queue (the default is AC_QUEUE), and specifies the applications and data files and any resources required to run the job.
2. LSF dispatches the job to a desktop server based on the queue specified.
3. The job is pushed to the desktop server.
4. The desktop server waits for a qualified desktop client to request a job, and gives the desktop client the URL to locate the job.
5. The desktop client connects to the URL and downloads the job and its associated applications and data.

Running a job

1. The desktop client runs the job at its first opportunity.
2. The desktop client returns the status of the job to the desktop server.

Type of jobs you can run

LSF desktop support can run jobs that execute the following types of commands:

- `.cmd`
- `.bat`
- `.exe`

Typically, LSF desktop support runs jobs that require more than a few minutes to run.

All jobs that run within an LSF desktop support environment are assumed to be re-runnable, i.e., if a job overruns its run limits on a desktop client, the desktop client terminates it, and the desktop server re-dispatches the job to another desktop client.

Types of applications you can run

LSF desktop support can run commercial off-the-shelf applications, and it can distribute any Windows or DOS application to a desktop client. Some commercial applications require pre-installation on the desktop client. For example, Java applications require JVM and Java runtime libraries.

Any applications that a desktop client is requested to execute while running a job must be transferred to the desktop client, or must reside in the Program Files directory.

LSF desktop support supports all scripting and interpretive languages—VBScript, Perl, Python, Windows batch files, and Java and other compile-on-demand languages.

Changing desktop client user rights

By default, the desktop client user (by default, `seduser`) runs with the right to **Log on as a batch job**. Certain scripting languages may require additional user rights to function correctly. For example, CScript, a version of Windows Scripting Host, requires the desktop client user to have the right to **Log on locally**.

The exact method for changing user rights depends on the version of Windows. The following example changes the rights of the default `seduser` user to run CScript in Windows 2000 Professional:

1. Click **Start > Settings > Control Panel > Administrative Tools > Services > SED**.
2. Select **Action > Stop** to stop the desktop service.
3. Return to the **Administrative Tools** window, and select **Local Security Policies**.
4. In the Local Security Settings window, expand **Local Policies > User Rights Assignment**.
5. Allow the `seduser` user to log on locally:
   a. Double-click **Log on locally**.
   b. Click **Add**.
   c. Enter **`seduser`**, and click **OK**.
6. Return to the **Administrative Tools** window, and select **Services > SED**.
7. Select **Action > Start** to restart the desktop service.

See Chapter 4, “Configuring the Components in LSF Desktop Support” for information about configuring Windows Group Policies for SED users.

Execution limits

The user can place limits on jobs:

- Execution limits and elapsed time limits, set in the LSF desktop support queue or at job submission time

- Memory limits, set in the LSF desktop support queue or at job submission time
- Disk limits, number and type of CPUs required

Submitter can control jobs

Users can see the status of their jobs, and kill jobs, if necessary.

The following LSF commands are supported within an LSF desktop support environment:

- `badmin`
- `bhist`
- `bhosts`
- `bjobs`
- `bkill` (most options)
- `bqueues`
- `bsub` (used to submit jobs)

Note: In earlier versions, the LSF desktop support-only `asub` command was also used to submit jobs directly to the LSF desktop support queue.The asub command is obsolete.

- `busers`
- `lsadmin`

Order in which jobs are run

Generally, LSF desktop support jobs run on a first-in, first-out basis. However, if a user specifies particular desktop client resource requirements when submitting a job, the order in which the jobs run depends on the availability of desktop clients that meet the specified requirements.

Configuration Overview

LSF desktop support uses the following configuration files:

LSF desktop support configuration files

Configuration File	Location	Description	For more information, see
`install.config`	*<LSF Server Host>*`/install`	Specifies the configuration settings Configuring this file is part of the process that converts an LSF Server to a desktop server.	"Converting the LSF Server Host to a Desktop Server" on page 25
`med.conf`	*<desktop server>*`/etc`	Configures the desktop server after installation, specifying the URLs required by the desktop server and the desktop clients for file and data transfer and for status reporting.	"Desktop Server Configuration Settings" on page 60
`sedsetup.cmd`	User-defined, in a central location, in the same directory containing `lsf7Update2_desktop.msi`, which used to run the silent installation on the desktop client.	Specifies the configuration settings for the desktop client before running the silent installation on the desktop client.	"Installing the Desktop Client" on page 33
`SEDConfig.xml`	*<Directory Service Server>* / *<directory specified by CONFIGDIRECTORY in* `install.config`*>*	Configures all desktop clients after installation.	"Configuring Desktop Clients" on page 44
`wscache.conf`	`TOMCAT_HOME`	Specifies location of job completion log files.	"Changing Job Completion Log Location" on page 55

LSF configuration files

Configuration File	Location	Description	For more information, see
`lsb.queues`	By default, this file is installed in `LSB_CONFDIR/`*`cluster_name`*`/configdir`.	◆ Configure to exclude desktop servers from LSF queues. ◆ Configures the default LSF desktop support queue (`AC_QUEUE`). ◆ Configures additional LSF desktop support queues.	◆ "Excluding Desktop Servers from LSF Queues" on page 30 ◆ "Configuring the Default LSF Desktop Support Queue" on page 57 ◆ "Creating an Additional LSF Desktop Support Queue" on page 59

CHAPTER 2

Planning the Installation

This chapter describes how to plan the installation of LSF desktop support to ensure it meets your user and system requirements.

Contents

Load Balancing and the Directory Services Host

Load balancing evenly distributes the available desktop clients among the desktop servers, so that no desktop server is overloaded with requests for jobs or status updates at any time. You can use two methods of load balancing within an LSF desktop support environment:

- Automatic load balancing
- Manual load balancing

Before you install LSF desktop support, you need to decide which method of load balancing to use.

When should you use automatic load balancing?

You should consider automatic load balancing under the following circumstances:

- Your environment includes a very large number of desktop clients, e.g. 1,000 desktop clients, and the number of available desktop clients varies dramatically over time.

  OR

- You are setting up multiple desktop servers.

How does automatic load balancing work?

Automatic load balancing uses one or more Directory Services hosts to manage a large network of desktop clients, pointing desktop clients to desktop servers via a 'round-robin' process. Usually, one Directory Services host is used. The Directory Services host uses a list of desktop servers that is created during installation. Each time a desktop client contacts the Directory Services host (at startup, and again at the configured time interval), the desktop client receives a new URL for the desktop server to contact.

Alternatively, you can set up multiple desktop servers as directory servers and round-robin between them.

If you choose to use automatic load balancing

If you choose to use automatic load balancing, then you must install at least one Directory Services host in your LSF desktop support environment. Therefore, when you install that desktop server, you must specify that it is a Directory Services host, using the **`DIRECTORYSERVICE=yes`** in `install.config`. The remaining desktop servers should be installed without the Directory Services option, by specifying **`DIRECTORYSERVICE=no`** in `install.config`.

Tip about specifying a failover host: If you are working with multiple Directory Services hosts with the same configuration, then you can create a DNS round-robin alias to point to these Directory Services hosts.

If you choose not to use automatic load balancing

If you choose not to use automatic load balancing, then every desktop server is configured as a Directory Services host. This means that it sends every desktop client that contacts it to its own desktop server URL. This is the default Directory Services installation option.

How does manual load balancing work?

You can achieve a certain amount of manual load balancing by using multiple desktop servers, and configuring a balanced number of desktop clients to report to each desktop server for work. This process requires advance planning, and may require adjusting the desktop client configuration at installation time accordingly.

Security

Over the network

LSF desktop support uses standard protocols to ensure network security. The desktop client-to-host communication uses XML-based SOAP protocol, which is Microsoft.NET compatible. SSL Encryption can be used to protect communication.

At the desktop client

A desktop client user is prevented from accessing LSF desktop support data on the desktop client by using a Windows sandbox with permissions to read/write only to the sandbox. The desktop client runs as a background Windows service, using a different user ID than the desktop client end user. This prevents the desktop client from accessing user data.

Cache control

LSF desktop support uses the Microsoft Internet Explorer cache to reduce the number of file transfers required for a series of jobs. The size and expiry policy for the care are taken from the Microsoft Internet Explorer settings for the `Default User` of the operating system.

By default, all files are cached. You can change the settings so that no files are cached. Alternatively, you can specify caching settings for each input file during job submission, as described in the *Platform LSF Desktop Support User's Guide*.

Increased security with HP ProtectTools Authentication Services

LSF desktop support increase security with HP ProtectTools Authentication Services, which mitigate security risks with strong user authentication including a customer-unique password hashing system, managed change of administration passwords, last successful and unsuccessful login information, multiple login denial, and timed auto log-out.

Requirements

Each component in the LSF desktop support environment has both hardware and software requirements. Before installing, make sure your environment meets the requirements described below

LSF Master Batch Daemon

Software

LSF desktop support requires LSF 6.2 and later.

Desktop server

Software

- Supported operating systems:
  - Linux 2.4 glibc 2.2 (x86, AMD64)
  - Linux 2.4 glibc 2.3 (x86, AMD64)
  - Linux2.6-glibc2.3 (x86, AMD64)
- J2RE (Java Runtime Environment) is installed with the desktop server.
- Make sure that `root` can write to the installation directory.
- Linux installations: Apache requires `/lib/libdb.so.3` (the Berkeley DB database library version 3.0). If this is not included with your Linux distribution, you can install it from `misc/examples`.

Hardware

- 256 MB memory
- 1 GB free disk space for installation, but 20 GB is recommended for file caching
- Fast processor

Desktop clients

Software

- Supported operating systems:
  - Microsoft Windows XP
  - Microsoft Windows 2000 Professional with Service Pack 4
  - Microsoft Windows 2003 server
- Microsoft Internet Explorer 6.0 or later
- The desktop client installation checks for the DLL versions of Microsoft C Runtime Library, Microsoft C++ Runtime Library, and the Microsoft XML parser library.

  If the DLL versions are older than those required by the desktop client, the DLLs are replaced. The required DLLs are:
  - `msvcrt.dll version 6.0.8797.0`
  - `msvcp60.dll version 6.0.8168.0`
  - `msxnl3.dll version 8.10.8308.0`
  - `msxnl3r.dll version 8.10.8308.0`
  - `msxnl3a.dll version 8.10.8308.0`
  - `ExtRes.dll` (used for user-defined resource reporting)
- Update MSXML: You should install the latest MSXML 3.4 and higher patches and hotfixes on each desktop client, which are available at `http://www.msdn.microsoft.com` (search for MSXML).

Hardware

- 1 GB of free disk space to run the desktop client
- 10 MB free disk space on the system drive

Determining the Applications Required on Desktop Clients

Sometimes a job may require a specific application before it can run on a desktop client machine. The application and any corresponding data files must reside on the desktop client when the job runs. In general, the user can point to the required applications and data files when submitting the job, and the application and data are sent to the desktop client along with the request to run the job. However, if installing the application requires rebooting the desktop client machine, then the application must be physically installed on the desktop client machine before running the job. Installing the application in advance also reduces the amount of data sent across your network.

Before starting this task, you need to know the following information:

- The types of applications to run on the desktop clients
- What, if any, prerequisite software is required to run the applications, such as a Java runtime environment (JRE)

To determine the required applications:

1. Obtain a list of applications that are expected to run on the desktop clients.
2. Verify the prerequisite software required to run each application.
3. Install the prerequisite software in the Program Files directory on all desktop clients that may be used to run each application. Make sure all desktop clients for a particular desktop server have the required software installed, since a job may run on any desktop client in the LSF desktop support environment.

CHAPTER 3

Installing LSF Desktop Support

This chapter describes how to install LSF desktop support. To ensure a complete and error-free installation, use the “Installation Overview” on page 24.

Contents

Installation Overview

These are the steps you complete when installing LSF desktop support components.

To install all components:

- If this is a new LSF installation, follow the installation instructions in *Installing Platform LSF on UNIX and Linux*, and install an LSF master host.
- Specify each host where you will install a desktop server as an LSF server host, by editing the `LSF_ADD_SERVERS` parameter in `install.config`.

  Although these are LSF server hosts, they are dedicated servers for LSF desktop support and cannot be used for standard LSF operations.

  See *Installing Platform LSF on UNIX and Linux* and the *Platform LSF Reference* for more information about parameters in `install.config`.
- Install LSF on a local file system on each host where you will install a desktop server. Each of these LSF installations should reference the central, common LSF configuration files.
- You must know the fully qualified domain names of these hosts in DNS.
- Create a DNS alias including both the server and client configuration for all desktop servers.
- Add any desktop servers to round-robin to `MED.lst`. The path of this file is specified by the `CONFIGDIRECTORY` parameter in `install.config`.
- Make sure that the LSF cluster, including the LSF server host added earlier, is running.
- Convert each new LSF server host to a desktop server. If you plan to enable automatic load balancing, make sure to install at least one server as a Directory Services host.

  If you will not be using automatic load balancing, make sure you install every server as a Directory Services host. See “Converting the LSF Server Host to a Desktop Server” on page 25 for instructions.
- Enable LSF desktop support on the LSF master host. See “Enabling Desktop Support on LSF” on page 28 for instructions.
- Verify the Web servers are running. See “Checking Web Server Availability” on page 31 for instructions.
- Install the desktop client software on each computer where it will run jobs. See “Installing the Desktop Client” on page 33 for instructions.
- During installation, a default LSF desktop support queue, AC_QUEUE, is created. If you require additional LSF desktop support-dedicated queues, add them now, ensuring they are compatible with LSF desktop support. See “Creating an Additional LSF Desktop Support Queue” on page 59 for instructions.
- Prevent desktop servers from running standard LSF jobs. See “Excluding Desktop Servers from LSF Queues” on page 30 for instructions.
- Distribute `lsf_desktop_user.pdf` to LSF desktop support end users, or place it in a central location.

Converting the LSF Server Host to a Desktop Server

These instructions explain how to convert an LSF server host to a desktop server, once:

- An LSF server host is installed
- The LSF cluster is up and running.

You must follow these instructions for each LSF server host to convert to a desktop server.

This process:

- Installs LSF desktop support, Apache 1.3.33, Tomcat 4.1.30, and Java 1.4.1. Apache, Tomcat, and Java are installed within the LSF desktop support directory tree.
- Creates the `/etc/med.conf` file.

To convert to a desktop server:

1. Prepare the LSF server host for conversion to a desktop server:
   a. LSF desktop support requires dedicated Web servers (Apache and Tomcat). If you have either or both of those servers installed on this host, LSF desktop support can use them, provided they use the correct ports. LSF desktop support requires that Apache and Tomcat use port 80. If you do not have existing Web servers on this host, the installation script installs them.
   b. If you have Web servers currently running on this host, shut them down.
   c. Make sure no other applications are using port 80. These must be dedicated for use by the desktop server itself.
2. Untar the software distribution package in the directory on the LSF server host where you will install the desktop server. For example:

   ```
   cd AC_INSTALL
   zcat lsf7Update2_desktop_linux-x86.tar.Z | tar xvf -
   ```

3. Specify the configuration parameters. On the LSF server host, in the `install` directory, edit the file `install.config`: (The entries can be in any order, and the actual order in the sample `install.config` may vary.)
   - LSF_ENVDIR parameter. Required. Specify the location of the LSF configuration file `lsf.conf`, which is a shared file. For example:

     ```
     LSF_ENVDIR="/usr/local/lsf/mnt/conf"
     ```

   - ACH_TOP parameter. Required. Specify the full path to the top-level installation directory where you want the desktop server software installed. For example:

     ```
     ACH_TOP="/usr/local/lsfac"
     ```

     We recommend that *ACH_TOP* be within the local file system.
   - STATSDIRECTORY parameter. Required. Specify the path to the directory where you want all job completion logs stored. This is also the location where directory service connection logs are stored if directory services is enabled. It is recommended that this be a shared directory (for example NFS).
   - SERVER_NAME parameter. Optional. Specify a valid DNS name or IP address for your desktop server if it is different from the specified host name. If your host does not have a registered DNS name, enter its IP address here. This is the name sent back to Web clients. For example:

```
SERVER_NAME=achost.lsf.acme.com
```

- JAVA_HOME parameter. Optional. If Java Runtime Environment (J2RE) is already installed on this host, and you want to use it for LSF desktop support, specify the directory where it is installed. By default, `achinstall` installs JRE under *ACH_TOP*`/j2sdk1.4.1_01`. For example:

  ```
  JAVA_HOME="/usr/local/j2sdk1.4.1_01"
  ```

- APACHE_HOME parameter. Optional. If an Apache server is already installed on this host, and you want to use it for LSF desktop support, change it to use port 80 and shut it down until the installation is completed. Then, in the APACHE_HOME parameter, specify the directory where it is installed. For example:

  ```
  APACHE_HOME="/usr/local/apache"
  ```

- TOMCAT_HOME parameter. Optional. If a Tomcat server is already installed on this host, and you want to use it for LSF desktop support, change it to use port 80 and shut it down until the installation is completed. Then, in the TOMCAT_HOME parameter, specify the directory where it is installed. For example:

  ```
  TOMCAT_HOME="/usr/local/jakarta-tomcat-4.1.30-LE-jdk14"
  ```

- USER_NAME/GROUP_NAME parameter. Optional. Specify a user or a group name if you want the Web servers to run under a user or group name other than `nobody`. For example:

  ```
  USER_NAME=acweb
  ```

- DIRECTORYSERVICE parameter. Specifies whether this desktop server is also a Directory Services host. If you are implementing automatic load balancing, you must specify **`yes`** for this parameter for at least one server, and **`no`** for the remaining servers.

  If you are not implementing automatic load balancing, let this value default to **`yes`**. For example:

  ```
  DIRECTORYSERVICE=yes
  ```

- CONFIGDIRECTORY parameter. Optional. Specifies the path to where the configuration files (`SEDConfig.xml` and `MED.lst`) used by Directory Services are located. Defaults to *ACH_TOP*`/config`. For example:

  ```
  CONFIGDIRECTORY="/usr/local/config"
  ```

- AC_SERVER_LIST parameter. Optional. Specifies a list of desktop servers to which it sends desktop clients. If you are implementing automatic load balancing, and this server is a Directory Services host, specify the list of fully qualified (for example, DNS resolvable) names for all desktop servers to which this Directory Service host will refer desktop clients, and gather statistics from. Separate the names with a space. For example:

  ```
  AC_SERVER_LIST="test.acme.com test1.acme.com"
  ```

  If you are not implementing automatic load balancing, you do not need to specify a value here. The value defaults to this desktop server.

- DIGESTAUTH parameter. Specifies whether to configure digest authentication. If digest authentication is used, all requests to the desktop server

are authenticated. For details, see “Authenticating Users and Controlling User Access” on page 80. The default value of this parameter is `no`.

- ACCESS_CONTROL parameter. Optional. Specifies the permission settings of multiple directories and files. Possible values are `yes` or `no`. The default value is `no`. If ACCESS_CONTROL is set to `yes`, global read and write access is revoked, and only owners can access files and directories, which implies that `lsfadmin` may not be able to view job-related files.

4 Save the `install.config` configuration file.

5 Install the desktop server software:

Change to the LSF desktop support installation directory (where the files reside after you untar the package) and run the following as `root`:

```
cd AC_INSTALL
./achinstall -f install.config
```

If your LSF cluster enables EGO management of LSF daemons, you must run the following command to log on to EGO before runnnig `achinstall`:
`egosh user logon -u Admin -x Admin`

6 If you want to set the default as no file caching of jobs on desktop clients, then once the installation is complete, edit `/etc/med.conf`, and add the following line:

`LSB_MED_CACHEFILE_DEFAULT=No`

For additional information about `LSB_MED_CACHEFILE_DEFAULT` and cache control, see “Desktop Server Configuration Settings” on page 60.

7 Repeat the above steps for each LSF server host where you want to install the desktop server.

Enabling Desktop Support on LSF

When the desktop server is defined, you need to enable LSF desktop support on the LSF master host.

To enable LSF desktop support:

If your LSF cluster enables EGO management of LSF daemons, you must run the following command to log on to EGO before runnnig `acsetup`:
`egosh user logon -u Admin -x Admin`

1. Log on as root or the primary LSF administrator:
2. Run the following command:

   `acsetup`

3. When prompted, specify the location of LSF_ENVDIR, which is the location of `lsf.conf`.
4. When prompted, specify a list of the desktop server host names, separating the names with commas (,). The names you specify here must exactly match the names specified in `lsf.cluster`. For example, if the full DNS name is specified in `lsf.cluster`, you must specify the full DNS name here.

Changes acsetup makes

When you run `acsetup`, it does the following:

1. Ensures you are a primary LSF administrator
2. Adds the following parameters to `lsf.conf`:
   - LSB_CHUNK_NO_RLIMIT=Y
   - LSB_MBD_CONNTIMEOUT=60
   - LSF_ENABLE_EXTSCHEDULER=Y
3. Adds the following parameters to `med.conf`:
   - LSB_SBD_CONNTIMEOUT=60
   - LSB_SBD_READTIMEOUT=300
4. Adds the parameter MAX_JOB_NUM to `lsb.params`.
5. Adds a new host type for LSF desktop support called `AC` to `lsf.shared`.
6. Adds a new resource called `activecluster` to `lsf.shared`.
7. Updates LSF host configuration in `lsf.cluster.`*`clustername`*.
8. Creates a host limit for LSF desktop support in `lsb.resources`. This specifies the number of active jobs that can run on each desktop server. By default, this is set to fifty. For example: `SLOTS=50`. This should be set to the maximum number of desktop clients that will be connecting to a particular desktop server.
9. Creates the default LSF desktop support queue AC_QUEUE in `lsb.queues`.

   Review the settings of the following parameters in AC_QUEUE:

   - `RUNLIMIT` is the expected application run time. For example, if the average run time is 30 minutes and should never exceed 180 minutes, it is set to:

     `RUNLIMIT=30 180`

     If only one value is specified, then it is the maximum run time.

Note: The value of RUNLIMIT in the queue is scaled based on the CPU factor of the execution host. In Platform LSF desktop support, it is scaled by the speed of the desktop server rather than by the speed of the desktop client. Therefore, in the queue, you should set DEFAULT_HOST_SPEC=pc_host_model. For additional information refer to the *Platform LSF Reference*.

- CHUNK_JOB_SIZE is the number of jobs queued on the desktop server from the LSF master for each job slot. Therefore, if the expected run time of jobs is low, the value of CHUNK_JOB_SIZE should be high; if the expected run time of jobs is high, the value of CHUNK_JOB_SIZE should be low.

  The total number of jobs that can be queued on the desktop server from the LSF master is SLOTS (defined in lsb.resources) * CHUNK_JOB_SIZE (defined in lsb.queues).

10 Reconfigures LIM.

11 Restarts mbatchd.

Excluding Desktop Servers from LSF Queues

If LSF users in your organization typically specify that their LSF jobs can run on any available LSF host (specifying **-R "type==any"** in a `bsub` command), you may need to configure your LSF queues to exclude the desktop servers from the list of eligible hosts, to avoid running standard LSF jobs on an LSF desktop support host.

To exclude desktop servers:

1. Edit `lsb.queues` for this cluster.
2. In the HOSTS parameter for each LSF queue, exclude the desktop servers. For example:

   HOSTS=~AChost1 ~AChost2

3. Reconfigure the cluster by running **badmin reconfig**.

Checking Web Server Availability

The Apache and Tomcat Web servers should have been started by `achinstall`. You can verify that they are running before you proceed.

To verify Apache Web server is running:

- Open a Web browser at: `http://host_name`

If this is unsuccessful, check *ACH_TOP*`/apache/logs` for details.

To verify Tomcat Web server is running:

1. Open a Web browser at: `http://host_name`
2. Click both links.

If either or both links open unsuccessfully, check *ACH_TOP*`/jakarta-tomcat-4.1.30-LE-jdk14/logs` for details.

Enabling Caching on the Desktop Client

Overview

The desktop client transparently uses Microsoft Explorer's caching mechanism to manage the cache. It uses the Microsoft Windows Internet Explorer's cache directory to upload and download files. In Windows 2000, Windows 2003 Server, and Windows XP, the desktop client runs as the LocalSystem account.

- If you do not plan to enable caching on the desktop client, skip this section and proceed to "Installing the Desktop Client" on page 33. If you want to enable desktop client caching after installation, you can always make these changes later.
- If you want to enable caching on the desktop client, follow the instructions below.

To enable file caching on the desktop client:

1. Specify `Write` permission for the `Everyone` account running on all desktop clients on the cache folder. (The default permissions are `Read & Execute`, `List Folder Contents`, and `Read`.)

   Tip: The location of the cache is defined by the following registry key: `HKEY_USERS\.DEFAULT\Software\Microsoft\Windows\CurrentVersion\Explorer\User Shell Folders\Cache`.

2. Set the caching policy. It is defined by the following registry key:

   `HKEY_USERS\.DEFAULT\Software\Microsoft\Windows\CurrentVersion\InternetSettings\SyncMode5.`

   Note: If `SyncMode5` does not exist in `HKEY_USERS\.DEFAULT\Software\Microsoft\Windows\CurrentVersion\InternetSettings`, manually add it as a REG_DWORD registry key.

   Set the REG_DWORD value to one of the following:

Value	Option in Internet Explorer	Description
0	Never	Never checks whether a newer version of a page exists since it was last accessed. Always obtains a page from the cache.
2	Every time you start Internet Explorer	Checks whether a newer version of a page exists only if the page was last accessed during a previous session of Internet Explorer or on a previous day. Otherwise, obtains a page from the cache.
3	Every visit to the page	Every visit to a page, the browser checks for a newer version of a stored page. Never obtains a page from the cache.
4	Automatically	Checks whether a newer version of a page exists only if the page was last accessed during a previous session of Internet Explorer or on a previous day. Over time, if some images on the page are changing infrequently, Internet Explorer checks for new images even less frequently. Otherwise, obtains a page from the cache.

   Tip: You should set SyncMode5 to 3, which means that every visit to the page, the browser checks for a newer version.

3. Restart the desktop client so that the change in settings will take effect.

Installing the Desktop Client

The desktop client is the component that resides on desktop client computers distributed throughout your installation. These are the computers where the jobs are run.

Overview

All desktop client users run the installation on their machines, and users are not prompted for any installation options. Installation options are read from a file containing the configuration values for their desktop clients. This method requires that you edit a setup file prior to the installation. The desktop client user does not need to specify any values during the installation. This method enables you change the default configurations for large numbers of desktop clients.

If you have a firewall

If you are using a firewall, the desktop client service `sed.exe` must be granted privileges to access the Internet.

If you have auto-locking enabled on your firewall, you must allow `sed.exe` to access the Internet at all times. Otherwise, the desktop client will not work. Some firewalls have features that disable Internet access after some idle time or when a screen saver runs. Check your firewall.

To install the desktop client:

1. Download the file `sedsetup.cmd` from `ftp.platform.com`. Place it in a central location.
2. Edit `sedsetup.cmd` and define the following parameters:

Parameter	Optional/ Required	Description
`INSTALLDIR`	Optional	If you want to change the default directory for the program files on the desktop client, specify the appropriate directory. The default is `Program Files\Platform Computing\LSF desktop support`. For example: `INSTALLDIR=Program Files\Platform Computing\ LSF_AC` If you have machines with different locations for the `\Program Files` folder, for example `C:\` or `D:\`, you can comment out this parameter by placing a `rem` before `set INSTALLDIR`, and the setup program automatically detects the correct directory. If you comment out this directory, you also need to remove the following `INSTALLDIR=\%INSTALLDIR%\"` (including the space after the quote character (`"`)) from the line at the end of `sedsetup.cmd`.
`MEDURL`.	Required	Specifies a default URL where the desktop client can go to request jobs if the Directory Services host is not available. For example: `MEDURL=http://`*`host_name`*`/servlet/SedSoap`
`MEDDSURL`	Required	Specifies the URL for the Directory Services. This is the URL where the desktop client gets its configuration. If directory services is used, this should point to the directory services location. For example: `MEDDSURL=http://`*`host_name`*`/servlet/DynamicSEDConfig`
`ENABLEDISKQUOTA`	Optional	For Windows 2000 with NTFS only. For more information, see “Windows 2000 with NTFS” on page 36.
`SEDUSERDISKQUOTA`	Optional	For Windows 2000 with NTFS only. For more information, see “Windows 2000 with NTFS” on page 36.

Parameter	Optional/ Required	Description
SEDUSERDISKQUOTARATIO	Optional	For Windows 2000 with NTFS only. For more information, see "Windows 2000 with NTFS" on page 36.
SEDPOLLINTERVAL	Optional	If you want to change the amount of time between requests for work from an idle desktop client, specify the time the desktop client waits before requesting another job. Use this value to reduce network congestion at the desktop server. Specify the time in seconds. The default is 300 seconds, which is 5 minutes.For example: SEDPOLLINTERVAL=900
SCREENSAVERMODE	Obsolete	This parameter is obsolete and has been replaced by SEDMode. The following information is included for backward compatibility only. ◆ Setting this to Off is equivalent to setting SEDMode to Always. ◆ Setting this to On is equivalent to setting SEDMode to Screensaver. For additional information, see "SEDMode" on page 35. ◆ If SCREENSAVERMODE is not defined, then the value of SEDMode is used. ◆ If SCREENSAVERMODE is defined, then the value of SEDMode is ignored and a warning is logged in the SED log file. Recommended value: Set to Off so that jobs can always run on the desktop.
SEDSYSTRAYINSTALL	Optional	Specify whether to install the desktop client system tray during installation. Specify **Yes** or **No**. Any character other than Yes or No is interpreted as No. The default is Yes. For example: SEDSYSTRAYINSTALL=Yes Recommended value: Set to No so that the desktop client system tray is not installed.
SEDSYSTRAYINSTSTART	Optional	Applies only when SEDSYSTRAYINSTALL is set to Yes. Specify whether to start the desktop client system tray after installation. Specify **Yes** or **No**. Any character other than Yes or No is interpreted as No. The default is Yes. For example: SEDSYSTRAYINSTALL=Yes Recommended value: Set to No, since the desktop system tray is not installed.
SEDSYSTRAYSHORTCUT	Optional	Applies only when SEDSYSTRAYINSTALL is set to Yes. Specify whether to add a system tray shortcut to the program menu. Specify **Yes** or **No**. Any character other than Yes or No is interpreted as No. The default is Yes. For example: SEDSYSTRAYSHORTCUT=Yes Recommended value: Set to No, since the desktop system tray is not installed.
SEDUSERNAME	Optional	Specifies the user name for the desktop client user. This can be a local user name or an existing domain user name, in the format domain\user. The default user name is seduser.
SEDUSERCRED	Optional	Specifies the user credential for the desktop client user. To obtain the user credential, run the SEDPassword command, as described in "Generating user credentials" on page 36. The credential is used to create the password for a new desktop client user, and to authenticate the password of an existing user. The installer does not check the validity of the password.
SEDUSERLOCALGROUP	Optional	Specify the local group the new user is to join. This applies only when the user does not currently exist. For an existing user ID, this parameter is ignored.

Parameter	Optional/ Required	Description
SEDDisableCache	Optional	Indicates whether the desktop client is allowed to write to the cache. Valid values are yes, true, false, and no (the default), case insensitive. ◆ yes or true indicates that the desktop client does not write to the cache, however it still retrieves valid files from the cache. ◆ no or *false* indicates that the desktop client can write to the cache. However, when submitting a job, the user can disable writing to the cache for a specific file. For additional information, refer to the *Platform LSF Desktop Support User's Guide*. Note that the LSB_MED_CACHEFILE_DEFAULT parameter in med.conf prevents all desktop clients connecting to a specific desktop server from caching files. For additional information, see "Desktop Server Configuration Settings" on page 60.
MEDUSERNAME	Optional	Indicates the user name for authentication to the desktop server.
MEDUSERCRED	Optional	Indicates the user credential for authentication to the desktop server. To obtain the user credential, run the SEDPassword command. The installer does not check the validity of the password.
SEDModeSelectableByUser	Optional	Indicates whether the desktop client mode is configurable by the desktop client user. Valid values are yes or no. Otherwise, the default value is used. Default value is yes (case insensitive). For additional information, see "Setting the conditions for a desktop client requesting work" on page 46.
SEDMode	Optional	Indicates the desktop client mode. Valid values are Screensaver (the default), Always, Idle, and Logon (case insensitive). Otherwise, the default value is used. The default value is Screensaver. For additional information, see "Setting the conditions for a desktop client requesting work" on page 46.
SEDConsoleIdle	Optional	If the desktop client is running in idle mode, this parameter specifies the period of keyboard or mouse inactivity in seconds before the desktop client can request work. The default is 900 seconds (15 minutes). Any integer above 300 seconds is a valid value. Otherwise, the default value is used, and a warning is logged in the SED log file.
SEDSuspendOnBattery	Optional	Indicates whether jobs can run if the desktop client is running on a machine using battery power (e.g. a laptop computer). If this parameter is set to Yes, then if the desktop client switches to battery power: ◆ Any running job is terminated and the desktop server is notified of the termination ◆ The desktop client cannot start running jobs

Windows 2000 with NTFS

The following parameters are optional and applicable only to Windows 2000 with NTFS. If you want to limit the amount of disk space the desktop client can use while running work, set the following parameters as described below:

Parameter	Description
ENABLEDISKQUOTA	Enable disk quota by ensuring the value is set to **Yes**. This specifies to use the values for SEDUSERDISKQUOTA and/or SEDUSERDISKQUOTARATIO. If you specify a value for one of the parameters, that value is used. If you specify a value for both parameters, the value that is greater is used. If you do not specify a value for either parameter, the defaults apply, and the value that is the greater of the two is used. For example: ENABLEDISKQUOTA=Yes S Recommended value: Set to No so that disk quota, specified by either SEDUSERDISKQUOTA or SEDUSERDISKQUOTARATIO, described below, is not used.
SEDUSERDISKQUOTA	Set the maximum number of megabytes the desktop client can use by specifying the number in MB. The default is 100 MB. For example: SEDUSERDISKQUOTA=500
SEDUSERDISKQUOTARATIO	Set the maximum percentage free disk space the desktop client can use by specifying a number from 1 to 100. For example, 50% would be specified as 50. The default is 33%. For example: SEDUSERDISKQUOTARATIO=50

Note: If both the SEDUSERDISKQUOTA and SEDUSERDISKQUOTARATIO parameters are defined, the parameter with the greater value is used.

3 Save sedsetup.cmd.

4 Make sure that sedsetup.cmd resides in the same directory containing lsf7.0_desktop.msi, from which the users install LSF desktop support. If you changed the name or location of lsf7.0_desktop.msi, you can modify the SETUPMSI variable in sedsetup.cmd to reflect the change.

5 Instruct the users to run sedsetup.cmd to install the desktop client software.

Generating user credentials

Secure passwords

LSF desktop support increases security with HP ProtectTools Authentication Services. Therefore, if you are using a secure password when you run SEDPassword.exe, you must prefix it with <SE>\.

For example, if the password is my_password, you run:

```
SEDPassword "<SE>\my_password"
```

To generate user credentials:

1 Locate the file SEDPassword.exe.

2 Run SEDPassword.exe as follows:

```
SEDPassword.exe password
```

where *password* is the correct password for the user ID you specify for the SEDUSERNAME parameter. SEDPassword.exe generates an authorization string you will use in the SEDUSERCRED parameter for the silent installation of the clients. For example:

```
C:\lsf_desktop7.0>SEDPassword.exe barney
Encrypted length: 8
Encrypted data:
0xf8, 0x7b, 0xd7, 0x18, 0x29, 0xf4, 0xf6, 0x7d
```

3 Copy the text in the Encrypted data and paste it into the SEDUSERCRED parameter as follows:

SEDUSERCRED=`0xf8, 0x7b, 0xd7, 0x18, 0x29, 0xf4, 0xf6, 0x7d`

Setting up screen-saver mode

To run the desktop client in screen-saver mode, you can use any screen saver. However, the desktop client must have a screen saver enabled. In a corporate network environment, an IT administrator can often enable screen savers on all desktop clients. However, in an ISP environment, the desktop client user may need to enable the screen saver.

If you specified `Screensaver` in `sedsetup.cmd` during installation, then the desktop client mode is already set to `Screensaver` and the desktop client screen saver is set as the default screen saver for the user who installs the desktop client.

When screen-saver mode is enabled on the desktop client, the user can choose which mode to run in by selecting an option in the **Details** dialog box.

If you do not enable screen-saver mode during the silent installation of the desktop client, a user can later enable a screen saver and select screen-saver mode as required.

To change the system tray icon:

1 Create five bitmap files with the following attributes:

- Each file is saved in `.BMP` format.
- The image in each file uses the **256 Colors** color setting.
- The image in each file is 138 pixels by 138 pixels.
- The files should have the following names (each name describes the circumstance in which it will be used):

This file	Is used when...
`sed_idle.bmp`	The desktop client is not currently running a job for LSF desktop support.
`sed_complete.bmp`	The job that was running on the desktop client has completed.
`sed_running.bmp`	A job is currently running on the desktop client.
`sed_disabled.bmp`	LSF desktop support is disabled
`sed_error.bmp`	There is an error.

2 Place all files in the LSF desktop support directory on each desktop client, for example: `C:\Program Files\Platform Computing\LSF desktop support`.

Note: The customized system tray icon loads only if all five bitmap files exist and meet the requirements described above.

To replace these bitmap files on multiple desktop clients, either:

- Send the new bitmap files to every desktop client with instructions about where to save them
- Place the new bitmap files in a central shared folder that can be accessed by all desktop client users, and instruct them to copy over the files

Install LSF Desktop Reports

At least one LSF desktop server (MED) host is installed correctly.

Contents

Top-level directories

The reporting feature resides in the `perf` directory, which is a subdirectory of the top-level LSF installation directory. This document uses LSF_TOP to refer to the top-level LSF directory and `LSF_TOP/perf` to describe the `perf` directory.

Download and deploy the LSF desktop reports package

1. Obtain the LSF desktop reports package from Platform Computing.

   The LSF desktop reports package is only available for Linux hosts.

2. Copy the LSF desktop reports package into a temporary directory.

   For example, copy the package into your home directory.

   ```
   cp lsf_desktop_7Update2_reports.tar ~
   ```

3. Extract the package to the temporary directory. For example:

   ```
   cd ~
   tar xvf lsf_desktop_7Update2_reports.tar
   ```

   The LSF desktop reports content is installed in the `ego` subdirectory under your temporary directory.

4. Copy the `ego` subdirectory to the top level EGO directory. For example:

   ```
   cd ../EGO_TOP
   cp -r ~/ego .
   ```

   The installation is successful if the following file exists: `LSF_TOP/perf/ego/version/lib/perf_lsfdesktop_loader.jar`.

5. Create the LSF desktop data schema for your database.

   For example, for the Derby database, run the following commands to create the schema:

   ```
   ij.sh
   ij> connect 'jdbc:derby://database_host:port_number/app';
   ij> run
   '$LSF_TOP/perf/ego/version/DBschema/Derby/lsfdesktopdata.sql';
   ij> exit;
   ```

   where

   - *database_host* is your database host
   - *port_number* is the port number used to connect to your database host

   If you are using a production database, create the database schema as described in the *Administering Platform LSF*.

Configure and Test LSF Desktop Reports on your Cluster

Configure a unique loader controller (`plc`) service for every LSF desktop server (MED) host, depending on where your MED hosts are installed.

Contents

Configure the loader controller for MED hosts in an NFS or a shared directory

For MED hosts in an NFS or a shared directory, you do not need to create a new loader controller. Configure the original loader controller to collect the MED data.

1. In the command console, open the EGO service directory (EGO_ESRVDIR).

   cd LSF_CONFDIR/ego/*cluster_name*/eservice/esc/conf/services

2. Edit `plc_service.xml`.
3. Add the AC_TOP environment variable to specify the path to the MED host Tomcat server.

   For example, if MED_TOP is the top-level directory where the LSF desktop server is installed, add the following line:

```
<ego:EnvironmentVariable name="AC_TOP">MED_TOP/jakarta-tomcat-4.1.30-LE-jdk14</ego:EnvironmentVariable>
```

4. In the command console, stop all services and restart EGO on the master host to activate these changes.

   egosh service stop all
   egosh ego restart *master_host_name*

Create new loader controllers for local MED hosts

For local MED hosts, you need to configure a local loader controller on each local MED host. Perform the following steps on each local MED host to configure a local loader controller.

1. Obtain the name of the local MED host.

   Run the `hostname` command while logged in to the local MED host.

2. Configure a new loader controller as described in "Configure the new loader controller" on page 41.
3. In the command console, open the PERF configuration directory.

   cd *LSF_TOP*/perf/conf

4. Create a new directory with the same name as the one you specified in the new loader controller configuration file.

   For example, if the local host is `hostMED1`,

   mkdir plc_hostMED1

5 Copy the contents from the old loader controller directory to the new controller. For example:

```
cp plc/* plc_hostMED1
```

6 Remove the redundant EGO and LSF configuration files from the new loader controller directory, and remove the redundant LSF desktop configuration files from the old loader controller directory.

For example, delete `plc_ego.xml` and `plc_lsf.xml` from the `plc_hostMED1` directory, and delete `plc_lsfdesktop.xml` from the `plc` directory.

```
rm plc_hostMED1/plc_ego.xml
rm plc_hostMED1/plc_lsf.xml
rm plc/plc_lsfdesktop.xml
```

7 Create a new loader controller service as described in "Create new loader controllers for local MED hosts" on page 40.

8 In the command console, stop all services and restart EGO on the master host to activate these changes.

```
egosh service stop all
egosh ego restart master_host_name
```

Configure the new loader controller

1 In the command console, open the EGO service directory.

```
cd LSF_TOP/perf/conf
```

2 Copy the current `plc.xml` configuration file to a new configuration file with the name of the local MED host. For example

```
cp plc.xml plc_hostMED1.xml
```

3 Edit the new configuration file.

For example, edit `plc_hostMED1.xml`.

Change the port number to a new, unique port number that is not being used by another process.

Edit the `<Port>` tag to use a new number.

Tip: Run netstat -an to see if the new port is already being used by an existing process..

4 Specify a new loader controller configuration directory.

Edit the `<PLCDir>` tag to change the default directory from `plc` to a new loader controller directory with the name of the local MED host. For example

```
<PLCDir>plc_hostMED1</PLCDir>
```

Create a new loader controller service

1 In the command console, open the EGO service directory.

```
cd LSF_CONFDIR/ego/cluster_name/eservice/esc/conf/services
```

2 Copy the current `plc_service.xml` file to a new service configuration file with the name of the local MED host. For example:

```
cp plc_service.xml plc_hostMED1_service.xml
```

3 Edit the new service configuration file.

For example, edit `plc_hostMED1_service.xml`.

4 Change the service name to that of the new loader controller service.

Navigate to the `<sc:ServiceDefinition>` tag and edit the `ServiceName` paramater. For example:

```
<sc:ServiceDefinition ... ServiceName="plc_hostMED1" ...>
```

5 Edit the command that runs the `plc` script using the old loader controller configuration file to use the new loader controller configuration file:

Navigate to a line resembling the following and change `plc.xml` to the name of the new loader controller configuration file.

```
<ego:Command>.../perf/etc/plc.sh -f plc.xml ...</ego:Command>
```

For example:

```
<ego:Command>.../perf/etc/plc.sh -f plc_hostMED1.xml ...</ego:Command>
```

6 Add the AC_TOP environment variable to specify the path to the MED host Tomcat server.

For example, if *MED_TOP* is the top-level directory where the LSF desktop server is installed, add the following line:

```
<ego:EnvironmentVariable name="AC_TOP">MED_TOP/jakarta-tomcat-4.1.30-LE-
jdk14</ego:EnvironmentVariable>
```

7 Change the `JobController` command that runs the `plc` script using the old loader controller configuration file to use the new loader controller configuration file.

Navigate to a line resembling the following and change `plc.xml` to the name of the new loader controller configuration file:

```
<ego:JobController>.../perf/etc/plc.sh -k -f plc.xml ...</ego:JobController>
```

For example:

```
<ego:JobController>.../perf/etc/plc.sh -k -f plc_hostMED1.xml
...</ego:JobController>
```

8 Change the resource requirement to specify the MED host as an execution host.

Navigate to the `<ego:ResourceRequirement>` tag and specify the MED host. For example:

```
<ego:ResourceRequirement>select('hostMED1')</ego:ResourceRequirement>
```

9 Change the consumer ID from a management host ID to an execution host ID.

Navigate to the `<ego:ConsumerID>` tag and change the consumer tree to point to applications. For example:

```
<ego:ConsumerID>/SampleApplications/EclipseSamples</ego:ConsumerID>
```

10 Optional. Change the resource group name from a management host to an execution host.

Navigate to the `<ego:ResourceGroupName>` tag and change the name to `ComputeHosts`.

If the MED host is an EGO management host, you can skip this step.

CHAPTER 4

Configuring the Components in LSF Desktop Support

Once LSF desktop support is installed, it is designed to run out of the box. You need to specify very few parameters to begin using the LSF desktop support. However, you may need to periodically update configuration parameters for the desktop clients.

Contents

Configuring Desktop Clients

When you want to change the configuration of all of the desktop clients, you can edit the desktop client configuration file, `SEDConfig.xml`.

`SEDConfig.xml` is located on the directory service server in the directory specified by the `CONFIGDIRECTORY` value in `install.config`. You can use an XML editor or a text editor such as vi to modify this file.

Each desktop client in the LSF desktop support environment reports to the desktop server on startup and once every 24 hours, checking for an updated configuration. If the desktop client detects a new version of the configuration, it downloads the new configuration file and applies it.

You can configure the following options for desktop clients:

Example: SEDConfig.xml

```
<?xml version="1.0"?>
<SEDConfig>
<MEDDSURL>http://dirservhost.platform.com/servlet/DynamicSEDConfig</
  MEDDSURL>
<MEDURL>http://dirservhost.platform.com/servlet/SedSoap</MEDURL>
<SEDFullyKillNTJob>yes</SEDFullyKillNTJob>
<SEDPollInterval>300</SEDPollInterval>
<SEDLatestVersion>41</SEDLatestVersion>
<SEDLatestVersionURL>http://dirservhost.platform.com/SED/sedupd.exe
  </SEDLatestVersionURL>
</SEDConfig>
```

Changing the time between work requests from an idle desktop client

Edit the `SEDPollInterval` parameter in `SEDConfig.xml`:

```
<SEDPollInterval>300</SEDPollInterval>
```

To change the amount of time between requests for work from an idle desktop client, specify the time the desktop client waits before requesting another job or reporting status in seconds. The default is 300 seconds, which is five minutes.

Setting the time period for which a desktop client pauses

Edit the `SEDOptOutInterval` parameter in `SEDConfig.xml` to specify the number of seconds for which the desktop client pauses. At the end of this time period, the desktop client starts processing again.

Specify the time period as follows:

`<SEDOptOutInterval>`*`ssssss`*`</SEDOptOutInterval>`

where *`ssssss`* is the number of seconds. For example, to pause the desktop client for 2 hours, specify 7200 seconds.

Updating desktop client applications with a new version

Note: The `sedupd.exe` file is included with patch kits.

Edit the following parameters in `SEDConfig.xml`:

- Use the `SEDLatestVersionURL` parameter to specify the host name in the URL where the desktop client will go to get a new version of its binary, during an automatic upgrade of the desktop client. The URL can be any URL that supports the http GET method. Specify the URL as follows:

  `<SEDLatestVersionURL>http://dirservhost.platform.com/SED/sedupd.exe</SEDLatestVersionURL>`
- Use the `SEDLatestVersion` parameter to specify the exact version number of the new desktop client software. This tells the desktop client to look for that new version of its binary. Specify the value as follows:

  `<SEDLatestVersion>41</SEDLatestVersion>`

  To find your current version number, see `sed.log`.

Changing the Directory Services host

Edit the MEDDSURL parameter in `SEDConfig.xml` to specify the host name in the URL pointing to the updated configuration file. Specify the URL as follows:

`<MEDDSURL>http://dirservhost.platform.com/servlet/DynamicSEDConfig</MEDDSURL>`

Do not change other values in the URL other than the host name (dirservhost.platform.com).

All desktop server host names must be included in the `MED.lst` file. The location of this file is specified by the CONFIGDIRECTORY parameter in `install.config`. If CONFIGDIRECTORY is not defined, the default location is `$ACH_TOP/config`.

The dynamic Web page reads the `MED.lst` file and returns the name of an desktop server in the MEDURL field of the XML file. The desktop client then uses this desktop server until the next reconfiguration.

Ensuring full cleanup on an Windows desktop client

When a job completes on an Windows desktop client, some child processes associated with the job may still be running. You can enable an option to kill all processes associated with `seduser` (or the user account that runs the jobs) when a job completes, using the `SEDFullyKillNTJob` option. By default, this option is not enabled.

Edit the `SEDFullyKillNTJob` parameter in `SEDConfig.xml`. Change the value from **`no`** to **`yes`** as follows:

```
<SEDFullyKillNTJob>yes</SEDFullyKillNTJob>
```

Warning: Make sure that the user account under which LSF desktop support jobs run on Windows is not a real end-user's account—the user's processes could also be killed when the Windows job completes.

Setting the conditions for a desktop client requesting work

Specify the circumstances under which a desktop client requests work

Set the `SEDMode` parameter in `SEDConfig.xml` to one of the following values:

- Always: Jobs can always run, even if an interactive user is signed on to the desktop client.
- Logon: Jobs can run only when no user is logged on to the desktop client. A user is logged on during an interactive session that is initiated from a Windows logon dialog box. If a job is running when an interactive user logs on, the job is terminated and the desktop server is notified of the termination. The desktop server then redispatches the job.
- Screensaver: Jobs can run only when a screen saver is active. If a job is running when the screen saver becomes inactive, then the job is suspended.

  When the screen saver becomes active:

  - If the job has checkpointing capabilities, it restarts from where it left off (all data files and temporary files remain on the desktop client).
  - Otherwise, the job starts running again from the beginning.
- Idle: Jobs can run only when the desktop client has been idle for a configurable period, determined by `SEDConsoleIdle`, as described in "Specify the period of inactivity before the desktop client can request work" on page 47. If a job is running when the desktop client stops being idle, then the job is suspended.

  When the desktop client becomes idle:

  - If the job has checkpointing capabilities, it restarts from where it left off (all data files and temporary files remain on the desktop client).
  - Otherwise, the job starts running again from the beginning.

If no valid value is specified, then the default value of Screensaver is used.

For example: `<SEDMode>Logon</SEDMode>`

Determine whether the desktop client mode is configurable by individual desktop client users

Use the `SEDModeSelectableByUser` parameter in `SEDConfig.xml` to determine whether the desktop client mode (defined by the value of the `SEDMode` parameter, described above,) is configurable by individual desktop client users.

- No: The value of `SEDMode` in `SEDConfig.xml` is used by all desktop clients.
- Yes: The value of `SEDMode` in `SEDConfig.xml` is ignored. This uses the value specified by the user in the desktop client configuration.

Valid values are `yes` or `no`. Otherwise, the default value is used. Default value is `yes` (case insensitive).

Specify the period of inactivity before the desktop client can request work

Use the `SEDConsoleIdle` parameter in `SEDConfig.xml` to specify the period of keyboard or mouse inactivity in seconds before the desktop client can request work. This is relevant only if the `SEDMode` parameter, described above, is specified as `Idle`.

The default is `900` seconds (15 minutes). Any integer above 300 seconds is a valid value. Otherwise, the default value is used, and a warning is logged in the SED log file.

Changes to the above configuration parameters take effect only when the desktop client next checks the desktop server for configuration changes. By default, this check occurs every 24 hours. You can use the `SEDUpdateCheckInterval` parameter in `SEDConfig.xml` to configure this interval, as described below, in "Specifying the interval to check for configuration changes" on page 47.

Specifying the interval to check for configuration changes

Edit the `SEDUpdateCheckInterval` parameter in `SEDConfig.xml` to specify, in hours, the interval at which the desktop client checks the desktop server for configuration changes. The default value is 24 (hours).

Note that the desktop client only checks for configuration changes if it is idle, i.e. if it is not currently running a job. If the desktop client is not idle when the check is scheduled to occur, it occurs the next time the desktop client becomes idle.

Setting conditions for jobs running on battery power

Edit the `SEDSuspendOnBattery` parameter in `SEDConfig.xml` to indicate whether jobs can run if the desktop client is running on a machine using battery power (e.g. a laptop computer). If this parameter is set to Yes, then if the desktop client switches to battery power:

- Any running job is terminated and the desktop server is notified of the termination
- The desktop client cannot start running jobs

Specifying whether desktop clients can write to the cache

Edit the `SEDDisableCache` parameter in `SEDConfig.xml` to indicate whether the desktop client can write to the cache. Valid values are **`yes`**, **`true`**, **`false`**, and **`no`** (the default), case insensitive.

- `yes` or `true` indicates that the desktop client does not write to the cache, however it still retrieves valid files from the cache.
- `no` or `false` indicates that the desktop client writes to the cache. However, when submitting a job, the user can disable writing to the cache for a specific file. For additional information, refer to the *Platform LSF Desktop Support User's Guide*.

Note: The `LSB_MED_CACHEFILE_DEFAULT` parameter in `med.conf` specifies the default file caching setting for all desktop clients connecting to a specific desktop server. For additional information, see "Desktop Server Configuration Settings" on page 60 and refer to the *Platform LSF Desktop Support User's Guide*.

Enabling file caching on the desktop client

The desktop client transparently uses Microsoft Internet Explorer's cache to upload and download files. In Windows 2000, Windows 2003 Server, and Windows XP, the desktop client runs as the LocalSystem account. Therefore, for each desktop client user, you need to specify:

- Write permissions
- Caching policy, i.e. under which circumstances, if any, LSF desktop support obtains files from the cache
- The size of the cache folder

Note: Changing any of these settings requires restarting the desktop client to take effect.

For detailed instructions on changing these settings, see "Enabling Caching on the Desktop Client" on page 32.

Configuring EGO Management of Platform LSF Desktop Support Services

You can configure EGO management of the master execution daemon (MED) and the Web server components (Tomcat and Apache) so that they run as EGO services. This enables automatic startup of these services, which improves availability.

IMPORTANT EGO must be enabled in the LSF cluster to configure EGO management of LSF desktop support services. See *Administering Platform LSF* for information about enabling EGO for LSF.

About EGO management of LSF desktop support services

With EGO management of LSF desktop support services enabled, you can

- Enable automatic restart of the MED, Tomcat, and Apache services
- Use EGO commands to view the status of EGO services, including LSF desktop support services
- Use EGO commands to start LSF desktop support services

Configuring EGO management of LSF desktop support services provides additional high availability in cases where the MED or Web services fail during job execution. EGO can restart the services, which prevents rerunnable jobs from wasting valuable CPU time by rerunning unnecessarily.

In the following examples, the term MED/WS refers to the master execution daemon and web server processes running on a Platform LSF desktop support server.

Without EGO management of LSF desktop support services (feature not enabled): Web service

Stage	Process	Action	Result
1	mbatchd	Dispatches a job to MED/WS on hostA	Job waits on MED/WS on hostA
2	SED	Requests a job from the MED/WS on hostA	WS on hostA sends a job to the SED
3	SED	Initiates the job to run on the client host	Client host successfully runs the job
4	Tomcat or Apache or both	Crash on hostA while the job is running	Jobs cannot be redispatched, because mbatchd sees that MED is running.
5	SED	Tries to upload job results to the WS	Upload of job results fails
6	SED	Tries again to upload job results a maximum of 5 times	CPU time to run job is wasted and there is a performance cost due to the SED querying the Directory Service

With EGO management of LSF desktop support services enabled: Web service

Stage	Process	Action	Result
1 -3		Same as with EGO management of LSF desktop support services not enabled	
4	Tomcat or Apache or both	Crash on hostA while the job is running	Jobs cannot be redispatched, because mbatchd sees that MED is running.
5	EGO	Automatically restarts failed service	SED can upload job results
6	SED	Uploads job results to the WS	CPU time is not wasted because the job ran only once

Without EGO management of LSF desktop support services (feature not enabled): MED

Stage	Process	Action	Result
1	mbatchd	Dispatches a job to MED/WS on hostA	Job waits on MED/WS on hostA
2	SED	Requests a job from the MED/WS on hostA	WS on hostA sends a job to the SED
3	SED	Initiates the job to run on the client host	Client host successfully runs the job
4	MED	Crashes on hostA while the job is running	mbatchd redispatches the job
5	SED	Uploads job results to the WS that is still running on hostA	Job results are lost when the job is redispatched, because the job reruns on a new host
6	SED	Tries again to upload job results. Job fails after 5 retries.	CPU time to run job is wasted because the job was rerun even though it already completed successfully

With EGO management of LSF desktop support services enabled: MED

Stage	Process	Action	Result
1 -3		Same as with EGO management of LSF desktop support services not enabled	
4	MED	Crashes on hostA while the job is running	EGO restarts MED, mbatchd does not need to redispatch the job
5	SED	Uploads job results to the WS on hostA	CPU time is not wasted because the job ran only once

Scope and Limitations

EGO management of LSF desktop support services applies to the MED and to the Web servers (Tomcat and Apache). With EGO management of LSF desktop support services enabled, you should not use the command lsfac_daemons to start or stop Apache or Tomcat services. Instead, you should use the egosh command to start and stop these services.

Configuration to enable EGO management of LSF desktop support services

Enable EGO management of the MED

Configuration file	Parameter and syntax	Behavior
lsf.conf	LSF_EGO_DAEMON_CONTROL=Y	◆ Enables EGO Service Controller to control LSF res, sbatchd, and MED startup. Set the value to "Y" at installation if you want EGO Service Controller to start services, and restart services if they fail.

Enable EGO management of LSF desktop support Web services

During installation, the files LSFDesktopApache.xml and LSFDesktopTomcat.xml are created in the directory $AC_TOP/config, using the correct variable values. To enable EGO management of LSF desktop support Web services, you must copy these files and increase the number of available slots to accommodate the two additional services (Apache and Tomcat) managed by the EGO Service Controller.

1 Copy LSFDesktopApache.xml and LSFDesktopTomcat.xml to *EGO_ESRVDIR*/esc/conf/service

2 Increase the availableSlots in the `InternalResourceGroup` by 2, as shown in the following example.

Configuration file	Parameter and syntax
`LSF_CONFDIR/ego/cluster_name/ResourceGroup.xml`	`<ResourceGroup ResourceGroupName="InternalResourceGroup" availableSlots="5"/>`

Important To add the new services, you must restart EGO using the command `egosh ego restart all`. See *Administering and Using Platform EGO*.

Configuring Platform LSF Desktop Support High Availability

The high availability feature provides a way to maximize CPU usage by ensuring that successfully completed rerunnable jobs run only once, even if master execution daemon (MED) and Web server (WS) processes fail during job execution. With high availability enabled, client hosts can upload job results to a new desktop server if they can no longer connect to the original desktop server.

About LSF desktop support high availability

There are three key components of the Platform LSF desktop support high availability feature:

- A directory that functions as a job status pool for every job dispatched to a slave execution daemon (SED)
- A job status file for each job, stored in the job status pool directory and removed when the job finishes or is killed
- At least one backup Directory Service to make sure that desktop clients (SEDs) can always access information about jobs

Without high availability (feature not enabled)

In the following description, the term MED/WS refers to the master execution daemon and web server processes running on a Platform LSF desktop support server.

Stage	Process	Action	Result
1	mbatchd	Dispatches a job to MED/WS on hostA	Job waits on MED/WS on hostA
2	SED	Requests a job from the MED/WS on hostA	WS on hostA sends a job to the SED
3	SED	Initiates the job to run on the client host	Client host successfully runs the job
4	MED and Tomcat or Apache	Crash on hostA while the job is running	mbatchd thinks hostA is down
5	mbatchd	Redispatches the job to another desktop sever	SED does not know the job was redispatched
6	SED	Tries to send job completion results to the WS on hostA	CPU time to run job is wasted because the job was rerun even though it already completed successfully

With high availability enabled

Stage	Process	Action	Result
1 -4		Same as with high availability not enabled	
5	mbatchd	Redispatches the job to another MED	A new desktop server owns the redispatched job
6	SED	Tries to send job results to hostA	SED cannot upload the job results to hostA
7	SED	Contacts the Directory Service and requests the identity of the new desktop sever	Directory Service looks in the job status pool directory for the information
8	Directory Service	Gets the identity of the new desktop server from the job status pool and sends this information to the SED	SED knows where to send the job results, and the job does not run again
9	SED	Uploads job results to new desktop server	CPU time is not wasted because the job ran only once

Scope and behavior

The high availability feature applies only to jobs submitted as rerunnable, and when both the MED, the lim, and one or more components of the associated WS fail. For cases where high availability does not apply, you can configure EGO to control MED, Apache, and Tomcat as services that EGO automatically restarts.

In the following description, (ok) indicates that a process is running and (-) indicates that the process failed during job execution.

High availability takes effect	MED and LIM	Apache	Tomcat	Behavior
Yes	-	-	-	Jobs can be redispatched, because mbatchd sees the desktop server (MED/WS) host as down. The SED sends the job results to the new desktop server.
	-	ok	-	
	-	-	ok	
	-	ok	ok	
No	ok	ok	-	Jobs cannot be redispatched, because mbatchd sees that the MED is running. If EGO control of LSF services is enabled, EGO can restart Apache and Tomcat automatically.
	ok	-	ok	
	ok	-	-	

Configuration to enable the high availability feature

The high availability feature is configured for desktop servers (MED) in wscache.conf and for desktop clients (SED) in SEDConfig.xml. You must configure both servers *and* clients.

Before you enable the high availability feature:

- Create a job status pool directory with the following characteristics:
  - Shared, with permissions that allow all Platform LSF desktop support servers to create/write/read/delete job status files
  - Created on a stable and high-performance host
  - Limited user account access; only the user account that runs Platform LSF desktop support services should have access
- Set up at least two Directory Service hosts
- In lsb.queues, for all LSF desktop support queues, define RERUNNABLE=yes

To enable the high availability feature, you must define the following parameters.

Configuration file	Parameter and syntax	Default	Behavior
wscache.conf	FTDIRECTORY *path_name*	Not defined	• Enables high availability for the desktop server • Specifies the absolute path to the directory that will contain the job status pool • Must be defined on every desktop server host

Configuration file	Parameter and syntax	Default	Behavior
SEDConfig.xml	<SEDJobFaultTolerance> Yes </SEDJobFaultTolerance>	Not defined	◆ Enables high availability for the desktop client ◆ If left undefined or set to No, high availability is not enabled
	<MEDBACKUPDSURL> *URL*;*URL*;*URL*… </MEDBACKUPDSURL>	Not defined	◆ Specifies the URL for the backup Directory Service(s) ◆ Not required if your Directory Services are configured as round-robin

Important To apply configuration changes to desktop servers, you must restart the Web services on all desktop server hosts. To apply changes to desktop clients, you can either restart all SEDs or let each SED automatically get the new values when the SED checks for configuration changes.

Configuration to modify the high availability feature

You can define the following optional parameter to modify the length of time a desktop server holds a job before dispatching the job to a new SED.

Configuration file	Parameter and syntax	Default	Behavior
wscache.conf	FTOVERTIME *hours*	24	◆ Specifies the amount of time in hours that the desktop server waits to receive job results from the SED ◆ The desktop server waits the specified amount of time before redispatching the job to a new SED ◆ If the original SED contacts the desktop server within the specified time, the job is not redispatched ◆ The value must be an integer greater than zero

Changing Job Completion Log Location

Each desktop server has a log file `job_completion_log` with a date stamp in the YYYY_MM_DD format in which daily job completion information is stored. For example: `job_completion_log.`*`host_name`*`.2001_12_31`

The location of the log file is specified by the JOBDIRECTORY parameter in `wscache.conf` in the TOMCAT_HOME directory of each desktop server.

Whenever a job is completed, a line is logged in the file. If no jobs are completed on a specific day, no file is created for the day. Each day has a separate log file.

Job completion log file format

```
TimeStamp|HumanReadableTime|JobID:rerun|SedID|ExitedCode|FailureCode|
  ElapsedTime|UserTime|KernelTime
```

For example:

```
1010985060497|15/01/2002 14:30:15|123[34]:0|CR185119-B_886056_1|0|0|2908|0|7531
```

- `TimeStamp`—The difference, measured in milliseconds, between the current time and midnight, January 1, 1970 UTC
- `HumanReadableTime`—Time in human readable format, dd/mm/`yyyy hh:mm:ss`
- `JobID`—ID of the job. If it is a job array, the format is JobID[index]. The number after the : indicates the number of times the job has been rerun
- `ExitedCode`, `FailureCode`—Any non-zero value indicates a problem
- `ElapsedTime`—Job run time in seconds
- `UserTime`, `KernelTime`—`UserTime+KernelTime=CPU Time` in seconds

To change the directory in which job log files are stored:

You can change the location where the log files are stored.

1. On the desktop server, locate and open the file `wscache.conf` for editing. It is located in the TOMCAT_HOME directory.
2. To change the location where the job completion log files are stored, in the JOBDIRECTORY parameter, specify the full path for the directory. For example:
   JOBDIRECTORY *ACH_TOP*/wscache
3. Save the file.

Setting Up Licensing

Make sure that you have the correct number of licenses for your desktop servers and your desktop clients.

To set up licensing:

1. Contact Platform Technical Support to get authorization keys for the following:
   - Each desktop server requires an LSF Server license
   - A feature license for the appropriate number of desktop servers
   - A feature license for the appropriate number of desktop clients
2. Follow the procedures in the *Platform LSF Administrator's Guide* to set up licenses for your desktop servers.

Configuring the Default LSF Desktop Support Queue

The default LSF desktop support queue is AC_QUEUE, which is configured during the installation using the default options. Because LSF desktop support uses a subset of the standard LSF queue parameters, do not add to or delete parameters from this queue. If required, you can change the following values:

- HOSTS
- SWAPLIMIT
- RUNLIMIT
- AC_RES_REQ
- STOP_COND
- RERUNNABLE

To configure the default queue:

1. Find the default LSF desktop support queue AC_QUEUE in `lsb.queues`.
2. HOSTS parameter. Optional. You can change the desktop server host name for this queue as follows:

   `HOSTS=`*`host_name`*

3. RUNLIMIT parameter. Optional. If you want to change the amount of time that a job in this queue can run before the job is terminated automatically, specify the following:

   `RUNLIMIT = [`*`hh`*`:`*`mm`*`] `*`maxhh`*`:`*`mm`*

   where *maxhh:mm* is the maximum run limit allowed, and *hh:mm*, if specified, is the default run limit for the queue. The default is 120 10000000, which sets the default run limit to 2 hours and the maximum to 10 million minutes.

   Note: The value of `RUNLIMIT` in the queue is scaled based on the CPU factor of the execution host. In Platform LSF desktop support, it is scaled by the speed of the desktop server rather than by the speed of the desktop client. Therefore, in the queue, you should set `DEFAULT_HOST_SPEC=pc_host_model`. For additional information refer to the *Platform LSF Reference*.

4. SWAPLIMIT parameter. Optional. If you want to change the total virtual memory that a job in this queue can use before the job is terminated automatically, specify the following:

   `SWAPLIMIT = `*`nnnn`*

   where *nnnn* is the maximum number of kilobytes. The default is 102400, or 100 MB.

5. AC_RES_REQ parameter. Optional. If you want to specify desktop client resource requirements that apply to all jobs submitted to this queue, specify a resource requirements statement using this parameter. For example:

   `AC_RES_REQ=cput==PII&&mem>128`

   The above states that all jobs submitted to this queue require a Pentium II processor with more than 128 megabytes of free physical memory. See “Built-in resource definitions” on page 62 for the possible values you can specify for resources.

6. STOP_COND parameter. Optional. If you want to stop a job and submit it to a different desktop client if the available memory on the desktop client drops beneath

a certain threshold, you can set the threshold level here using the STOP_COND parameter. The only accepted threshold is mem. For example:

```
STOP_COND=mem<50
```

7 RERUNNABLE parameter. Optional. 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.:

8 Run `badmin reconfig` to reconfigure mbatchd.

Creating an Additional LSF Desktop Support Queue

If you want to use multiple queues in your LSF desktop support, make sure that the queues you create are LSF desktop support-compatible. LSF desktop support supports a limited set of LSF queue parameters. Therefore the recommended method for creating an additional queue is to copy the queue created by the install script.

To create an additional queue:

1. Find the default LSF desktop support queue AC_QUEUE in `lsb.queues`.
2. Copy AC_QUEUE. You can edit the new queue, but do not delete or add any parameters to the queue. The default queue contains all of the valid parameters for an LSF desktop support queue.
3. QUEUE_NAME parameter. Required. Change the name of the queue as follows:

   `QUEUE_NAME = newname`

   Specify any ASCII string up to 40 characters long. You can use letters, digits, underscores (_) or dashes (-). You cannot use blank spaces, and you cannot use the reserved name default.
4. Specify the remaining parameters as described in “Configuring the Default LSF Desktop Support Queue” on page 57.
5. Run `badmin reconfig` to reconfigure mbatchd.

Desktop Server Configuration Settings

The desktop server configuration file `/etc/med.conf` contains the URLs required by the desktop server and the desktop clients for file and data transfer, and for status reporting.

When you installed the desktop server, the install script `install.config` configured the desktop server for you. A description of each of the parameters is included here to help your understanding of the configuration.

CAUTION **Do not change the values in this file. To change the values, edit the install script and reinstall the desktop server.**

About med.conf parameters

The following parameters are defined within `med.conf`:

- LSB_MED_WEB_URL parameter specifies the URL for the Tomcat Web server. This is where the host posts jobs waiting to be run. For example:

  `http://AChost/servlet/P2PServer`

  where AChost is the desktop server name.
- LSB_MED_DATA_WEB_URL parameter specifies the URL for the file transfer location, where the desktop clients obtain any files required by the jobs they run. For example:

  `http://AChost`
- LSB_MED_STAGE_DIR parameter specifies the URL for the working files that contain job information. For example:

  `/usr/local/lsfac/apache/htdocs`
- LSB_MED_TOP_DIR parameter specifies the directory name where the desktop server is installed. For example:

  `/usr/local/lsfac`
- LSB_MED_SHARED_FILESYS parameter indicates whether a shared file system exists between desktop servers. The default is N (no). For example:

  `LSB_MED_SHARED_FILESYS=Y`
- LSB_API_CONNTIMEOUT parameter is used for MED communication with LSF mbatchd. The time-out in seconds when connecting to the Batch system. For example:

  `LSB_API_CONNTIMEOUT=60`
- LSB_API_READTIMEOUT parameter is used for MED communication with LSF mbatchd. The time-out in seconds when reading information from the Batch system. For example:

  `LSB_API_READTIMEOUT=120`
- LSB_MED_CACHEFILE_DEFAULT parameter is optional. Valid values are `yes` and `no`, case insensitive. The default is `yes`, which means that jobs can be cached on all desktop clients connecting to the specified desktop server. To prevent files from being cached on all desktop clients connecting to the specified desktop server, add the following line to `med.conf` after installation:

  `LSB_MED_CACHEFILE_DEFAULT=No`

Note: Desktop users can override this setting when submitting a job by using the + operator to instruct LSF desktop support to write a specific file to the desktop client cache. For example: `bsub -f "local_file+ > remote_file"`. For additional information, refer to the *Platform LSF Desktop Support User's Guide*. To prevent files from ever being cached on the desktop client, set the `SEDDisableCache` parameter in `SEDConfig.xml` to `yes`. For additional information, see "Configuring Desktop Clients" on page 44.

- USER_ID is the user ID of the USER_NAME parameter specified in `install.config`. The USER_NAME owns the Web server process. The default value of USER_ID is `nobody`. If you change this value, you must restart the desktop server.
- GROUP_ID is the group ID of the USER_NAME parameter specified in `install.config`. The USER_NAME owns the Web server process. The default value of GROUP_ID is `nobody`. If you change this value, you must restart the desktop server.

Example med.conf

```
LSB_MED_WEB_URL=http://host1/servlet/P2PServer
LSB_MED_DATA_WEB_URL=http://host
LSB_MED_STAGE_DIR="/usr/local/lsfac/apache/htdocs"
LSB_MED_TOP_DIR=/usr/local/lsfac
LSB_MED_SHARED_FILESYS=y
LSB_API_CONNTIMEOUT=60
LSB_API_READTIMEOUT=120
USER_ID=99
GROUP_ID=99
```

Supporting Desktop Client Resource Requirements

LSF desktop support supports desktop client resource requirement specifications—the user can specify the minimum configuration of the desktop client required to run a particular job. Alternatively, you can set up LSF desktop support queues that define the minimum configuration required for all jobs submitted to that queue.

The desktop client resource types must first be defined to the cluster, and the desktop clients must be enabled to report on their resource availability before the user can successfully specify resource requirements.

LSF desktop support supports both built-in resource definitions and custom resource definitions. See "Built-in resource definitions" on this page for a list of the built-in resource definitions. See "Adding New Resources to the Desktop Server" on page 64 for instructions on creating new resource definitions. See "Resource requirement syntax" on page 63 for the syntax for defining the resources required, either within the queue definition, or within the job submission itself.

Built-in resource definitions

The following resource definitions are built in to LSF desktop support and available for use immediately:

Resource Name	Value Type	Description	Example
cput	string	CPU type	PII or PIII
ncpus	numeric	How many CPUs there are	1
cpusp	numeric	Main frequency of CPU (MHZ)	500
mem	numeric	Free physical memory (MBytes)	256
maxmem	numeric	Max physical memory (MBytes)	64
disk	numeric	Max free disk space LSF desktop support jobs can use (MBytes)	1000

The following CPU types are supported:

CPU Types	Description
Am486	AMD-Am486
ATHLON	AMD Athlon™
ATHLON64	AMD Athlon™ 64
Celeron	Intel Celeron™ processor, models 5 or 6
K5	AMD-K5™
K6	AMD-K6™
K6-2	AMD-K6-2™
OPTERON64	AMD Opteron™ 64
Pen	Intel Pentium® processor
PPro	Intel Pentium® Pro processor
PII	Intel Pentium® II processor, model 3 or 5
PIIX	Intel Pentium® II Xeon™ processor
PIII	Intel Pentium® III processor, Intel Pentium® III Coppermine processor

CPU Types	Description
PIIIX	Intel Pentium® III Xeon™ processor
PIV	Genuine Intel Pentium® 4 processor

Resource requirement syntax

Specify the resource specification as a logical expression, as follows:

```
AC_RES_REQ=resource_name operator value[operator resource_name operator
value...]
```

resource_name

The name of the resource as defined in `ac.restype.xml`.

operator

The operator that defines the relationship between the resource name and the value, or between nested expressions. The operators are listed below in the order of precedence (the order in which they are evaluated, from top to bottom). The operator can be one of the following:

Operator	Description
(	Beginning of expression. Expressions within parentheses are evaluated first.
)	End of expression.
>	Greater than. Use only with value type numeric.
<	Less than. Use only with value type numeric.
>=	Greater than or equal to. Use only with value type numeric.
<=	Less than or equal to. Use only with value type numeric.
==	Equal to. Use with any value type.
!=	Not equal to. Use with any value type.
&&	Logical AND. Both expressions must be true.
||	Logical OR. One of the expressions must be true.

value

The value of the resource to be compared to.

To specify multiple resources, combine specifications together using && or || combination operators. You can join specifications using brackets. For example:

```
"AC_RES_REQ=((cput==PII||mem>256)&&disk>500)"
```

The above example specifies either a Pentium II CPU or 256 MB free physical memory, and 500 MB disk space available for LSF desktop support jobs.

Adding New Resources to the Desktop Server

You can schedule jobs based on available resources. There are many resources built into LSF desktop support, but you can also add your own resources, and then use them in the usual way.

LSF desktop support provides powerful resource selection mechanisms, so that only the desktop clients with the required resources are allowed to run your jobs. For maximum flexibility, characterize your resources clearly, so that users can make the appropriate choices. For example, if some of your desktop clients are connected to both Ethernet and FDDI, while others are only connected to Ethernet, you probably want to define a resource called *fddi* and associate the *fddi* resource to desktop clients connected to FDDI. This way, users can specify the resource *fddi* if they want their jobs to run on desktop clients connected to FDDI.

To add new resources to a desktop server:

1. Log in to the desktop server as the LSF desktop support administrator.
2. Open the file `ac.restype.xml` for editing in an XML editor or in a simple text editor such as Notepad or Textpad. Specify a name and value type for the resource, by adding the following to the file:

   ```
   <Resource>
        <ResName>name</ResName>
         <ValueType>string</ValueType>
   </Resource>
   ```

   where *name* is the name of the resource you are defining, up to 16 characters in length, and ValueType is either `boolean`, `string` or `numeric`. Resource names cannot begin with a number, and cannot contain any of the following characters:

   : . ( ) [+ - * / ! & | < > @ = ' "

   Resource names are case-sensitive, and a resource name cannot be any of the following reserved names:

   cput cpusp ncpus maxmem mem disk

   Refer to “Example: ac.restype.xml” on page 65 for an example of `ac.restype.xml`.
3. Save `ac.restype.xml`.
4. Create an external resource plug-in that detects the new resources for all desktop clients connected to that host. See “To create an external resource plug-in:” on page 66 for instructions.
5. Use one of the following methods to deploy the external resource plug-in to all the desktop clients connected to this host:
   a. Use a software-management tool that can deploy software in enterprise working environments to deploy the external resource plug-in to all the desktop clients. Deploy the plug-in to the same folder as `SED.exe` on the desktop clients.
   b. Use the self-upgrade feature of LSF desktop support to deploy the external resource plug-in to all the desktop clients. See “To build a distributable package for the plug-in:” on page 66 for instructions.

Example: ac.restype.xml

The following is an example of ac.restype.xml, where a new resource name *hname* is defined in addition to the built-in resource definitions:

```
<?xml Version="1.0"?>
<ACRestype>
     <Resource>
            <ResName>cput</ResName>
              <ValueType>string</ValueType>
      </Resource>
      <Resource>
            <ResName>cpusp</ResName>
              <ValueType>numeric</ValueType>
      </Resource>
      <Resource>
            <ResName>ncpus</ResName>
              <ValueType>numeric</ValueType>
      </Resource>
      <Resource>
            <ResName>mem</ResName>
              <ValueType>numeric</ValueType>
      </Resource>
      <Resource>
            <ResName>maxmem</ResName>
              <ValueType>numeric</ValueType>
      </Resource>
      <Resource>
            <ResName>disk</ResName>
              <ValueType>numeric</ValueType>
      </Resource>
      <Resource>
            <ResName>hname</ResName>
              <ValueType>string</ValueType>
      </Resource>
</ACRestype>
```

Adding a Resource Plug-In to the Desktop Clients

To add a new resource definition to LSF desktop support, you need to add the new resource name to `ac.restype.xml`. If you have not already done so, go to "Adding New Resources to the Desktop Server" on page 64 to define the new resource. Then return here to add it to the desktop client plug-in.

About the external resource plug-in

A desktop client supports only one plug-in, called `extRes.dll`. The plug-in must be in a Win32 dynamically-linked library. The plug-in must have two APIs called by the SED:

```
BOOL WINAPI getExtRes(LPTSTR lpBUFFER[out], LPDWORD
lpnSize[in|out])
int  WINAPI getVersion()
```

To create an external resource plug-in:

1. Get the sample project of an external plug-in `extRes.dll` from *ACH_TOP*`/plugin_example`.
2. Open the project with Visual C++ 6.0.
3. Read the files to know how the sample works. Be sure to modify only the recommended files.
4. Write a function to detect the new resource just like the sample function `detectDesktopName`.

   Make sure that you report the resource last. For example:

   ```
   void detectDesktopName(CSEDExtRes* pClassExtRes)
   {
   TCHAR szDesktopName[256]={0};
   DWORD cchBuff=256;
   GetComputerName(szDesktopName,&cchBuff);
   pClassExtRes->reptNewExtRes("hname","string",
   szDesktopName);
   }
   ```

5. In the function `reportAllResInfo()`, call your function:

   ```
   void reportAllResInfo(CSEDExtRes* pC;assExtRes)
   {
   detectDesktopName(pClassExtRes);
   detectIEVersion(pClassExtRes);
   detectOS(pClassExtRes);
   }
   ```

6. In the function `getVersion()`, set the new version number based on the last version number.

To build a distributable package for the plug-in:

You can use this method to deploy the plug-in to the desktop clients using the automatic upgrade feature. Because the plug-in will be downloaded by the desktop client, you should have the upgraded binary file digitally signed by Platform Computing. Follow these steps to create the package:

1. Send the upgraded `extRes.dll` to your Platform Support representative, who will sign and package it for you in `pluginupd.exe`.

2 Copy `pluginupd.exe` to the same directory as that specified in PluginLatestVersionURL in *`APACHE_TOP`*`/htdocs/SED/SEDConfig.xml`.

3 Set the permissions of `pluginupd.exe` to 755.

4 In the file `SEDConfig.xml`, in the PluginLatestVersion parameter, specify the exact version number of the `extRes.dll` and save the file. Each desktop client will check periodically for a change in the version. When it detects a change, it does to the specified location and gets the new executable, which it installs automatically.

Entries in wscache.conf for Desktop Servers

The `wscache.conf` file is present on each desktop server and on the Directory Services host.

The following parameter is present in the *`ACH_TOP`*`/jakarta-tomcat-4.1.30-LE-jdk14/wscache.conf` file of desktop servers.

JOBDIRECTORY

Description The full path to the directory in which the internal state files of the system are stored.

Example

```
JOBDIRECTORY /usr/local/lsfac/wscache
```

Entries in wscache.conf for Directory Services Server

The following parameters are present in the `wscache.conf` file of the Directory Services host.

STATSDIRECTORY

Description The directory in which the job completion log files are stored. Job completion log files are named as follows:

`job_completion_log.`*`host_name`*`.`*`YYYY-MM-DD`*

Example `STATSDIRECTORY /usr/local/lsfac/wscache`

CONFIGDIRECTORY

Description The location where configuration for the directory service is stored. This entry does not exist if directory services are not enabled.

Example `CONFIGDIRECTORY /usr/local/lsfac/wscache`

JOBSTATISTICSURLS

Description Automatically generated value. Do not change. URLS generated by each desktop server indicating where data about the server can be retrieved.

BLACKLIST_POLL_INTERVAL

Description Specifies how frequently hosts on the block list connect to the desktop server to request work. For additional information and an example, see “Controlling Desktop Client Access” on page 92.

WHITELIST_POLL_INTERVAL

Description Specifies how frequently hosts on the whitelist connect to the desktop server to request work. For additional information and an example, see “Controlling Desktop Client Access” on page 92.

DEFAULT_POLL_INTERVAL

Description Specifies how frequently hosts on neither the blacklist nor the whitelist connect to the desktop server to request work. For additional information and an example, see “Controlling Desktop Client Access” on page 92.

FTDIRECTORY

Description Enables high availability for the desktop server. Specifies the absolute path to the directory that will contain the job status pool. Must be defined on every desktop server host. For additional information and an example, see “Configuring Platform LSF Desktop Support High Availability” on page 52.

FTOVERTIME

Description Specifies the amount of time in hours that the desktop server waits to receive job results from the SED. The desktop server waits the specified amount of time before redispatching the job to a new SED. If the original SED contacts the desktop server within the specified time, the job is not redispatched. The value must be an integer greater than zero. For additional information and an example, see “Configuring Platform LSF Desktop Support High Availability” on page 52.

Adding an LSF Desktop server to your cluster

1. Add a server host to your cluster, as described in *Administering Platform LSF*.
2. Convert the new LSF server host to a desktop server.

   See "Converting the LSF Server Host to a Desktop Server" on page 25 in Chapter 3, "Installing LSF Desktop Support".
3. If you are implementing automatic load balancing, add the new desktop server to file `MED.lst`.

   Each line is a server name. The path of this file is specified by the CONFIGDIRECTORY parameter in `install.config`.
4. To prevent desktop servers from running standard LSF jobs, see "Excluding Desktop Servers from LSF Queues" on page 30 in Chapter 4, "Configuring the Components in LSF Desktop Support".
5. Configure the Default LSF Desktop Support queue.

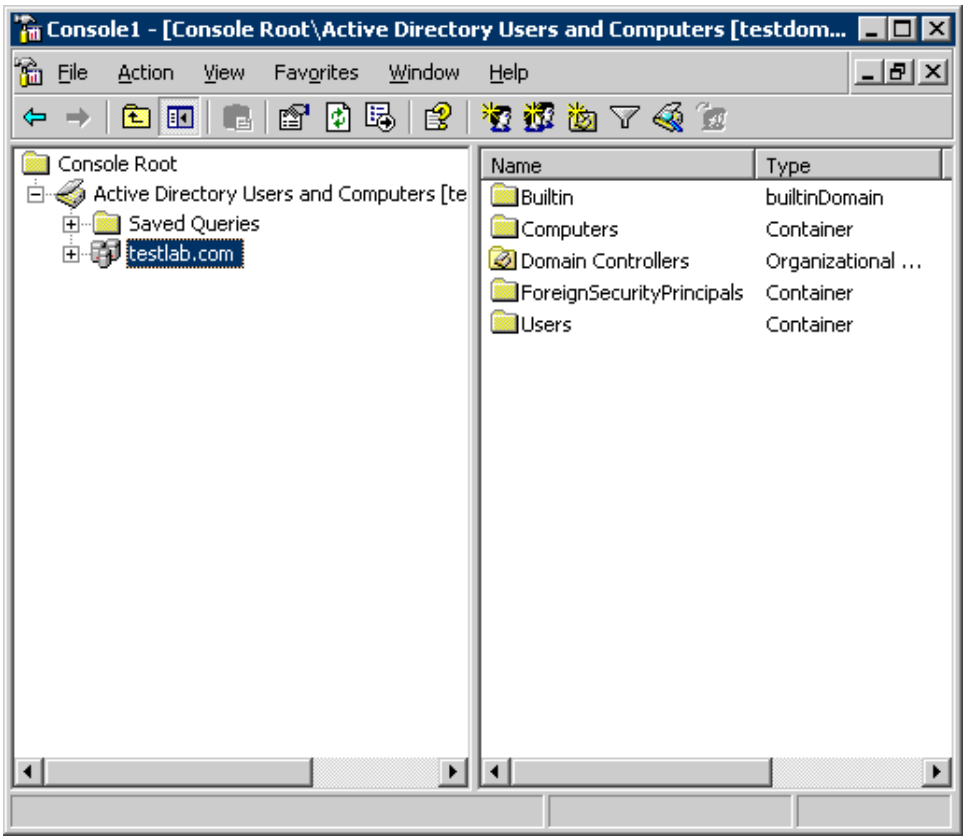
   Edit the HOSTS parameter of AC_QUEUE to include the new desktop server.
6. Update the LSF host configuration.
   a. Edit `lsf.cluster.`*`cluster_name`*.
   b. In the host section, change the host type of the new desktop server to `AC`, and add `activecluster` to its resource list.
7. Update the LSF resource configuration.
   a. Edit `lsb.resources`.
   b. In the Limit section, add the new desktop server to the PER_HOST parameter.
8. Restart `lim` and `mbatchd`.
9. Enable the client to use the new server:
   a. If you do not use automatic load balancing, install a client that points to the new server.
   b. If you use automatic load balancing, client will get new server name from directory server.

Configuring Windows Group Policies to Define Security Settings for SED

If you have `seduser` as a domain account, you can set permissions by defining Group Policy to apply to all desktop in the domain.

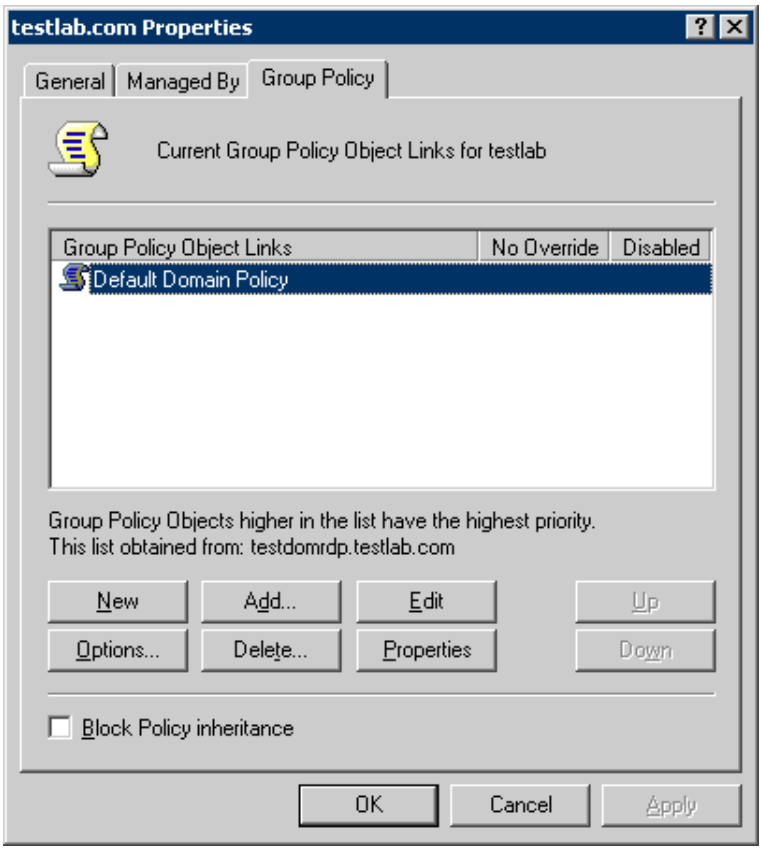
The following example sets Group policy on a Windows 2003 server:

1. Choose **Start > Run**, enter `mmc` to start the Microsoft Management Console, and click **OK**.
2. In the Console1 window, choose **File > Add/Remove Snap-in**.
3. In the Add/Remove Snap-in dialog, click **Add**.
4. In the Add Standalone Snap-in dialog, choose **Active directory users and computers**, and click **Add**.
5. Close the dialog and return to Console1 window.



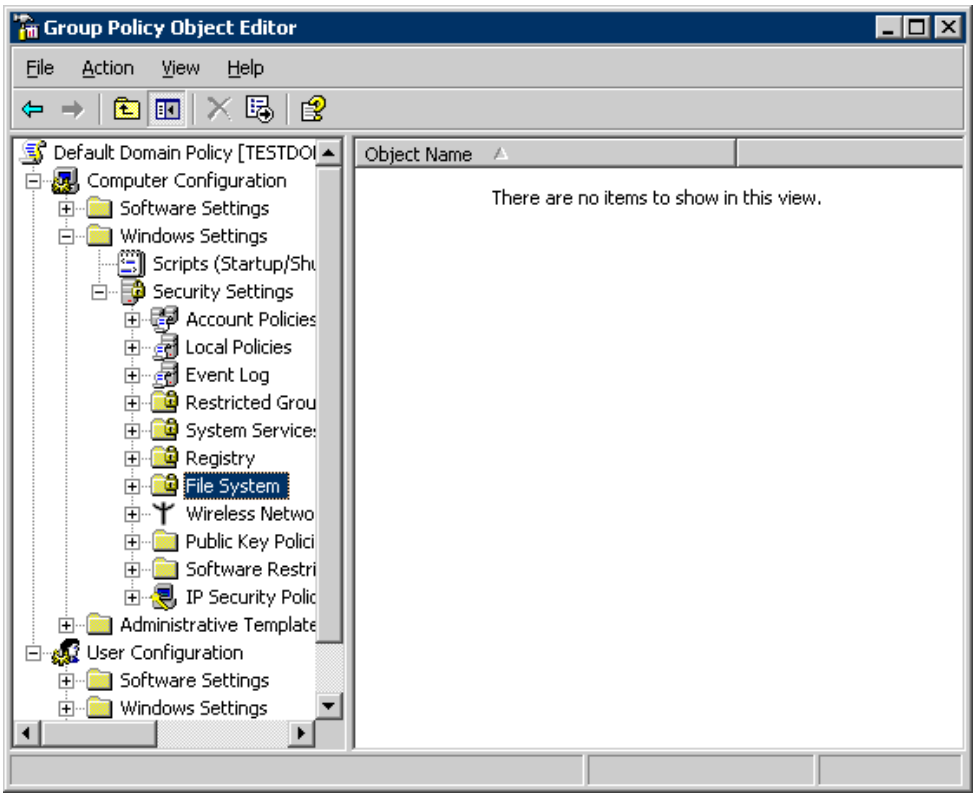
6. Click **+** to expand Active Directory Users and Computers.
7. Right click on the Organizational Unit (OU) you want to create a Group Policy Object for (for example, `testlab.com`), and select **Properties**.

8 Select the **Group Policy** tab.



9 Select the Group Policy Object of your domain, and click **Edit** to open the Group Policy Object Editor.

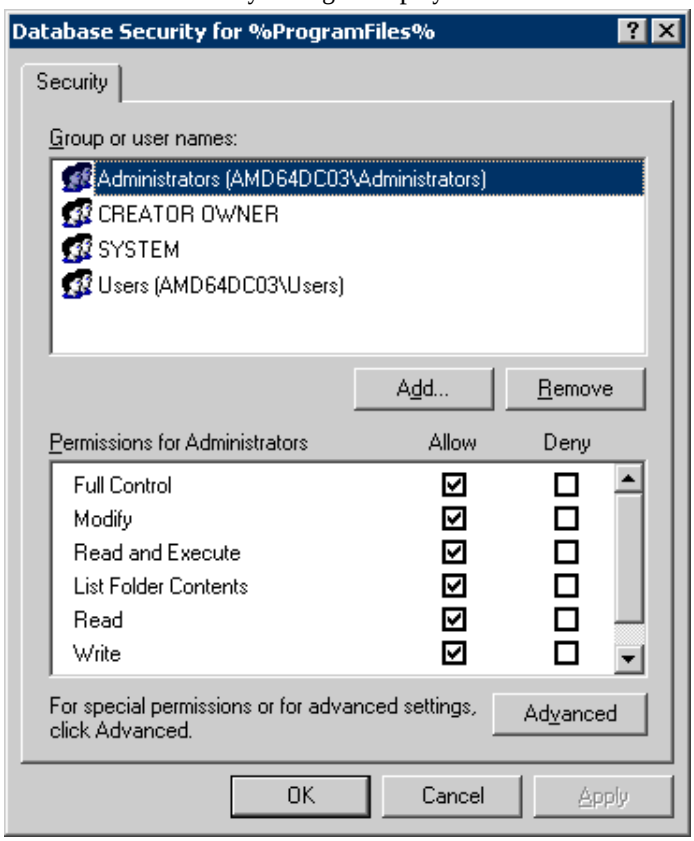
10 Expand **Computer Configuration > Windows Settings > Security Settings > File System**.



11 Right click **File System**, and click **Add File**.

12 In the Add a file or folder dialog, select a driver or folder where you want to set the permissions for `seduser`, and click **OK**.

The Database Security dialog is displayed.



13 In the Database Security dialog box, set the permissions of `seduser` for that driver or folder, and click **OK**.

CHAPTER 5

Managing LSF Desktop Support

This chapter describes how to manage LSF desktop support, from starting and stopping desktop servers to managing upgrades to the desktop client software.

Contents

Starting the Desktop Servers

Many of the components of the LSF desktop support start automatically when the desktop server is started. However, in some instances, you may need to start a component manually.

To start the desktop server:

1. Make sure that Apache and Tomcat are started. If not, do the following:
   a. Log in as root.
   b. Change the directory to the LSF desktop support home directory:
      cd *ACH_TOP*/etc
   c. Run the following command:
      lsfac_daemons start
2. Run the following command as LSF administrator:
   badmin hstartup *host_name*
   where *host_name* is the name of the desktop server. This starts the desktop server.

To start a desktop client manually:

1. On the desktop client machine, go to the Services Control Panel.
2. Find the service called SED in the list of services.
3. Start SED.

To start EGO managed Web services manually

If EGO management of LSF desktop support services is enabled, you must use an EGO command to start a managed service.

From the command line, enter the command:

egosh service start LSFDesktopApache LSFDesktopTomcat

Important With EGO management of LSF desktop support services enabled, you should not use the command lsfac_daemons to start Apache or Tomcat services. Instead, you should use the egosh command to start these services.

Stopping the Desktop Servers

Use the following procedures to stop the LSF desktop support components.

To stop the desktop server:

1. Run the following command as LSF administrator:

   `badmin hshutdown` *`host_name`*

   where *host_name* is the desktop server. This stops the desktop server.

2. Stop Apache and Tomcat, by doing the following:
   a. Log in as root.
   b. Change the directory to the LSF desktop support home directory:

      `cd ACH_TOP/etc`

   c. Run the following command:

      `lsfac_daemons stop`

To stop a desktop client manually:

1. On the desktop client machine, go to the Services Control Panel.
2. Find the service SED in the list of services.
3. Stop SED.

To stop EGO managed Web services manually

If EGO management of LSF desktop support services is enabled, you must use an EGO command to stop a managed service.

From the command line, enter the command:

`egosh service stop LSFDesktopApache LSFDesktopTomcat`

EGO shuts down Apache and Tomcat services on all LSF desktop support hosts.

Important With EGO management of LSF desktop support services enabled, you should not use the command lsfac_daemons to stop Apache or Tomcat services. Instead, you should use the egosh command to stop these services.

Authenticating Users and Controlling User Access

This section describes digest authentication. By default, digest authentication is disabled. If you have not enabled digest authentication, as described in "Converting the LSF Server Host to a Desktop Server" on page 25, you can skip this section.

User authentication

The desktop server installation creates two digest-authenticated logins within the Apache configuration.

The following users are created:

User name	Password	Purpose
desktop	desktop	Administrative access to the servlet Web pages.
admin	admin	For the desktop clients to authenticate when requesting work and transferring files.

The following groups are created that contain the users:

To modify a user password:

1. Change to: `$ACH_TOP/apache/bin/htdigest`
2. Type: `$ACH_TOP/apache/conf/digest.db ActiveCluster` *`user_name`*, where you want to change the password for the specified user name.

   Note: If you want to create a new password file (`digest.db`), use the `-c` option. For example, type: `htdigest [ -c ] digest.db ActiveCluster` *`user_name`*

3. When prompted, follow on-screen instructions to change the password. Note that once you enter the password, it is hashed and encrypted by the MD5 algorithm.

   Note: Digest passwords cannot be longer than eight characters.

For information on adding additional users, refer to the Apache documentation.

Controlling User Access to Directories

You can use digest authentication, described in "Authenticating Users and Controlling User Access" on page 80, to control access to directories.

The Apache server configuration file (`httpd.config`) used by the desktop server has directives to control access using digest authentication. An administrator can change which groups or users have permission to access which directories and locations. The desktop server installation configures security as follows:

This directory....	Can be accessed by these groups...
`lsf_datadir/*`	admin and desktop
`SED/*`	admin and desktop
`stats/*`	admin
`servlet/SedSoap`	admin and desktop
`servlet/DynamicSedConfig`	admin and desktop

This directory....	Can be accessed by these groups...
`servlet/P2PStatus`	admin
`servlet/StatusViewer`	admin

Viewing Overall Job Statistics

When you want to see overall job statistics in LSF desktop support, you can use the LSF desktop support Web page. This option is available only if your site uses directory services.

To view overall job statistics for the LSF desktop support environment:

1. Open a Web browser.
2. Go to `http://host_name/servlet/Statistics`, where *host_name* is the name of the LSF desktop support Directory Services host. The Status page is displayed in XML format.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <Statistics>
    <TotalPCs>3</TotalPCs>
    <NumberOfDownloadedJobs>73</NumberOfDownloadedJobs>
    <NumberOfCompletedJobs>73</NumberOfCompletedJobs>
    <HarvestCPUTime>1659 </HarvestCPUTime>
  </Statistics>
```

   The following statistics are displayed:

   - `<TotalPCs>`—The number of unique desktops participating
   - `<NumberOfDownloadedJobs>`—The number of jobs downloaded
   - `<NumberOfCompletedJobs>`—The number of jobs completed
   - `<HarvestCPUTime>`—The amount of CPU time (in seconds) harvested in this project

   The statistics are updated every five minutes.

Viewing the Job Statistics of a Desktop Server

When you want to see the status of the desktop servers at a glance, you can use the LSF desktop support Web pages.

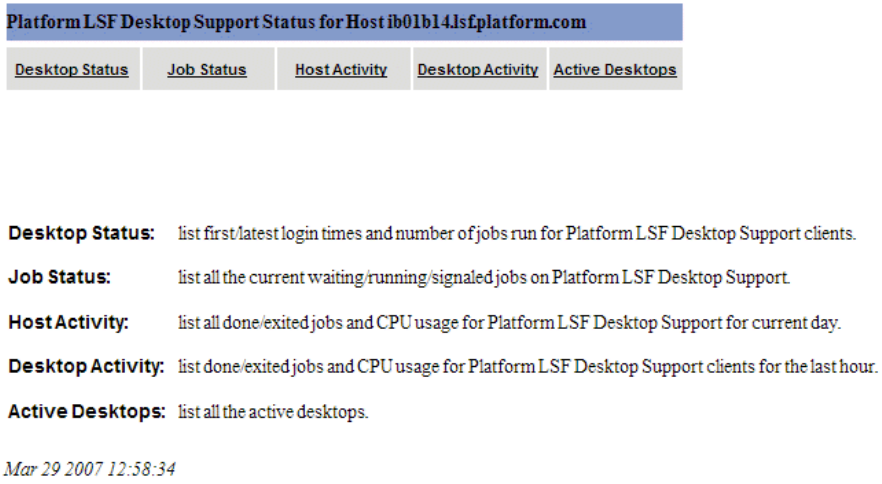
To view the job statistics of a desktop server:

1. In a Web browser, open `http://host_name`, where *host_name* is the name of the desktop server.

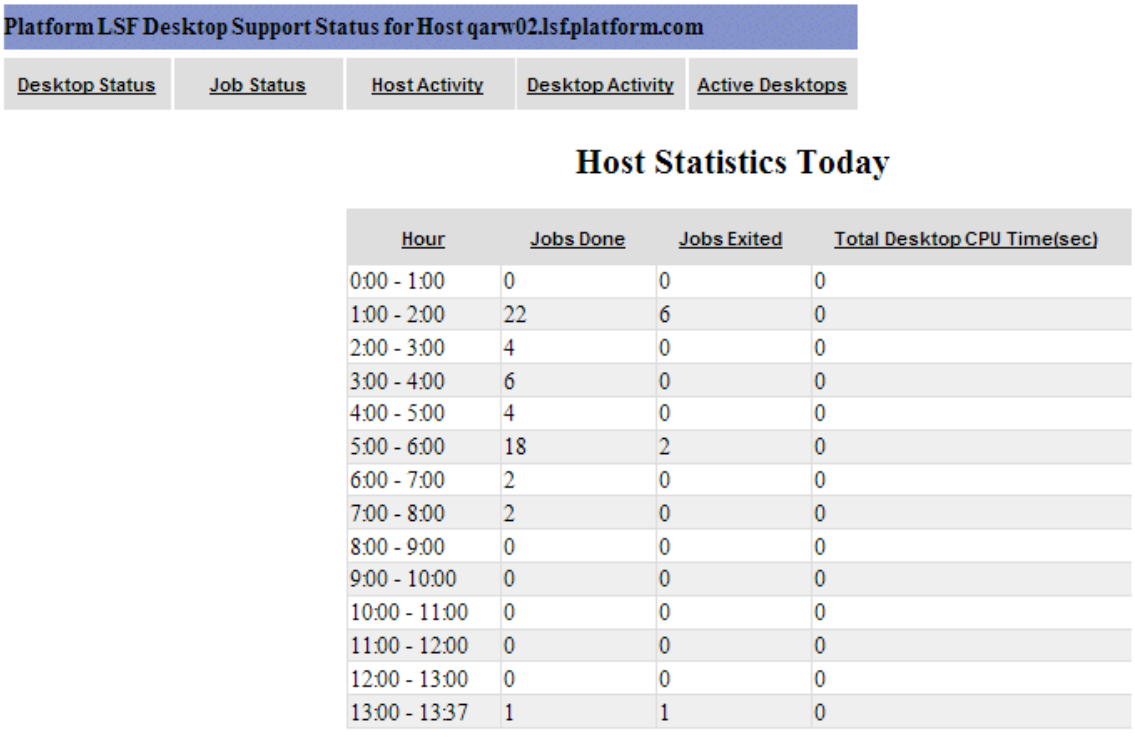
   Note: You will be prompted to log in with the user name and password defined in in `$APACHE_HOME/conf/digest.db`.

2. Click the **View hosts and queued jobs** link.

   The Status page opens.



3 Click **Host Activity**. The Host Activity page opens.



Platform LSF Desktop Support Status for Host qarw02.lsfplatform.com

Desktop Status | Job Status | Host Activity | Desktop Activity | Active Desktops

Host Statistics Today

Hour	Jobs Done	Jobs Exited	Total Desktop CPU Time(sec)
0:00 - 1:00	0	0	0
1:00 - 2:00	22	6	0
2:00 - 3:00	4	0	0
3:00 - 4:00	6	0	0
4:00 - 5:00	4	0	0
5:00 - 6:00	18	2	0
6:00 - 7:00	2	0	0
7:00 - 8:00	2	0	0
8:00 - 9:00	0	0	0
9:00 - 10:00	0	0	0
10:00 - 11:00	0	0	0
11:00 - 12:00	0	0	0
12:00 - 13:00	0	0	0
13:00 - 13:37	1	1	0

Mar 29 2007 13:37:04

The Host Activity page displays the following information:

- The number of jobs done within a certain number of hours
- The number of jobs exited within a certain number of hours
- The total desktop client CPU time (in seconds) used within a certain number of hours

Viewing the Status of a Job

You can look at the status of one or more jobs in the LSF desktop support using the LSF desktop support Web pages.

To view the status of one or more jobs:

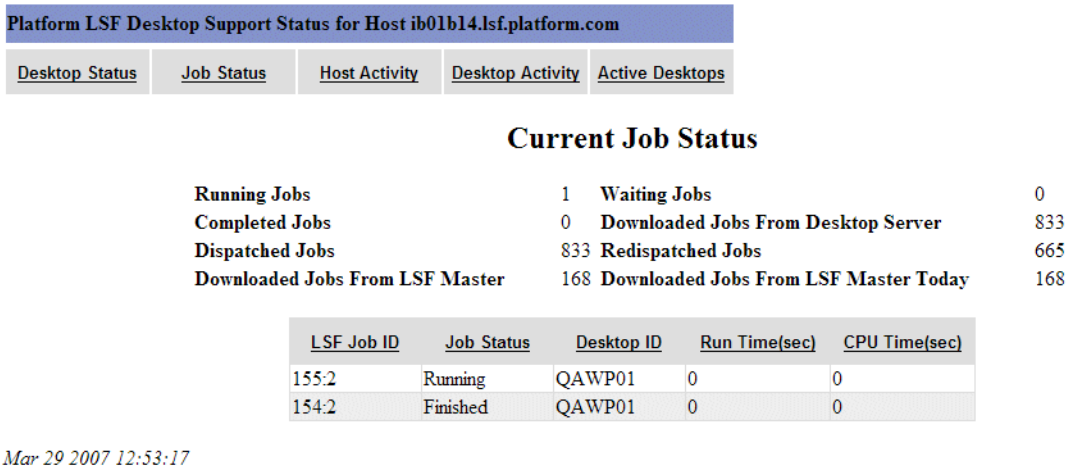
1 In a Web browser, open `http://host_name`, where *host_name* is the name of the desktop server.

Note: You will be prompted to log in with the user name and password defined in in `$APACHE_HOME/conf/digest.db`.

2 Click the **View hosts and queued jobs** link.

The Status page opens.

3 Click **Job Status**. The Job Status page opens.



The Job Status page displays the following information:

- **Running Jobs**—The number of jobs running in LSF desktop support
- **Waiting Jobs**—The number of jobs waiting in LSF desktop support

- **Completed Jobs**—The number of jobs completed in LSF desktop support
- **Downloaded Jobs From Desktop Server**—The number of jobs the desktop client downloaded from WS.

  Downloaded Jobs in previous releases has changed to **Downloaded Jobs From Desktop Server** to distinguish it from **Downloaded Jobs From LSF Master**.

- **Dispatched Jobs**—The number of jobs MED dispatched to WS. This number includes re-dispatched jobs.
- **Redispatched Jobs**—The number of jobs MED redispatched to WS. Redispatch can occur, for example, when a user pauses a running job. MED will delete the old job from WS and redispatch the job to WS.
- **Downloaded Jobs From LSF Master**—The total number of jobs the LSF master host (`mbatchd`) dispatched to the LSF desktop support server (MED).
- **Downloaded Jobs From LSF Master Today**—The number of jobs `mbatchd` dispatched to MED today.
- A list of all jobs in the LSF desktop support by job ID:
  - The current status of each job
  - If the job is running, it indicates on which desktop client it is running
  - If the job recently completed, it indicates the time it took to run, and the actual CPU time used

Viewing Desktop Client Statistics

You can view the following information about a desktop client:

- The job statistics for the day, if the desktop client was active: the number of successful and unsuccessful jobs
- The desktop client history since installation

To view the desktop client job statistics:

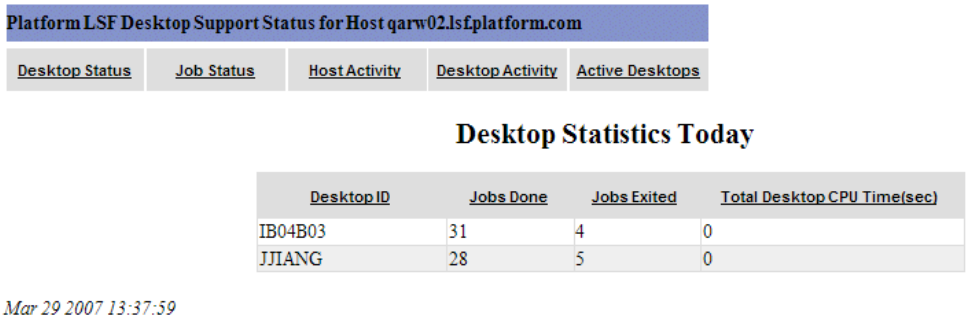
1. In a Web browser, open `http://host_name`, where *host_name* is the name of the desktop server.

   Note: You will be prompted to log in with the user name and password defined in in `$APACHE_HOME/conf/digest.db`.

2. Click the **View hosts and queued jobs** link.

   The Status page opens.

3. Click **Desktop Activity**. The Desktop Activity page opens.



   The Desktop Activity page displays the following information:

   - A list of the desktop clients that ran work since midnight
   - The number of jobs done in the last hour on each desktop client
   - The number of jobs exited in the last hour on each desktop client
   - The total CPU time (in seconds) used in the last hour on each desktop client

To view the desktop client status:

1 In a Web browser, open `http://host_name`, where *host_name* is the name of the desktop server.

Note: You will be prompted to log in with the user name and password defined in in `$APACHE_HOME/conf/digest.db`.

2 Click the **View participating hosts** link.

The Desktop Status page opens.

Tip: You can also access the Desktop Status page by selecting Desktop Status for the Status page for a particular host.

The Desktop Status page displays the following information:

- **Desktop ID**—A list of the desktop clients in this LSF desktop support
- **Version**—The current version of the desktop client
- **Plug-in Version**—The plug-in version of the desktop client
- **First Login**—The date the desktop client first requested work
- **Last Login**—The last time the desktop client requested work or returned status
- **Jobs**—The number of jobs run by each desktop client
- **Status**—The current status of the desktop client

Desktop client status

Idle—The desktop client can run a job but has not downloaded a job from the desktop server. By default, reported every 300 seconds when SED is idle.

Running—SED is running a job. By default, reported every 300 seconds when a job is running.

Suspend in Power saving mode—Desktop goes to power saving mode, for example, the desktop user selects **Stand by** from Windows shut down dialog to switch the system to power-saving mode.

Suspend in Screensaver mode—The desktop client is running in screen saver mode (**Run when screen saver runs**), and the screen saver stops running.

Suspend in Idle mode—The desktop client is running in idle mode (**Run when idle**), and the desktop client switches from idle to not idle. In idle mode, taking any action when the console is idle while a job is running, changes the status to **Suspend in Idle mode**.

Suspend in Logon mode—The desktop client is running in logon mode (**Run when not logged on**), and a user logs on to the desktop client.

Stop—The desktop client service is stopped. For example, in **Run always** mode, stops the SED service while a job is running changes the status to **Stop**.

Opt out—The desktop client is paused by user (**Pause for 8 hours**).

Unknown—The desktop client service has not reported for a long time (by default, for more than 600 seconds); for example, while the Tomcat application server is restarting.

Viewing the State of EGO Managed Services

If EGO management of LSF desktop support services is enabled, you can use an EGO command to view the state of all managed services.

View the status of EGO services

From the command line, enter the command `egosh service list`

The state of each EGO service is displayed:

- DEFINED—The service definition is loaded or created by API without syntax error.
- INIT—Service is being initialized. Dependencies are being checked. A transitional state before becoming `ALLOCATING`.
- ALLOCATING—Allocating resources to the service. A transitional state before becoming `STARTED`. (Note: A service could remain in this state for some time if resources are not available to start it.)
- STARTED—The service is active and running. Service instances are running (the minimum number have been started).
- ERROR—Error has been detected. Needs manual debugging.
- DEALLOCATING—The service has been disabled by the system. Cleanup is occurring. A transitional state before becoming `DEFINED`.

Determining Which Desktop Clients are Active

You can see a list of desktop clients that are currently running jobs in an LSF desktop support.

To see which desktop clients are active:

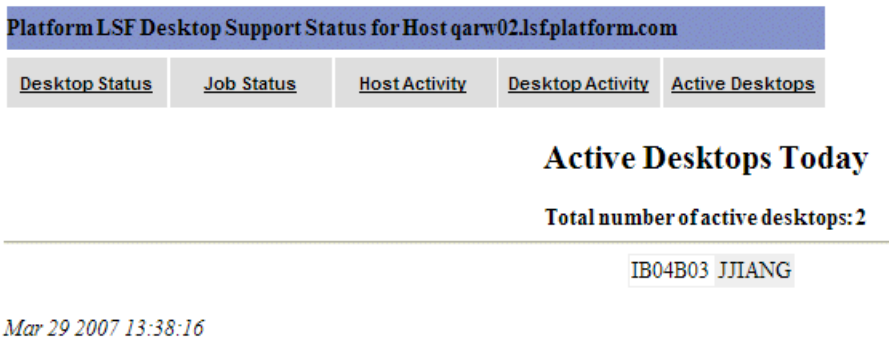
1 In a Web browser, open `http://host_name`, where *host_name* is the name of the desktop server.

Note: You will be prompted to log in with the user name and password defined in in `$APACHE_HOME/conf/digest.db`.

2 Click the **View hosts and queued jobs** link.

The Status page opens.

3 Click **Active Desktops**. The Active Desktops page opens.



The Active Desktops page displays the following information:

- A list of all the desktop clients in this environment that have run LSF desktop support jobs since midnight
- The total number of active desktop clients in this environment since midnight

Controlling Desktop Client Access

Overview

Under certain circumstances, you may want to restrict a desktop client's access to the desktop server to prevent it from requesting work. For example:

- If there is a particularly slow desktop client and you want a job to run on a faster machine.
- If the desktop client user frequently pauses the desktop client on the machine.
- If a desktop client is configured incorrectly and it has trouble completing jobs.
- If you want to install LSF desktop support as part of the standard desktop client build, but you want to exclude certain systems from running jobs.

You can create either kind of list, depending on how many desktop clients you need to add to either list. For example, if you want to exclude only a few desktop clients, then you should create a blacklist, while if you want include only a few desktop clients, then you should create a whitelist.

Note: If you create both a blacklist and a whitelist, and the same desktop client is in both lists, then it is excluded from LSF desktop support, and cannot request work.

Identifying desktop clients

You can identify desktop clients in these lists using:

- SED IDs (case insensitive)
- Specific IP addresses
- Multiple ranges of IP addresses and subnets

There are two ways to control which desktop clients can request work:

- A *blacklist* lists the desktop clients that cannot request work.
- A *whitelist* lists the desktop clients that are allowed to request work.

Default settings

The following table displays the default settings for how frequently hosts on these lists connect to the desktop server to request work:

Host type	How frequently the hosts connect to the desktop server to request work
Hosts on the blacklist	every 30 days
Hosts on the whitelist	every 5 minutes
Other hosts	every 24 hours

To modify these settings, edit the following parameters in `wscache.conf`:

`BLACKLIST_POLL_INTERVAL`

`WHITELIST_POLL_INTERVAL`

`DEFAULT_POLL_INTERVAL`

Denying access to a desktop client has no impact on any jobs it is currently running. However, the next time the desktop client requests a new job, it will be told there are no new jobs. It will continue to request a new job at the specified polling interval until you remove the desktop client ID from the list of locked out desktop clients. For information on setting the polling interval for desktop clients, see "Modifying the polling intervals" on page 94.

Note for users upgrading from a previous version of ActiveCluster: Previous versions of ActiveCluster (now LSF desktop support) used the *ACH_TOP*/wscache/deny.lst file to create a blacklist. In the interests of backward compatibility, any hosts in deny.lst are denied access. If a machine is specified in both deny.lst and allow.xml, it is denied access. If you modify deny.lst, you need to restart the Web server for your changes to take effect.

Sample blacklist and whitelist files

The sample files allow.xml.sample and deny.xml.sample are in the folder specified by the CONFIGDIRECTORY parameter in install.config. If CONFIGDIRECTORY is not defined, the default folder is $ACH_TOP/config. These files are owned by root and have access permission mode 644, so that the Web server can read them.

Using a blacklist to deny access to a desktop client

To block a host, you must create a blacklist containing its ID.

1. Create or open the file deny.xml for editing.
   - You require root authority to edit this file.
   - This file requires permission mode 644 so that the Web server can read it.
   - This file should be in the folder specified by the CONFIGDIRECTORY parameter in install.config. If CONFIGDIRECTORY is not defined, the default folder is $ACH_TOP/config. This is described in greater detail in "Converting the LSF Server Host to a Desktop Server" on page 25.

   Note: If CONFIGDIRECTORY is not defined, and deny.xml does not exist in $ACH_TOP/config and /usr/local/lsfac/wscache, the Web server considers deny.xml to be undefined, and there is no blacklisting control over desktop client requests for jobs.
2. Add the desktop client ID for the desktop client to exclude from LSF desktop support, using one of the formats described in "Specifying a desktop client in a list" on page 94.
3. Save the file.

Using a whitelist to allow host access to a desktop client

To whitelist a host, you must create a whitelist containing its ID.

1. Create or open the file allow.xml for editing.
   - You require root authority to edit this file.
   - This file requires permission mode 644 so that the Web server can read it.
   - This file should be in the folder specified by the CONFIGDIRECTORY parameter in install.config. If CONFIGDIRECTORY is not defined, the default folder is $ACH_TOP/config. This is described in greater detail in "Converting the LSF Server Host to a Desktop Server" on page 25.

   Note: If CONFIGDIRECTORY is not defined, and allow.xml does not exist in $ACH_TOP/config and /usr/local/lsfac/wscache, the Web server considers allow.xml to be undefined, and there is no whitelisting control over desktop client requests for jobs.

2 Add the desktop client ID for the desktop client to include in LSF desktop support, using one of the formats described in “Specifying a desktop client in a list” on page 94.
3 Save the file.

Specifying a desktop client in a list

You can specify desktop clients in a blacklist or a whitelist using:

- SED IDs (case insensitive)
- Specific IP addresses
- Multiple ranges of IP addresses and subnets

Both the blacklist (`deny.xml`) and the whitelist (`allow.xml`) have the same structure and use the following DTD (document type definition):

```
<!ELEMENT hostlist (host*)>
<!ELEMENT host (#PCDATA)>
<!ATTLIST host kind CDATA #IMPLIED>
```

where the `kind` attribute is `IP` (default) or `SEDID`:

- If the kind is `IP` (the default), then the content of the element must be an IP address or a range of addresses. A dash (-) specifies an inclusive range of numbers, while an asterisk (*) represents any additional numbers (up to 255). For example:

  ```
  123.45.67.89
  123.45.*.*
  123.45.67.1-5
  ```

- If the kind is `SEDID`, then the content of the element must identify a single desktop client. For example:

  ```
  MY_PC
  ```

Suppose that `deny.xml` has the following content:

```
<?xml version="1.0"?>
<hostlist>
<host kind="SEDID">MY_PC</host>
<host>123.45.67.89</host>
<host kind="IP">123.45.*.*</host>
<host>123.45.67.1-5</host>
</hostlist>
```

then LSF desktop support blocks the following hosts:

- Hosts with a SED ID of `MY_PC`
- Host with IP address 123.45.67.89
- All hosts in the 123.45 subnet
- Hosts 123.45.67.1, 123.45.67.2, 123.45.67.3, 123.45.67.4, and 123.45.67.5

The desktop server automatically detects the creation or modification of the blacklist or whitelist.

Modifying the polling intervals

You can specify separate polling intervals, in seconds, for blacklisted, whitelisted, and other desktop clients in `$TOMCAT_HOME/wscache.conf`.

- BLACKLIST_POLL_INTERVAL

- WHITELIST_POLL_INTERVAL
- DEFAULT_POLL_INTERVAL

These polling intervals override the setting for the SEDPollInterval parameter in the `$ACH_TOP/config/SEDConfig.xml`. If any of these properties is not defined, the setting for SEDPollInterval in `$ACH_TOP/config/SEDConfig.xml` is used instead. For information on this parameter, see "Changing the time between work requests from an idle desktop client" on page 44.

For example, the following configuration sets the polling interval for blacklisted hosts to 24 hours, whitelisted hosts to 5 minutes, and all others to 30 minutes:

```
BLACKLIST_POLL_INTERVAL=86400
WHITELIST_POLL_INTERVAL=300
DEFAULT_POLL_INTERVAL=1800
```

If you change the blacklist, whitelist or default poll intervals, you must restart the desktop server for the new values to take effect.

Managing Desktop Client Availability

Depending on the kind of work the desktop client user does, and how compute-intensive it is, the desktop client user may want to limit the amount of time in which the desktop client processes LSF desktop support jobs. The desktop client user can choose from four methods:

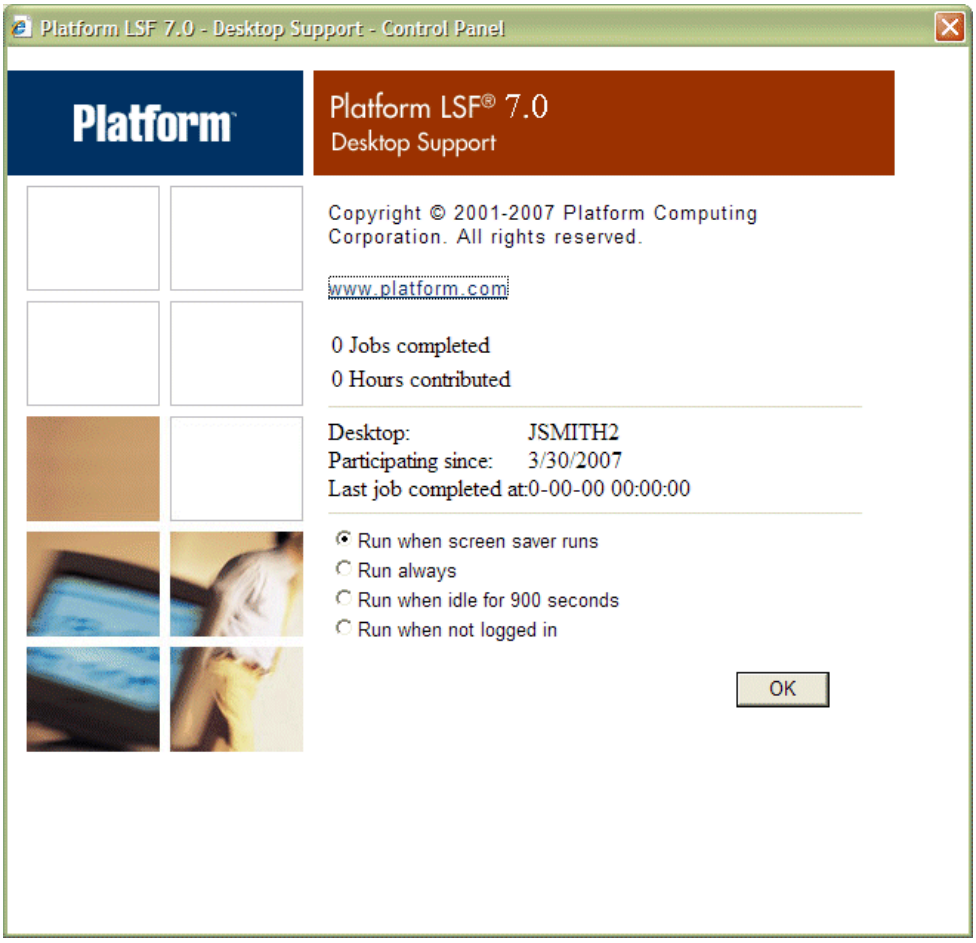
- Always mode (**Run always**), where the desktop client runs jobs in the background at all times. The desktop client runs jobs at the lowest priority to minimize interference with the desktop client user. Generally, users are not aware of the jobs running, unless their machines have limited memory available, or if a job requires transfer of one or more large files. However, if users feel that jobs are impacting the performance of their machines, they can pause the desktop client, as described in "Pausing desktop client processing" on page 97.
- Screensaver mode (**Run when screen saver runs**), where the desktop client runs jobs only when a screen saver is active (default).
- Logon mode (**Run when not logged in**), where the desktop client runs jobs only when no user is signed on to the desktop client. A user is logged on during an interactive session initiated from a Windows logon dialog box.
- Idle mode (**Run when idle for 900 seconds**), where the desktop client runs jobs only when the desktop client has been idle for a configurable period.

In any mode, the desktop client user can pause the desktop client for eight hours. The desktop client stops any currently running job and waits until the time period elapses before requesting another job. Rebooting the machine resumes the desktop client processing.

Specifying the desktop client mode

1. Right-click the LSF desktop support icon in the system tray. The desktop client menu opens.

2 In the menu, click **Details**. The desktop client control panel opens.



3 Select one of the desktop client modes described above.

4 Click **OK** to close the desktop client panel.

Note: If the `SEDModeSelectableByUser` parameter is set to **no** in `SEDConfig.xml`, information about the desktop client mode is not displayed.

Pausing desktop client processing

1 Right-click the Platform LSF Desktop icon in the system tray. The desktop client menu opens.

2 In the menu, select **Pause for 8 hours**.

Note: If the `SEDOptoutInterval` parameter in `SEDConfig.xml` has been modified from its default value of 28800 seconds (eight hours), another interval is displayed in the menu. For information on modifying the interval for which the desktop client pauses, see "Setting the time period for which a desktop client pauses" on page 45.

The desktop client will not run jobs until either the configured time period has elapsed or the desktop client user selects **Resume** on the desktop client menu.

Note: The Pause option is disabled if the `SEDModeSelectableByUser` parameter is set to `no` in `SEDConfig.xml`.

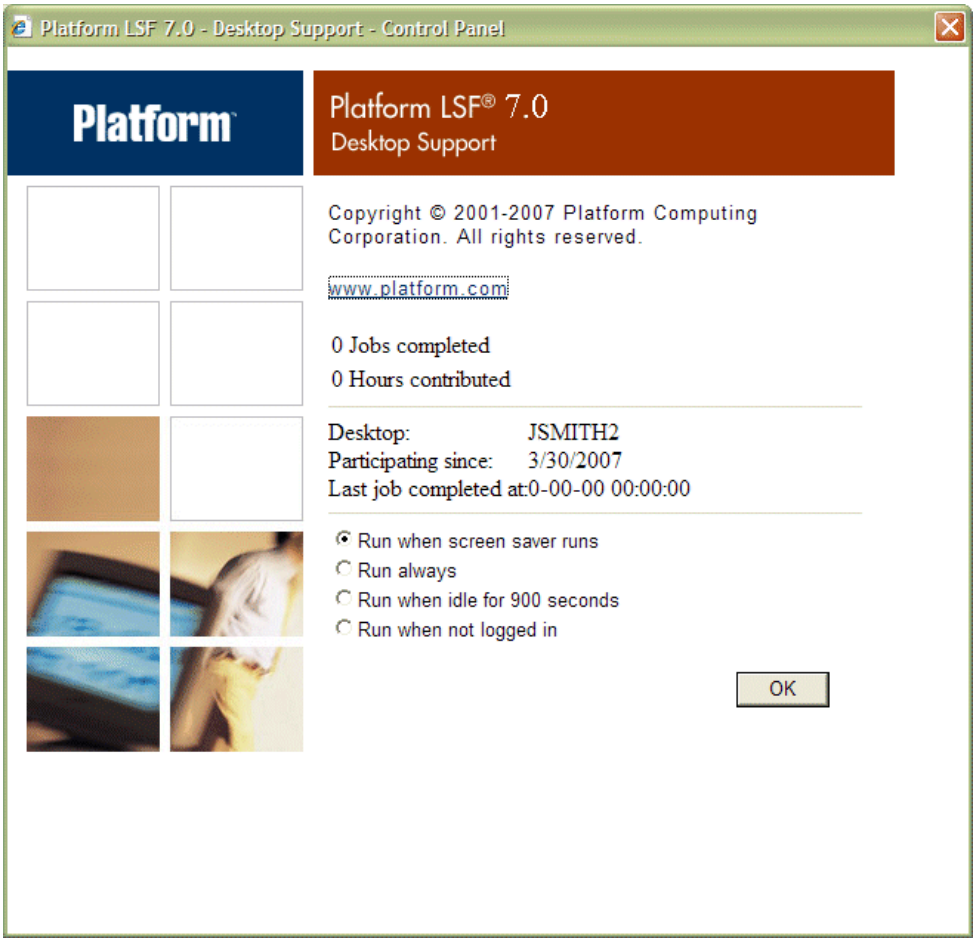
Displaying Current Contribution of the Desktop Client

Depending on your particular work environment, those users whose desktop clients are running jobs in an LSF desktop support may want to know what their contribution to the computations has been. Users can see the following statistics about their desktop clients from the **Details** option:

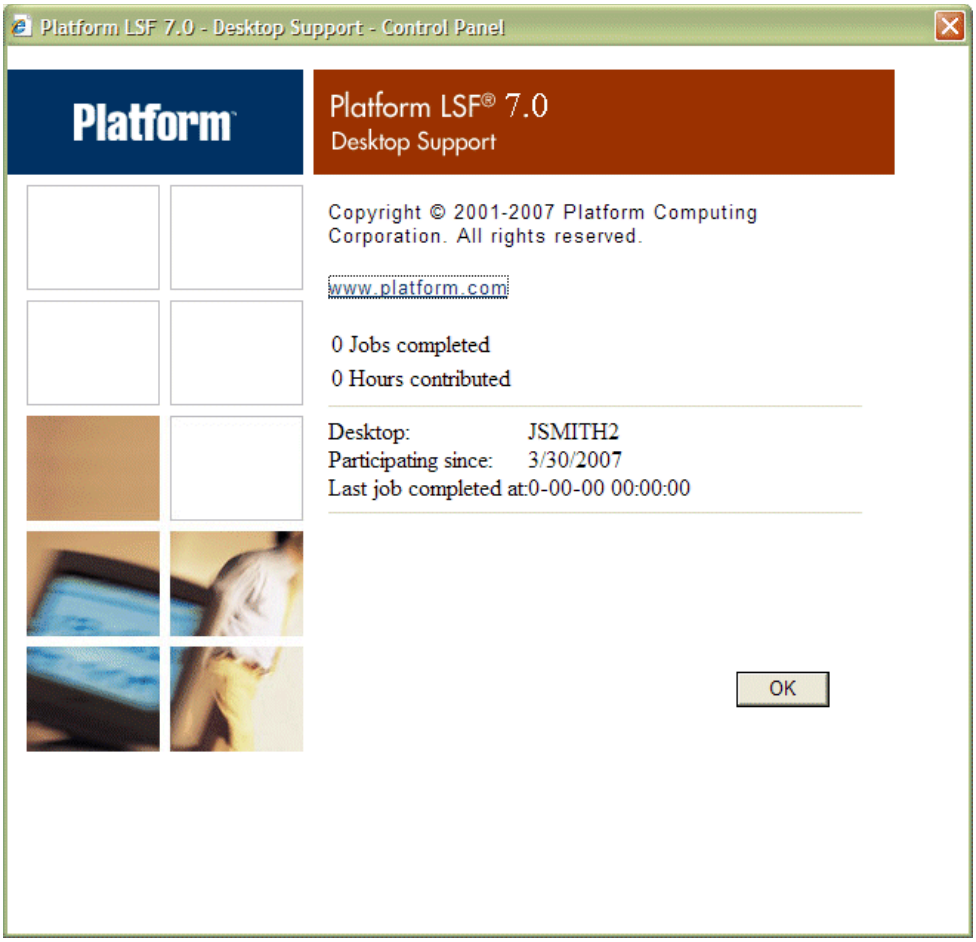
- Total number of jobs completed
- The number of computing hours contributed by this desktop client
- The date this desktop client began participating in the LSF desktop support
- The completion time of the last job run

To display the current desktop client contribution:

1. Right-click on the LSF desktop support icon in the system tray. The desktop client menu opens.
2. From the menu, choose **Details**. The Platform LSF desktop support panel opens, showing the current statistics.
   - If `SEDModeSelectableByUser` is set to yes in `sedsetup.cmd` before the silent installation or afterward in `SEDConfig.xml`, the Platform LSF Desktop desktop client control panel opens.



- If `SEDModeSelectableByUser` is set to no in `sedsetup.cmd` before the silent installation or afterward in `SEDConfig.xml`, the following control panels opens.



For information on `SEDModeSelectableByUser`, see "Installing the Desktop Client" on page 33 or "Configuring Desktop Clients" on page 44.

Specifying How Frequently the Desktop Server Submits a Job

In general, if the job on the desktop client does not complete successfully, the desktop server resubmits it up to the number of times specified by the `LSB_MED_MAX_JOB_DISPATCH` parameter in `lsf.conf`. By default, this is 5. For information on modifying `lsf.conf`, see the *Platform LSF Reference*.

Upgrading

You need to be aware that upgrading LSF will affect your LSF desktop support, because the upgrade may replace the desktop server daemon with a standard LSF slave batch daemon. Note that the upgrade procedure only updates the desktop client, it does not rerun the installation or change any configuration settings.

When you upgrade LSF:

1. Upgrade LSF.
2. Move the new `sbatchd` to `sbatchd.lsf`.
3. Create an `sbatchd` link pointing to `sbatchd.ac`.

To upgrade a desktop client:

The desktop client software is designed to self-upgrade after it is installed. It looks periodically for a new version of the software.

1. Download the latest version of `sedupd.exe` from `ftp.platform.com`.

   Note: The `sedupd.exe` file is included with patch kits.

2. Copy the executable to the same directory as that specified in `SEDLatestVersionURL` in *`CONFIGDIRECTORY`*`/SEDConfig.xml`.

   If you have multiple desktop servers, you need to copy the executable to each host location.

3. Set the permissions of `sedupd.exe` to 755.
4. In the file `SEDConfig.xml`, in the SEDLatestVersion parameter, specify the exact version number of the desktop client software and save the file. Each desktop client will check periodically for a change in the version. When it detects a change, it goes to the location specified and gets the new executable, which it installs automatically.
5. When the desktop client is upgraded, a new SEDSystray is installed on the desktop client. However, it must be started manually. Have the desktop client user restart it as follows:
   a. From the **Start** menu, select **Programs**.
   b. Select **Platform LSF Desktop > Show Icon**.

To downgrade a desktop client:

The desktop client software is designed to self-upgrade after it is installed. It looks periodically for a new version of the software. However, if you want to revert to an older version of the desktop client, you can.

1. In `SEDConfig.xml`, enable downgrade of the desktop clients, by adding the following line:

   ```
   <SEDDowngradeEnabled>yes</SEDDowngradeEnabled>
   ```

2. Copy the desired version of `sedupd.exe` to the same directory as that specified in SEDLatestVersionURL in *`CONFIGDIRECTORY`*`/SEDConfig.xml`.

   Note: The `sedupd.exe` file is included with patch kits.

   If you have multiple desktop servers, you need to copy the executable to each host location.

3 Set the permissions of `sedupd.exe` to 755.

4 In the file `SEDConfig.xml`, in the SEDLatestVersion parameter, specify the exact version number of the desktop client software and save the file. Each desktop client will check periodically for a change in the version. When it detects a change, it will look to see if the variable SEDDowngradeEnabled is set to yes. If it is, the desktop client goes to the location specified and gets the executable, which it installs automatically.

CHAPTER 6

Troubleshooting

This chapter describes some techniques you can use to determine the cause of a problem within an LSF desktop support environment. It also describes how the LSF batch commands behave in an LSF desktop support environment, and how each of the LSF scheduling policies affects LSF desktop support jobs.

Contents

Desktop Server Stops Dispatching Jobs

Follow this procedure if you determine that the desktop server is not dispatching jobs.

If jobs are not being dispatched:

1. Is the desktop server running?
   - ***No***. Make sure that Apache and Tomcat are running, and then start the desktop server.
   - ***Yes***. Continue to the next step.
2. Are Apache and Tomcat running?
   - ***No***. Run *ACH_TOP*`/etc/lsfac_daemons start` to start them. Restart the desktop server.
   - ***Yes***. Continue to the next step.
3. Run *ACH_TOP*`/etc/lsfac_daemons stop` to shut down the desktop server.
4. Shut down and restart Apache and Tomcat.
5. Start the desktop server.
6. Are jobs being dispatched now?
   - ***No***. Contact Platform Technical Support.
   - ***Yes***. Continue processing.

Important With EGO management of LSF desktop support services enabled, you should not use the command lsfac_daemons to start Apache or Tomcat services. Instead, you should use the egosh command to start these services.

The Desktop Client Stops Working

Follow this procedure if you determine that a desktop client stops running jobs.

If a desktop client is not working:

1. Look at http://`LSF desktop support_`*host*/servlet/StatsViewer to see when was the last time the desktop client requested a job or returned status. Has the desktop client logged in within the last polling interval?
   - ***No***. The desktop client has not logged in within the interval specified in the SEDPollInterval. Go to step 4.
   - ***Yes***. The desktop client has logged in within the interval specified in the SEDPollInterval. Continue with step 2.
2. Look at the Job Status page at `http://`*`host_name`*`/servlet/StatsViewer`. Are there jobs waiting to be run?
   - ***No***. There are no jobs to run.
   - ***Yes***. Continue with step 3.
3. Can you ping the desktop server from the desktop client?
   - ***No***. There is a problem with the network.
   - ***Yes***. Continue with step 4.
4. Restart the desktop client by shutting down the service SED and starting it again. Does the desktop client start to run a job?
   - ***No***. Call Platform Technical Support.
   - ***Yes***. The problem is resolved.

Writing LSF desktop support logs to a single directory

You can make it easier to find the Tomcat and Apache log files by choosing to write these files to the directory that contains log files for other LSF desktop support services.

You must restart Apache and Tomcat after changing the configuration.

Configure the Apache log files:

Configuration file	Parameter and syntax	Behavior
`apache/conf/httpd.conf`	Error Log *path_name*/error_log.*host_name*	◆ The Apache error log is written to the specified directory on the specified host
	CustomLog *path_name*/access_log.*host_name* common	◆ The Apache events log is written to the specified directory on the specified host

Configure the Tomcat log files:

Configuration file	Syntax	Behavior

`jakarta-tomcat-4.1.30-LE-jdk14/confserver-noexamples.xml.config`
`jakarta-tomcat-4.1.30-LE-jdk14/confserver.xml`
`jakarta-tomcat-4.1.30-LE-jdk14/webapps/admin.xml`
	<Logger className="org.apache.catalina.logger.FileLogger" directory="*logfile_directory*" prefix="*file_prefix_host_name*" suffix=".txt" timestamp="true"/>	◆ The Tomcat log files are written to the specified directory on the specified host
		◆ When your system has more than one desktop server (MED), adding the host name to the file prefix will prevent log files from being overwritten

Configure the Tomcat shell script:

Shell script	Syntax	Behavior
`jakarta-tomcat-4.1.30-LE-jdk14/bin/catalina.sh`	Line 213 - touch "$CATALINA_BASE"/logs/catalina.out + touch "*path_name*"/catalina.out Lines 225 and 237 - >> "$CATALINA_BASE"/logs/catalina.out 2>&1 & + >> "*path_name*"/catalina.out 2>&1 &	◆ The Tomcat log files are written to the specified directory

Configure the LSF desktop support log directory:

Configuration File	Syntax	Behavior
`AC_TOP/wscache.conf`	wscache log *path_name*	◆ LSF desktop support log files are written to the specified directory

Debugging the Desktop Client

You can place a desktop client in debug mode to log all debug information regarding the desktop client. The desktop client (SED) records connection error and debug messages created during job execution in the Windows application event service. LSF desktop support administrators can easily retrieve these error messages using the remote event viewer or a terminal services session.

LSF desktop support does not write any messages containing passwords or other authentication information to the Windows event service.

To set a desktop client to debug mode:

Configuration file	Parameter and syntax	Default	Behavior
SEDConfig.xml	<SEDDebug> <SEDName>*host_name*</SEDName> </SEDDebug>	Not defined	◆ Sets a single desktop client (SED) to debug mode
	<SEDDebug> <SEDName>*host_name*</SEDName> <SEDName>*host_name*</SEDName> </SEDDebug>	Not defined	◆ Sets multiple desktop clients (SEDs) to debug mode

Debugging a Desktop Application

LSF desktop support traps `stdout` and `stderr` messages in files specified by the `bsub -e`, `-eo`, `-o`, and `-oo` options.

Set a maximum log file size:

To prevent verbose applications from generating large log files, you can set a maximum log file size.

Configuration file	Parameter and syntax	Default	Behavior
`SEDConfig.xml`	<SEDMaxOutputLogSize>*file_size*</SEDMaxOutputLogSize>	Not defined The file size is unlimited	◆ Sets a maximum file size, in KB, for `stderr` and `stdout` log files

Recovering from a Power Outage

Under normal circumstances, even after a power outage, LSF desktop support jobs should continue to run. Follow the procedures listed here to ensure resumption of normal processing.

After a power outage:

Restart the desktop server. In most cases, jobs will continue to run normally within the LSF desktop support.

If the desktop server cannot start:

1. Check the file `sbatchd.log.`*`clustername`* (in the directory specified in LSF_LOGDIR) to see if an event record is corrupted. If an event record is corrupted, the log will point to the corrupted line number, be located in either $*ACH_TOP*`/work/.`*`clustername`*`.sbd/med.events` or $*ACH_TOP*`/work/.`*`clustername`*`.sbd/sbd.events`.
2. If an event record is corrupted, contact Platform Technical Support for assistance.

If a job seems to be ‘stuck’:

1. Issue the `bkill` command to kill the job if you do not want the job to continue. Otherwise, issue the brequeue command to redispatch the job.
2. If `bkill` does not work:
   a. Shut down the desktop server.
   b. Issue the `bkill` command on an LSF host to kill the job.
   c. Restart the desktop server.

Job Blocked at the Desktop Server with Many File Transfers

LSF desktop support supports a maximum of 32 file transfer requests with the `-f` option in the `bsub` command. Specifying more than 32 file transfer requests can cause abnormal behavior.

Workaround

Use zip and unzip commands to reduce the number of file transfer requests.

Example

In the following example, `myjob.exe` requires a total of 66 file transfers: 33 to copy files to the desktop client, and 33 to copy the results from the desktop client.

Incorrect usage

```
bsub -f "data1>data1" ... -f "data33 > data33" -f "result1 <
result1" ... "result33 < result33" myjob.exe
```

Recommended usage

1. Zip the data files together into one file. For example:

   ```
   zip data.zip data1 data2 data3 ...data33
   ```

2. Create a job wrapper that unzips the data files, runs the executable and zips the results. For example, the wrapper `myjob.bat` might look like this:

   ```
   unzip data.zip
   myjob.exe
   zip result.zip result1 result2 result3 ... result33
   ```

3. Submit the job, transferring the data files, the wrapper and the executable to the desktop client, and transferring the zipped results file back from the desktop client. For example:

   ```
   bsub -f "data.zip > data.zip" -f "myjob.bat > myjob.bat" -f
   "myjob.exe > myjob.exe" -f "result.zip < result.zip"
   myjob.bat
   ```

4. When the job is completed, unzip the result file:

   ```
   unzip result.zip
   ```

If you do not have zip and unzip on your system, you can get them from the Internet, and install them on each desktop client as required.

LSF Policies in LSF Desktop Support

Because LSF desktop support runs jobs on desktop clients rather than directly on LSF hosts, some LSF scheduling policies and commands behave differently or are unsupported in LSF desktop support.

Batch commands

Command	Supported in LSF desktop support?	Description
`bacct`	Partially	The data is kept, but items such as queue time are misleading, since the denote only the time before being dispatched to MED.
`bbot`	Yes	Jobs that are pending can be moved using `bbot`. Jobs that are running cannot.
`bchkpnt`	No	
`bclusters`	Yes	LSF desktop support does not affect this command.
`bhist`	Yes	Only displays some v queue and dispatch data. The desktop client that takes the job is logged in `bhist`.
`bhosts`	Yes	Note that this is only 'near' real-time data.
`bhpart`	Not applicable	
`bjobs`	Yes	The host listed is the MED host. The desktop client that actually takes the job is logged in `bjobs using bpost`.
`bkill`	Yes	Signals may be sent but they do not make any sense, since signals are not supported.
`bmgroup`	Not applicable	
`bmig`	Yes	bmig forces LSF desktop support to terminate a job, which is requeued onto another MED host or the same MED host.
`bmod`	Not applicable	None of the post-dispatch options are supported. You can change any parameter provided the job has not been dispatched to LSF desktop support yet.
`bparams`	Yes	
`bpeek`	No	You cannot peed at the output of a job running on a desktop client.
`bpost`	Yes	You can `bpost` and `bread` to LSF desktop support jobs.
`bqueues`	Partially	`bqueues` displays information about LSF queues, but once a job is dispatched to an MED, it is queued on the Web server. The Web server queue is not displayed in bqueues.
`bread`	Yes	
`brequeue`	Yes	`brequeue` forces LSF to kill and reschedule an LSF desktop support job.
`brestart`	No	
`bresume`	No	You cannot suspend or resume an LSF desktop support job.
`bstatus`	Yes	`bstatus` is part of the `bpost`, `bread` command set and is supported for individual LSF desktop support jobs.
`bstop`	No	You cannot suspend or resume an LSF desktop support job.
`bsub`	Yes	See "bsub options" on page 114.
`bswitch`	Yes	`bswitch` can change jobs that have not been dispatched to an MED to another queue. `bswitch` does not operate on any chunk job that has been dispatched (same as LSF).

Command	Supported in LSF desktop support?	Description
`btop`	Yes	Works on all pending jobs.
`bugroup`	Yes	
`busers`	Yes	
`ls*`	Partially	The `ls` commands work only in LSF, not in LSF desktop support.
`x*`	Yes	Graphical interfaces work in accordance with their respective command line batch commands listed above.

bsub options

bsub Option	Supported in LSF desktop support?	Description
`-B` (email upon dispatch)	Partially	The user receives the email when the job is sent to the MED, not when it is dispatched to the desktop client.
`-H` (hold job)	Yes	Even though you cannot stop a job in LSF desktop support, you can submit jobs and keep them in PSUSP state.
`-I` (interactive)	No	
`-K` (wait)	Yes	
`-N` (job report email)	Yes	Works much like LSF, although the user may receive the email before job cleanup occurs.
`-r` (rerunnable)	Partially	LSF desktop support has its own concept of rerunnable that is controlled at the MED/web server. Using the `-r` option allows all jobs to be rerun or moved off of an MED if required.
`-x` (exclusive execution)	Partially	If a single job is sent to LSF desktop support, exclusive works correctly. However, if a chunk job is sent to LSF desktop support, it assumes the entire chunk job is on e job and executes the entire chunk. LSF treats a chunk job as a serial pipeline, while LSF desktop support treats a chunk job as a parallel pipeline (all chunks can run at once).
`-b` (begin time)	Yes	
`-C` (core size limit)	No	
`-c` (cpu time limit)	Not recommended	Due to differences in CPU speeds, it is recommended that you use runlimit (`-W`) instead.
`-D` (data size limit)	No	
`-e` (stderr)	Yes	
`-eo` (stderr)	Yes	
`-E` (pre-execution)	Partially	The pre-execution command is run on the MED host, not on the individual desktop client.
`-f` (file copy)	Yes	Required parameter in LSF desktop support. LSF desktop support also uses this parameter to cache all files. Operators for this parameter are described in the *Platform LSF Desktop Support User's Guide*.
`-F` (file size limit)	No	
`-G` (user group)	Yes	
`-I` (stdin)	No	
`-J` (job array)	Yes	
`-k` (checkpoint)	No	

bsub Option	Supported in LSF desktop support?	Description
-L (login shell)	No	
-m (specified hosts)	Not recommended	You can specify the MED, but items will be set at the queue level, so you should specify a queue instead.
-M (memory limit)	No	Swap limit is supported.
-n (number of processors)	No	
-o (stdout)	Yes	
-oo (stdout)	Yes	
-P (project name)	Yes	
-q (queue)	Yes	
-R (resource requirements)	Partially	Only the resources attributed to the MED host are taken into consideration. Desktop clients do not have resources in LSF. Use -extsched.
-sp (priority)	Yes	Only works on pending jobs.
-S (stack size limit)	No	
-t (terminate time)	Yes	You can use terminate time to kill jobs, but runlimit is better. It is very difficult to guarantee that the job has actually started in LSF desktop support.
-u (email recipient)	Yes	
-v (swap limit)	Yes	
-w (job dependency)	Yes	
-W (run time limit)	Yes	Also indicates the time-out before rerunning a job.
-Zs (spooling)	No	Has no effect on LSF desktop support.
-extsched	Partially	Supports desktop client resource-aware scheduling.

LSF features

LSF Feature	Application to LSF desktop support	Description
Interactive jobs	No	
Fairshare	Partial	Host-level fairshare (i.e. host partitioning) is not supported, but user-level fairshare is supported. During the execution of a job, LSF desktop support reports some resource usage. Once the job completes, resource usage is recorded. This information is used by the fairshare algorithm.
esub	Yes	
eexec	No	
Pre/Post execution	No	
JobStarter	No	
Queues	Yes	All queues types and commands work with LSF desktop support. However, decisions on dispatch to the sbatchd/MED can realistically only be made based on the number of jobs currently being run at any MED. The MED has its own queues, which cannot be controlled.
Resource scheduling	Partial	Resource scheduling is done at the MED level, not at the individual desktop client level. LSF desktop support has its own desktop client resource scheduling.

LSF Feature	Application to LSF desktop support	Description
Resource requirements	Partial	Jobs can be restricted to only run on hosts with available disk and memory.
Resource limits	Yes, limited	Job resource usage can be limited by CPU, memory or time.
Checkpoint, Restart & Migration	No	These do not make sense in an LSF desktop support environment.
Pre-emption	Yes	
Chunk jobs	Yes	
Authentication	Yes	Authentication of nodes is done via SSL protocol, if desired.
User account mapping, environment replication	No	No attempt is mad to duplicate the submission environment. This must be completed by the job itself.
Job controls	No	
Job arrays	Yes	
Parallel jobs	No	
Job email	Yes	
Slot limits	Yes	Slot limits for user, queue and hosts are applicable to LSF desktop support.
Advanced reservation	Partial	Slots are only reserved at the MED level.
Multicluster forward model	Yes	A job can be forwarded to remove clusters.
Multicluster leasing model	Yes	Jobs can be dispatched to MED at remote clusters.
Deadline constraints	Yes	

Index