FLEXlm™

End Users Guide

VERSION 9.0

MARCH 2003

macrovision™

Copyright Notice

© 2003 Macrovision Corporation. All rights reserved.
Macrovision products contain certain confidential information of Macrovision Corporation. Use of this copyright notice is precautionary and does not imply publication or disclosure. No part of this publication may be reproduced, transmitted, transcribed, stored in a retrieval system, or translated into any language in any form by any means without the written permission of Macrovision Corporation.

Trademarks

Globetrotter, Macrovision, FLEX*lm*, FLEX*lock*, FLEX*bill*, Flexible License Manager, and GT*licensing* are registered trademarks or trademarks of Macrovision Corporation.
All other brand and product names mentioned herein are the trademarks and registered trademarks of their respective owners.

Restricted Rights Legend

Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights of Technical Data and Computer Software clause of DFARS 252.227-0713 and FAR52.227-19 and/or applicable Federal Acquisition Regulation protecting the commercial ownership rights of independently developed commercial software.

Printed in the USA.
March 2003

Table of

Contents

Preface . . . . . **ix**

Chapter **1** Introduction and Overview **13**
1.1 Introduction to FLEX*lm* 13
1.2 How to Use This Manual 13
1.2.1 License Administrator 13
1.2.2 End User 14
1.3 FLEX*lm* Components 15
1.3.1 Served Licenses 15
1.3.2 Unserved Licenses 16
1.3.3 Component Overview 17
1.3.4 FLEX*lm* Components Shipped by Your Vendor 19
1.4 The License Request Process 21
1.5 Configuring FLEX*lm* 21
1.6 Getting Started Checklist 22
1.6.1 Installing Licensed Software 22
1.6.2 Notes for End Users 23

Chapter **2** License File Basics **25**
2.1 Specifying Location of the License File 25
2.1.1 Setting the Path with an Environment Variable 27
2.2 License File Format Overview 28
2.3 Types of License Files 29
2.3.1 Floating (Concurrent) Licenses 29
2.3.2 Node-Locked Licenses 29
2.3.3 Mixed Node-Locked and Floating Licenses 30

Chapter **3** Managing Licenses from Multiple Vendors **31**
3.1 Overview of Multiple License Management Strategies 31
3.2 Multiple Machines 32
3.3 One Machine with Multiple License Servers 33
3.4 One Machine with One License Server and Multiple License Files 34
3.5 Managing Multiple License Files 35

3.6 Additional Considerations 37
3.6.1 Combining license files 37
3.6.2 Version Component Compatibility 40

Chapter **4** Selecting Server Machines **41**
4.1 Resources Used by the Server 41
4.1.1 Sockets 41
4.1.2 CPU Time 42
4.1.3 Disk Space 42
4.1.4 Memory 42
4.1.5 Network Bandwidth 43
4.2 Remote Mounted Disks 43
4.3 Redundant License Servers 43
4.3.1 Redundancy via License-File List 44
4.3.2 Three-Server Redundancy 45
4.3.3 Comparing Three-Server to License-File List 45
4.4 Counted vs. Uncounted Licenses 46

Chapter **5** The Options File **47**
5.1 Creating an Options File 47
5.2 Options File Syntax 48
5.2.1 BORROW_LOWWATER 52
5.2.2 DEBUGLOG 53
5.2.3 EXCLUDE 53
5.2.4 EXCLUDE_BORROW 54
5.2.5 EXCLUDEALL 55
5.2.6 GROUP 55
5.2.7 HOST_GROUP 56
5.2.8 INCLUDE 56
5.2.9 INCLUDE_BORROW 57
5.2.10 INCLUDEALL 58
5.2.11 LINGER 58
5.2.12 MAX 59
5.2.13 MAX_OVERDRAFT 60
5.2.14 NOLOG 60
5.2.15 REPORTLOG 60
5.2.16 RESERVE 61
5.2.17 TIMEOUT 62
5.2.18 TIMEOUTALL 63
5.3 How the Vendor Daemon Uses the Options File 63
5.4 Rules of Precedence in Options Files 63

5.5 Options File Examples 64
5.5.1 Simple Options File Example 64
5.5.2 Limiting Access for Multiple Users 65
5.5.3 EXCLUDE Example 65
5.5.4 INCLUDE Example 66

Chapter **6** The License Manager Daemon **67**
6.1 lmgrd Command-Line Syntax 67
6.2 Starting the License Manager Daemon on UNIX Platforms 69
6.2.1 Manually 69
6.2.2 Automatically 70
6.3 Starting the License Manager Daemon on Windows 71
6.3.1 Manually from the Command Line 71
6.3.2 Configuring the License Manager as a Windows Service . . . 72
6.3.3 Manually from LMTOOLS 72
6.3.4 Automatically at System Start Up 74

Chapter **7** License Administration Tools **77**
7.1 Running Administration Tools 78
7.2 Universal lmutil Arguments 79
7.3 lmborrow 79
7.4 lmdiag 83
7.5 lmdown 84
7.6 lmhostid 85
7.7 lminstall 87
7.8 lmnewlog 88
7.9 lmpath 88
7.10 lmremove 90
7.11 lmreread 91
7.12 lmstat 93
7.13 lmswitch 95
7.14 lmswitchr 96
7.15 lmver 97
7.16 License Administration Tools—LMTOOLS for Windows 97
7.16.1 Configuration Using License File 98
7.16.2 Configuration Using Services 98

Chapter **8** Mobile Licensing **99**
8.1 Node-Locked to a Laptop Computer 99
8.2 Node-locked to a FLEX*id* (Windows Only) 99
8.3 Node-Locked to a FLEX*id* with FLOAT_OK (Windows Only) . . 100
8.3.1 Initiating FLEX*id* with FLOAT_OK 100

8.3.2 Returning a FLEX*id* with FLOAT_OK License 101
8.3.3 FLEX*id* with FLOAT_OK Example 101
8.4 License Borrowing with BORROW 102
8.4.1 Initiating License Borrowing 103
8.4.2 Borrowing a License 104
8.4.3 Support for License Borrowing 105
8.5 Node-locked to a User Name 106
8.6 Fulfilled from a Prepaid License Pool 106

Appendix **A** Hostids for FLEX*lm*-Supported Machines **107**
A.1 Hostid Formats 107
A.2 Expected FLEX*lm* Hostids 108
A.3 Special FLEX*lm* Hostids 109

Appendix **B** License File Format **113**
B.1 License File Syntax 114
B.1.1 Sample License File 114
B.1.2 SERVER Lines 114
B.1.3 VENDOR Lines 115
B.1.4 USE_SERVER Line 117
B.1.5 FEATURE/INCREMENT Lines 117
B.1.6 PACKAGE Lines 124
B.1.7 UPGRADE Lines 127
B.2 Decimal Format 127
B.3 License File Order 128

Appendix **C** Troubleshooting Guide **129**
C.1 General Debugging Hints 129
C.2 FLEXLM_DIAGNOSTICS 130
C.2.1 Level 1 Content 130
C.2.2 Level 2 Content 131
C.2.3 Level 3 Content (FLEX*lm* v6.0+ only) 132
C.3 FLEX*lm* Troubleshooting List 132
C.3.1 Problem Description Format 132
C.3.2 Hostid Problems 133
C.3.3 Connection Problems 134
C.3.4 Other FLEX*lm*-Licensed Application Problems 135
C.3.5 Other Server Problems 136

Appendix **D** FLEX*lm* Environment Variables **139**
D.1 How to Set Environment Variables 139

D.1.1 Registry 139
D.1.2 Precedence 139
D.2 Environment Variables 140

Appendix **E** Frequently Asked Questions **143**
E.1 License File Questions 143
E.1.1 I've received FLEX*lm* license files from two different companies. Do I have to combine them? 143
E.1.2 When is it recommended to combine license files? 143
E.2 FLEX*lm* Versions 143
E.2.1 I have products from several companies at various FLEX*lm* version levels. Do I have to worry about how these versions work together? 143
E.2.2 I've received a new copy of a product from a vendor, and it uses a new version of FLEX*lm*. Is my old license file still valid? 144
E.2.3 I've received a new license file, and the format is different from the old one I had from the same vendor? Why? Are they compatible? 144
E.3 Using FLEX*lm* 144
E.3.1 Does FLEX*lm* work across the Internet? 144
E.3.2 Does FLEX*lm* work with Internet firewalls? 144
E.3.3 If my FLEX*lm*-licensed application dies, does the server free the license? 145
E.3.4 What happens when the license server dies? 145
E.3.5 How do you tell if a TCP/IP port is already in use? 145
E.3.6 Does FLEX*lm* require root permissions? 145
E.3.7 Is it OK to run lmgrd as "root" (UNIX only)? 146
E.3.8 Does FLEX*lm* licensing impose a heavy load on the network? 146
E.3.9 Does FLEX*lm* work with NFS? 146
E.3.10 Does FLEX*lm* work with ATM, ISDN, Token-Ring, etc. . 146
E.3.11 Does FLEX*lm* work with subnets, fully qualified names, multiple domains, etc.? 146
E.3.12 Does FLEX*lm* work with NIS and DNS? 147
E.3.13 We are using FLEX*lm* over a wide-area network. What can we do to improve performance? 147
E.4 Windows Questions 148
E.4.1 What Windows Platforms are supported? 148
E.4.2 Will the Server run on Windows 95? 148

Appendix **F** FLEX*lm* Error Codes **149**
F.1 Error Message Format 149
F.1.1 Format 1 (short): 149
F.1.2 Format 2 (long—version 6.0+): 150
F.2 Error Code Descriptions 150

Appendix **G** The Report Log File **157**
G.1 Managing Report Log Output 157
G.2 Enabling Report Log Output for a Vendor Daemon 158
G.3 Redirecting Report Log Output for a Vendor Daemon 158

Appendix **H** The Debug Log File **159**
H.1 Managing Debug Log Output 159
H.1.1 Capturing Debug Log Output for a License Server 159
H.1.2 Capturing Debug Log Output for a Particular Vendor Daemon 160
H.1.3 Redirecting Debug Log Output for a Running Vendor Daemon 160
H.1.4 Limiting Debug Log Output for a Vendor Daemon 160
H.2 Debug Log Messages 160
H.2.1 Informational Messages 161
H.2.2 Configuration Problem Messages 163
H.2.3 Daemon Software Error Messages 165

Appendix **I** FLEX*lm* Versions **167**
I.1 Version Compatibility and Components 167
I.2 How to Tell the License File Version 167
I.3 Version Summary 168

Index **173**

Preface

Welcome to FLEX*lm*®, the *de facto* standard network license manager used by over 2000 leading software vendors to control the use of their software products. If you are a system administrator or user, chances are one or more of the products currently on your network is licensed by FLEX*lm*.

About This Manual

This manual explains FLEX*lm* for administrators and end users and describes how to use the tools which are part of the standard FLEX*lm* distribution kit. Macrovision Corporation also provides the SAM*suite* and SAM*report* asset management tools for more advanced license server control and reporting. SAM*wrap* is another Macrovision product that allows end users to provide FLEX*lm* license management for applications without embedded FLEX*lm* license management. Please contact Macrovision Corporation by email at info@globes.com or on the Internet at www.macrovision.com for more information about SAM*suite*, SAM*report*, or SAM*wrap*.

Keep in mind that certain topics (such as password encryption) are vendor-specific and proprietary so they are not documented in any detail. Also, because FLEX*lm* does not enforce a particular licensing strategy, each vendor's implementation has subtle differences. If you do not find out what you need to know here, contact your vendor's technical support group.

Versions of FLEX*lm*

This manual covers features of interest to license administrators and end users in FLEX*lm* versions 5.0 through 9.0. The text presents FLEX*lm* v9.0 behavior and functionality. Differing behavior and functionality, if any, between the current and past versions for a particular feature is noted at the end of the its section. Additionally, Appendix I, "FLEXlm Versions," covers version differences in detail.

Related Documents from Macrovision

The *SAMsuite Users Guide* describes the SAM*suite* license administration tool for managing FLEX*lm*-enabled applications, the *SAMreport Users Guide* describes the SAM*report* license usage reporting tool, and the *SAMwrap Users Guide* describes SAM*wrap* for administrators who wish to provide FLEX*lm* license management for applications shipped without embedded FLEX*lm* license management.

The *FLEXlm Programmers Guide* and *FLEXlm Reference Manual* are for programmers responsible for incorporating FLEX*lm* into their products.

Typographic Conventions

The following typographic conventions are used in this manual:

- The first time a new term is used it is presented in *italics*.
- Commands and path, file, and environment variable names are presented in a `fixed_font`.
- Other variable names are in an *`italic_fixed_font`*.
- API function calls are in a sans-serif font.

Contacting Technical Support

Technical Support is available to customers and prospects with current support contracts. Please include the following information with your inquiry:

- Product Name
- Product Version
- Operating System Name and Version

Support Centers

Contact the support center in your area:

- North America
  - Telephone: (408) 969-5441, 8:30am-5:00pm PST
  - Email: support@macrovision.com
  - Fax: (408) 445-7760
  - Online: http://www.globes.com
  - GTwebSupport
- Europe

- Telephone: +44 (0)870 873 6301, 8:30am-5:00pm GMT
- Email: support@globes-europe.com
- Fax: +44 (0)1928 706 329
- Online: http://www.globes.com/europe

• Japan
- Telephone: +81-35774-6253, 9:30am-6:00pm JST
- Email: support-japan@globes.com
- Fax: +81-35774-6269

All other customers and prospects please contact the North American office.

Chapter 1

Introduction and Overview

This chapter explains the basics of floating (network) licensing and gives a quick overview of the components of FLEX*lm*. It explains where license administrators have control and where end users have control. Section 1.6, “Getting Started Checklist,” tells both license administrators and end users how to start managing FLEX*lm*.

1.1 Introduction to FLEX*lm*

FLEX*lm* is the most popular license manager used in the software industry. FLEX*lm* is best known for its ability to allow software licenses to be available (or float) anywhere on a network, instead of being tied to specific machines. Floating licensing benefits both users and license administrators. Users make more efficient use of fewer licenses by sharing them on the network. License administrators control who uses the licensed application and the machine(s) where the licenses are available. See Section 2.3, “Types of License Files,” for details about the different licensing models supported by FLEX*lm*.

1.2 How to Use This Manual

This manual is written for two different audiences:

- Administrators of FLEX*lm* licenses and license servers
- End users of FLEX*lm*-enabled applications

1.2.1 License Administrator

If you are a license administrator, read these chapters:

Chapter:	**Explains:**
Preface	Overview of this manual.

Chapter:	Explains:
Chapter 1, ”Introduction and Overview”	FLEX*lm* basics: license manager and vendor daemons; the license file; configuring FLEX*lm*; the license request process.
Chapter 2, ”License File Basics”	License file basics; setting the path at start-up; different types of licensing policies.
Chapter 3, ”Managing Licenses from Multiple Vendors”	Using license files from multiple software vendors.
Chapter 4, ”Selecting Server Machines”	Selecting which machines run the license servers; resources required by the servers; multiple servers; quorums; redundant license servers.
Chapter 5, ”The Options File”	Creating and editing the options file.
Chapter 6, ”The License Manager Daemon”	Using the license manager daemon, `lmgrd`.
Chapter 7, ”License Administration Tools”	Managing FLEX*lm* using Macrovision-supplied utilities.
Chapter 8, ”Mobile Licensing”	Licensing to allow working on a computer disconnected from the license server.

In addition, refer to:

- Appendix B, “License File Format”
- Appendix C, “Troubleshooting Guide”
- Appendix E, “Frequently Asked Questions”

1.2.2 End User

If you are an end user, read these chapters:

Chapter:	Explains:
Preface	Overview of this manual.

Chapter:	Explains:
Chapter 1, ”Introduction and Overview”	FLEX*lm* basics: license and vendor daemons; the license file; configuring FLEX*lm*; the license request process.
Chapter 2, ”License File Basics”	License file basics; setting the path at start-up; different types of licensing policies.
Chapter 8, ”Mobile Licensing”	Licensing to allow working on a computer disconnected from the license server.

In addition, refer to:

- Appendix C, “Troubleshooting Guide”

1.3 FLEX*lm* Components

FLEX*lm* components are organized based on the license model used by your software vendor. License models are classified according to their requirement for a license server:

- Licenses are served by a license server. This is commonly referred to as a *served* license model. License files supplied by your software vendor that include SERVER, VENDOR, and optionally, USER_SERVER lines require a license server. See Section 1.3.1, “Served Licenses,” for details.
- Licenses are not served by a license server but are available directly to the application. This is commonly referred to as an uncounted, or *unserved* license model. See Section 1.3.2, “Unserved Licenses,” for details.

1.3.1 Served Licenses

For served licenses, there are four required FLEX*lm* components:

- The FLEXlm-Licensed Application, with the FLEX*lm* static client library linked into it.
- The License Manager Daemon (lmgrd).
- The Vendor Daemon, which, along with the license manager, `lmgrd`, comprises the license server.
- The License File.

In addition to these four components, there are three optional components:

- Debug Log File — created and written by `lmgrd`.
- Report Log File — created and written by the vendor daemon for use by SAM*report*.
- End-User Administration Options File — file created and maintained by the end user.

Figure 1-1 shows the relationship these components have to one another. See Section 1.3.3, “Component Overview” for a description of each component.

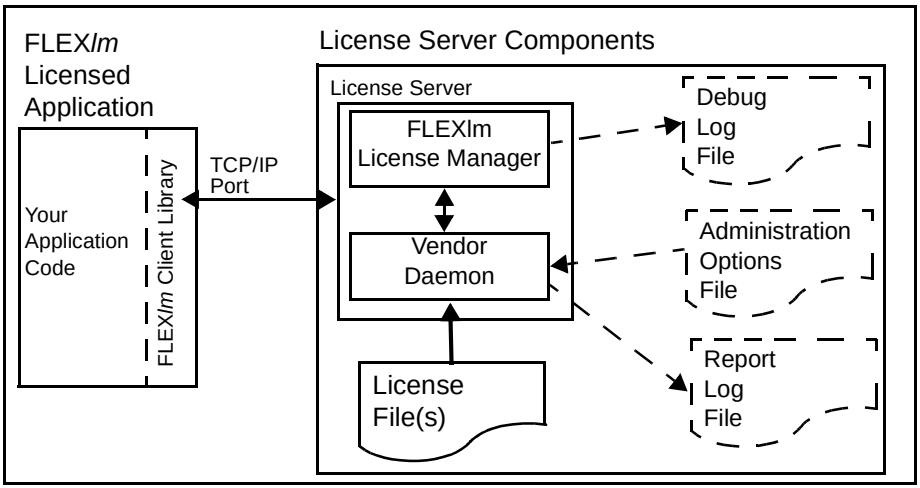


Figure 1-1: FLEXlm Served License Component Model

Typically, the license server components reside on a machine in the network but can optionally reside on the same machine as the licensed application.

The three optional files: Debug Log, Administration Options, and Usage Log files are configured by the end user.

1.3.2 Unserved Licenses

License models that do not need a license server have two components:

- The FLEXlm-Licensed Application, with the FLEX*lm* static client library linked into it.
- The License File.

Figure 1-2 shows this model. See Section 1.3.3, “Component Overview” for a description of each component.

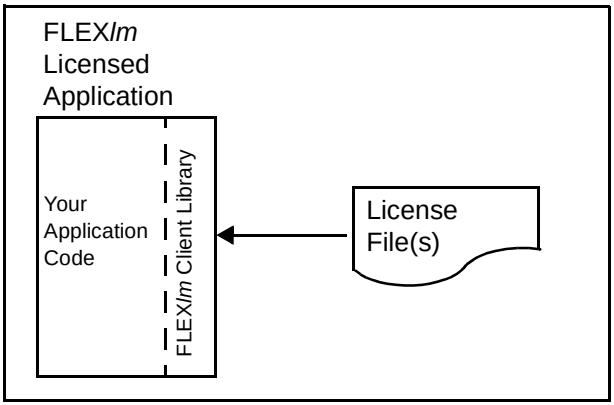


Figure 1-2: FLEXlm Unserved License Component Model

The licensed application and license certificate reside on the same machine.

1.3.3 Component Overview

THE LICENSE MANAGER DAEMON (LMGRD)

The *license manager daemon* (`lmgrd`) handles the initial contact with the FLEX*lm*-licensed applications, passing the connection on to the appropriate vendor daemon. It also starts and restarts the vendor daemons.

Chapter 6, “The License Manager Daemon,” contains information for configuring and starting the license manager daemon in your environment.

THE VENDOR DAEMON

In FLEX*lm*, counted (floating) licenses are granted by processes running on the system. There is one process for each vendor who has a FLEX*lm*-licensed product on the network. This process is called the *vendor daemon*. The vendor daemon keeps track of how many licenses are checked out, and who has them.

FLEX*lm*-licensed applications communicate with the vendor daemon, through TCP/IP network communications. The FLEX*lm*-licensed application and the daemon processes (the *license server*) can run on separate machines on your network, across any size wide-area network. Also, the format of the traffic between the FLEX*lm*-licensed application and the vendor daemon is machine independent, allowing for heterogeneous networks. This means the license server and the computer running an application can be either different hardware platforms or even different operating systems (Windows and UNIX, for example).

If the vendor daemon terminates for any reason, all users lose their licenses (though this does not mean the applications suddenly stop running). Users normally regain their license automatically when `lmgrd` restarts the vendor daemon, though they may exit if the vendor daemon remains unavailable. This behavior is defined by the vendor in the FLEX*lm*-licensed application.

The License File

Licensing data is stored in a text file called the *license file*. The license file is created by the software vendor, and edited and installed by the license administrator. It contains information about the server machines and vendor daemons, and at least one line of data (called FEATURE or INCREMENT lines) for each licensed product. Each FEATURE line contains a license key or signature based on the data in that line, the hostids specified in the SERVER line(s), and other vendor-specific data. FLEX*lm*-licensed applications that are licensed with a node-locked, uncounted license need only read a valid license file to run—they do not need a license server.

Most applications have an expected location for the license file, documented by that application. Override this location by setting the environment variable `LM_LICENSE_FILE` to point elsewhere, or by following instructions supplied with the licensed application. If your site has software from multiple vendors with incompatible license files (due to different sets of servers), keep the data in separate files and set the `LM_LICENSE_FILE` variable to reference multiple files.

It is recommended that you keep a link (on UNIX), a shortcut (on Windows), or copy of the license file in the vendor's expected license location, so that users do not need to set `LM_LICENSE_FILE` to run their applications. For license files containing counted (floating) licenses, it is recommended to place a USE_SERVER line directly after the SERVER line. For details, see Appendix B, "License File Format." See also Appendix D, "FLEXlm Environment Variables."

The FLEX*lm*-Licensed Application

The application program using FLEX*lm* is linked with the program module (called the FLEX*lm* client library) that provides the communication with the license server. During execution, the FLEX*lm*-licensed application communicates with the vendor daemon to request a license.

DEBUG LOG FILE

The debug log file contains status and error messages useful for debugging the license server. Some of the debug log output describes events specific to `lmgrd` and some of the debug log output describes events specific to each vendor daemon. See Appendix H, “The Debug Log File,” for more information about this file.

END-USER ADMINISTRATION OPTIONS FILE

The end-user administration options file allows the end-user license administrator to control various operating parameters of FLEX*lm* that remain within the license rights granted to them by the vendor. Specifically, the license administrator can:

- Allow the use of features
- Deny the use of features
- Reserve licenses
- Restrict the number of licenses available
- Control the amount of debugging information logged about license usage
- Enable a report log file

See Chapter 5, “The Options File.” for details on how to create this file.

REPORT LOG FILE

The report log file contains feature usage information and is generated by the vendor daemon. Report log output is encrypted and is consumed by reporting products such as SAM*report* and FLEX*bill*. See Appendix G, “The Report Log File,” details regarding this file.

1.3.4 FLEX*lm* Components Shipped by Your Vendor

This section provides a list of FLEX*lm* components that may be supplied to you by your software vendor along with the FLEX*lm*-licensed application. Table 1-1 provides a general list and may differ from that actually supplied. Your vendor is the definitive source for the list of FLEX*lm* components that are shipped.

Table 1-1:FLEXlm Components for End-User Distribution

Component	Description
Windows and UNIX Common Components	
`lmgrd.exe` `lmgrd` (UNIX)	Required for served license models. The license manager and vendor daemons, where *vendor* is the name of your vendor.
`vendor``.exe` *`vendor`* (UNIX)	
`lmutil.exe` `lmutil` (UNIX)	Optional component. FLEX*lm* utilities program. Used for license server management and administration. Also available from www.macrovision.com.
Additional Windows Components	
`lmtools.exe`	Optional component. Graphical user interface for FLEX*lm* utilities, `lmutil.exe`. Also available from www.macrovision.com.
`lmgr8b.dll`	Required if the FLEX*lm*-licensed application is dynamically linked.This library provides the standard FLEX*lm* client library routines.
`fldata.ini` and `flrsrc.dll`	Required for applications which support “try before you by”, FLEX*lock* licensing.
`flexid.exe` or `flexidsilent.exe`	Required if vendor supports FLEX*id* dongle hostids. `flexid.exe` is used to install FLEX*id* drivers for use with FLEX*id* dongles. Alternatively, the vendor may supply `flexidsilent.exe` to silently install the drivers as part of the software installation.

1.4 The License Request Process

When you run a counted FLEX*lm*-licensed application, the following occurs:

1. The license module in the FLEX*lm*-licensed application finds the license file, which includes the host name of the license server machine and TCP/IP port number of the license manager daemon, `lmgrd`.
2. The FLEX*lm*-licensed application establishes a connection with the license manager daemon (`lmgrd`) and tells it what vendor daemon it needs to talk to.
3. `lmgrd` determines which machine and TCP/IP port correspond to the master vendor daemon and sends that information back to the FLEX*lm*-licensed application.
4. The FLEX*lm*-licensed application establishes a connection with the specified vendor daemon and sends its request for a license.
5. The vendor daemon checks in its memory to see if any licenses are available and sends a grant or denial back to the FLEX*lm*-licensed application.
6. The license module in the application grants or denies use of the feature, as appropriate.

Uncounted features (where the license count is 0) do not require a license server, and the FLEX*lm* client library routines in the application grant or deny usage based solely upon the contents of the license file.

1.5 Configuring FLEX*lm*

Most of the parameters of FLEX*lm* are configurable by the license administrator. The license administrator can set the:

- Location of the license file (though it's recommended that a copy or link of the license remains at the location where the application expects it)
- Location of all executables
- Location of all log files
- TCP/IP port number used by the license manager process, `lmgrd`

In addition, the license administrator can reserve licenses for specific users, machines, or groups, and control other license-related options. Changing parameters is discussed in Chapter 5, "The Options File."

Note: Refer to your vendor's documentation before attempting to change file names, locations, or contents.

1.6 Getting Started Checklist

The following sections provide a quick overview of how to set up and use licensing for FLEX*lm*-licensed products.

1.6.1 Installing Licensed Software

As a license administrator you are responsible for setting up licensing on your system or network. This section tells you how to do that. If you are an end user of the application and you are not involved in installing it, then go to Section 1.6.2, “Notes for End Users.”

Remember that the installation guide for your application software is the final word on installing and configuring FLEX*lm*. Generally, however, installing FLEX*lm* licensing requires the following steps:

1. Select your license server machines and get their hostids. See Chapter 4, “Selecting Server Machines,” and Appendix A, “Hostids for FLEXlm-Supported Machines.”
2. Give the hostids to your software vendor and get a license file (or the data to enter in the license file) in return.
3. Consider how to manage license files from multiple vendors. See Chapter 3, “Managing Licenses from Multiple Vendors.”
4. Determine if an options file is desired, and if so, set it up.
5. Determine where to install the FLEX*lm* utility programs such as `lmgrd` and `lmutil` (`lmstat`/`lmdown`/etc.) and install them, unless your vendor’s installation script does so for you.
6. Start `lmgrd` (the license manager daemon) manually; you may also want to set it up to start automatically at boot time. See Chapter 6, “The License Manager Daemon.”

These steps are discussed briefly below.

License Server machine and Hostids

Before running any FLEX*lm*-licensed application using floating licenses, you first need to set up your license server machine (or machines). You must select which machine or machines to run your license servers on and provide the hostids of those machines to your software vendor. For pointers on selecting your server machine, see Chapter 4, “Selecting Server Machines.”

Get the hostid of the server machine by running FLEX*lm*'s `lmhostid` utility on that machine. If you don't have `lmhostid`, get the hostid of your machine by using the appropriate command as described in Appendix A, "Hostids for FLEXlm-Supported Machines."

Using the hostid of your server machines, your vendor issues a license file to you that enables the application software.

License Files and lmgrd

Once you have received a license file from your vendor, you must install it on your system and start up the license manager daemon, `lmgrd`.

- Your software vendor may have selected a default location for your license file. If not, use any location you wish. For more details, see Chapter 2, "License File Basics."
- To start `lmgrd` automatically at boot time, you have to modify your system files (UNIX) or use LMTOOLS (Windows). For details, see Section 2.2, "License File Format Overview."

Administration Tools

Macrovision supplies administration tools to your software vendor. The vendor usually includes these utilities with their product. Download the latest version from www.macrovision.com. See Chapter 7, "License Administration Tools," for more information about how to use the FLEX*lm* utilities.

Options Files

The options file controls various options such as reservations and timeouts of licenses. Most users run without an options file, but you may decide you want to use some options. For example, many administrators use an option to limit the quantity and content of logged messages. To set up an options file, see Chapter 5, "The Options File."

1.6.2 Notes for End Users

As a user of a FLEX*lm*-licensed application, you may need to know a few things to use the system effectively. The main things you need to know are:

- How to tell an application which license file to use
- How to query the system to find out who is using a license

How to Specify a License File Location

The license file determines what features are available to a FLEX*lm*-licensed application. It also contains information telling the application how to connect to the license server.

For information about the standard way of specifying the location of a license file for an application, see Chapter 2, “License File Basics.”

Getting Information About Licenses

To find out who is using a license run `lmstat`, described in Chapter 7, “License Administration Tools.”

Chapter 2

License File Basics

The license file contains information required by FLEX*lm* to manage licenses for a FLEX*lm*-licensed application. This information includes:

- License server names and hostids
- Vendor names and paths to vendor daemon executables
- Feature information

In general, the license file, or a copy of it, must be accessible to every machine that runs a FLEX*lm*-licensed application, and to each machine designated as a license server.

2.1 Specifying Location of the License File

Software vendors often recommend a specific location for your license file. If you are running the application on multiple machines, you have these options for making your licenses available on all the machines:

- Place the license file in a partition which is available to all machines in the network that need the license file.
- Copy the license file to all of the machines where it is needed.
- Set the `LM_LICENSE_FILE` or *VENDOR*_LICENSE_FILE (where *VENDOR* is the vendor daemon name) environment variable to `port@host`, where `host` and `port` come from the SERVER line in the license file. Alternatively, if the license file SERVER line specifies a TCP/IP port in the default port range (27000-27009) or does not specify a port (thereby allowing the system to choose one from the default range), use the shortcut specification, `@host`. Table 2-1 shows some examples of LM_LICENSE_FILE and *VENDOR*_LICENSE_FILE environment variable settings.

Table 2-1:Environment Variable Specification Examples

SERVER Line	**LM_LICENSE_FILE or *VENDOR*_LICENSE_FILE Setting**
`SERVER myserver 17007ea8 \ 40000` where: • *host* = `myserver` • *port* = `40000`	`40000@myserver`
`SERVER myserver 17007ea8 \ 27001` where: • *host* = `myserver` • *port* = `27001`, within the default range	`@myserver`
`SERVER myserver 17007ea8` where: • *host* = `myserver` • *port* = none specified, uses a default TCP/IP port number in the range of 27000-27009	`@myserver`

- On Windows, if the application cannot find the license file, the user is presented with a dialog that asks the user to specify the license file location, the license server, or license fulfillment from the internet.

Since the vendor daemon keeps track of license usage, and since the license file contains encrypted data to protect it against modification, you may move and copy the license file as much as necessary.

For counted licenses, no matter which option you choose, you must first copy `lmgrd` and the vendor daemon to a location accessible by the licensed application.

2.1.1 Setting the Path with an Environment Variable

Most applications specify a location where they expect to find the license file and install it automatically. However, you can change the license file location by setting the `LM_LICENSE_FILE` environment variable. For example, in the C shell:

```
setenv LM_LICENSE_FILE license_file_list
```

In the Korn and Bourne shells:

```
LM_LICENSE_FILE=license_file_list
export LM_LICENSE_FILE
```

On Windows 95/98/ME, add the following line to `C:\autoexec.bat`:

```
SET LM_LICENSE_FILE=license_file_list
```

On Windows NT/2000/XP, use the System Control Panel to change the global environment, adding *`license_file_list`* to `LM_LICENSE_FILE`.

Wherever *`license_file_list`* is specified it can consist of the following components:

- the full path to the license file
- a directory containing one or more license files with a `.lic` extension
- a *`port@host`* setting, where *`port`* and *`host`* are the TCP/IP port number and host name from the SERVER line in the license file. Alternatively, use the shortcut specification, *`@host`*, if the license file SERVER line uses a default TCP/IP port or specifies a port in the default port range (27000-27009)

Applications accept an environment variable (or Windows Registry) named *`VENDOR`*`_LICENSE_FILE`, where *`VENDOR`* is the vendor daemon name, e.g., `GSI_LICENSE_FILE`. This environment variable's scope is limited to just those applications from vendor *`VENDOR`*.

With `lmgrd` and `lmutil(lmstat, lmdown`, etc.), the `-c` option overrides the setting of the `LM_LICENSE_FILE` environment variable.

Note: Some applications do not recognize the `LM_LICENSE_FILE` environment variable. FLEX*lm*-enabled Java applications, in particular, do not recognize it.

SEE ALSO

- Section 3.5, "Managing Multiple License Files," for more information about `LM_LICENSE_FILE`.
- Appendix D, "FLEXlm Environment Variables"

FLEXLM VERSION NOTES

- Default TCP/IP port number range introduced in v6.0 of `lmgrd`.
- License file location dialog introduced in v6.0 of the FLEXlm client library.

2.2 License File Format Overview

License files usually begin with a SERVER line (or three lines for three-server redundant servers) followed by one or more VENDOR lines, followed by one or more FEATURE or INCREMENT lines. In some cases the license file requires no SERVER line and no VENDOR line.

You can modify these elements in the license file:

- Host names on the SERVER line(s)
- TCP/IP port numbers on the SERVER line(s)
- Paths on the VENDOR line(s)
- Options file paths on the VENDOR line(s)
- Optional TCP/IP port numbers on the VENDOR line(s) (for firewall support only)
- USE_SERVER line
- Values in *`keyword=value`* pairs on FEATURE lines, if *`keyword`* is specified in lowercase

SEE ALSO

- Section 4.3, “Redundant License Servers”
- Section 4.4, “Counted vs. Uncounted Licenses,”
- Appendix B, “License File Format,” for details on each one of these components.

FLEXLM VERSION NOTES

- USE_SERVER introduced in the v5.0 FLEX*lm* client library.
- VENDOR lines are known as DAEMON lines in the pre-v6.0 `lmgrd` and vendor daemon.

2.3 Types of License Files

License files are created by the software vendor. License files specify floating (concurrent) usage, node-locked (both counted and uncounted), or any combination of floating, counted, and uncounted.

2.3.1 Floating (Concurrent) Licenses

A *floating license* means anyone on the network can use the FLEX*lm*-licensed application, up to the limit specified in the license file (also referred to as *concurrent usage* or *network licensing*). Floating licenses have no hostids on the individual FEATURE lines. Floating licenses requires an `lmgrd` and a vendor daemon to be running to count the concurrent usage of the licenses.

An example of a license file that provides floating licenses is:

```
SERVER lulu 17007ea8
VENDOR sampled
FEATURE f1 sampled 1.00 1-jan-2005 2 SIGN=signature1
FEATURE f2 sampled 1.00 1-jan-2005 6 SIGN=signature2
FEATURE f3 sampled 1.00 1-jan-2005 1 SIGN=signature3
```

This license file specifies that two licenses for feature “f1,” six licenses for feature “f2,” and one license for feature “f3” are available anywhere on the network that can access the license server “lulu.” `lmgrd` uses one of the default FLEX*lm* ports.

2.3.2 Node-Locked Licenses

Node-locking means the licensed software can only be used on one machine or a set of machines. A node-locked license has a hostid on any FEATURE line that is node-locked to a particular host. There are two types of node-locked licenses; uncounted and counted.

If the number of licenses is set to 0 (or `uncounted`), then the license is uncounted and unlimited use is permitted on the specified machine. This configuration does not require an `lmgrd` or a vendor daemon because it is not going to count the concurrent usage of the features.

The following license file allows unlimited usage of feature “f1” on the machines with hostids of “17007ea8” and “1700ab12”:

```
FEATURE f1 sampled 1.000 1-jan-2005 uncounted SIGN=signature1 \
        HOSTID=17007ea8
FEATURE f1 sampled 1.000 1-jan-2005 uncounted SIGN=signature2 \
        HOSTID=1700ab12
```

Alternately, these two FEATURE lines could have been issued by your software vendor with a *hostid list*:

```
FEATURE f1 sampled 1.000 1-jan-2005 uncounted SIGN=signature \
        HOSTID="17007ea8 1700ab12"
```

If these were the only FEATURE lines in this license file, no `lmgrd` daemon is necessary and you do not need to start one.

The following license file provides three licenses for feature “f1”, locked to the machine with hostid “1300ab43.” Since the license server and licenses are locked to the same machine, the daemons run on the same machine that runs the licensed application.

```
SERVER lulu 1300ab43 1700
VENDOR sampled /etc/sampled
FEATURE f1 sampled 1.00 1-jan-2005 3 SIGN=signature \
        HOSTID=1300ab43
```

2.3.3 Mixed Node-Locked and Floating Licenses

Uncounted node-locked and concurrent usage licenses can be mixed in the same license file.

The following license file allows unlimited use of feature “f1” on machines “17007ea8” and “1700ab12,” while allowing two other licenses for feature “f1” to be used anywhere else on the network:

```
SERVER lulu 17001234 1700
VENDOR sampled C:\flexlm\sampled.exe
FEATURE f1 sampled 1.00 1-jan-2005 uncounted SIGN=signature1 \
        HOSTID=17007ea8
FEATURE f1 sampled 1.00 1-jan-2005 uncounted SIGN=signature2 \
        HOSTID=1700ab12
FEATURE f1 sampled 1.00 1-jan-2005 2 SIGN=signature3
```

This configuration requires an `lmgrd` and a vendor daemon because the concurrent usage of the two licenses on the third FEATURE line is counted.

Chapter 3

Managing Licenses from Multiple Vendors

Since more than 2500 vendors have chosen FLEX*lm* as their license manager, chances are good that you have to administer FLEX*lm* licenses from more than one vendor.

3.1 Overview of Multiple License Management Strategies

When you are running FLEX*lm*-licensed products from multiple vendors, you may need to take steps to prevent licensing conflicts during installation. There are several strategies to accomplish this, of which three are presented here:

- Multiple machines, each running one `lmgrd`, one vendor daemon, and using one license file.
- One license server machine running multiple `lmgrd`s, each of which running one vendor daemon and using one license file.
- One license server machine running one `lmgrd`, multiple vendor daemons each of which using its own license file. License files share a common directory.

These strategies are ordered from most to least independence among vendors. In the first option mentioned above, you have the most license server machines to monitor; in the third option you have only one server and one license file to administer. Each of these three strategies is described in detail in the following sections. Variations are mentioned in Section 3.6, “Additional Considerations.”

3.2 Multiple Machines

In this scenario, each distinct vendor daemon and its associated license file or files is located on a separate server machine. Each machine serves licenses just for its vendor daemon and runs its own local copy of `lmgrd`. Figure 3-1 shows this arrangement.

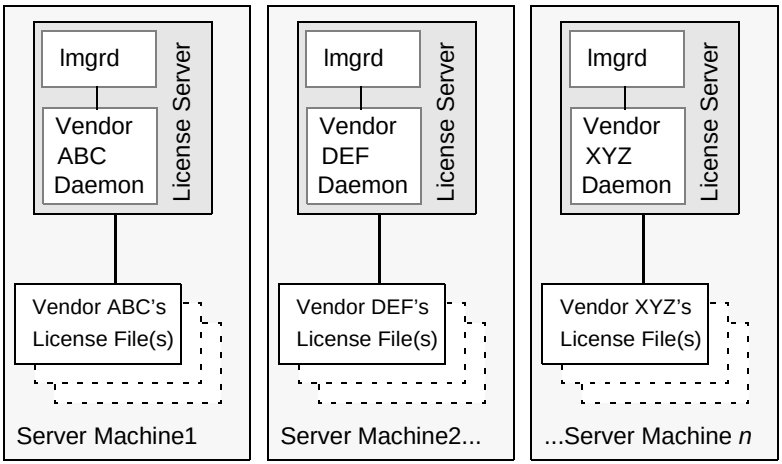


Figure 3-1: Multiple Server Machines

ADVANTAGES

- The license files for each vendor are independent from one another.
- Machines are maintained separately. If one machine goes down, the other machines continue to serve licenses for their vendors.
- Each server has its own debug log.
- Licensing load is distributed.

DISADVANTAGES

- Administrative overhead is the highest.
- If one machine goes down, licensing for the vendor's product associated with that machine is down.

STARTING THE LICENSE SERVER

Invoke the license manager daemon on each machine:

```
lmgrd -c server_machine_n_license_list
```

Where *`server_machine_n_license_list`* is a license-file list as described in Section 3.5, "Managing Multiple License Files." Each `lmgrd` starts the vendor daemon referred to in its license file(s).

3.3 One Machine with Multiple License Servers

In this model, each vendor daemon and its associated license file or files is served by its own lmgrd process, and everything is contained in one server machine. Figure 3-2 depicts this scheme.

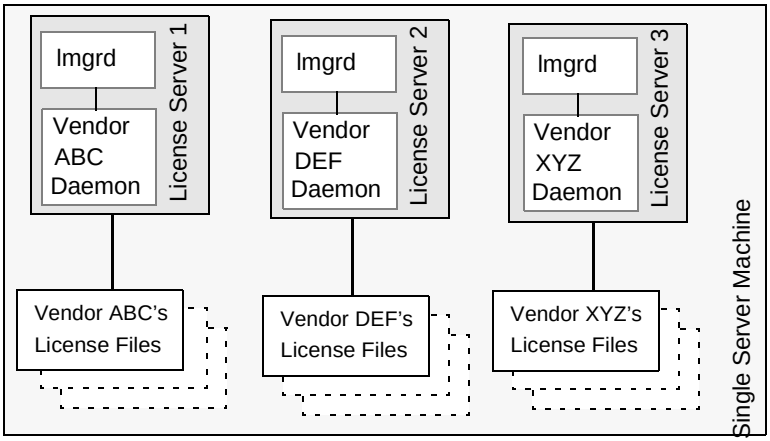


Figure 3-2: Multiple lmgrds, Multiple License Files

When maintaining separate license servers on the same machine, keep in mind:

- If the TCP/IP port number is specified on the SERVER line, it must be different for each license server. Use a standard text editor to change the TCP/IP port number in each license file so that they are all different. If you are running 10 license servers or less, you can omit all port numbers and `lmgrd` will choose unique ones for you within the default range of 27000-27009.
- You must make sure that you are using a compatible version of `lmgrd` for each particular license file. This is done by using an explicit path to `lmgrd`. See Section 3.6.2, "Version Component Compatibility."
- The number of license servers is limited only by the CPU memory and networking of the machine.

ADVANTAGES

- The license files for each vendor are independent from one another.
- License servers are maintained separately. If one server goes down, the other servers continue to serve licenses for their vendors.
- Each server has its own debug log.

DISADVANTAGES

- Administrative overhead is high.
- If the machine goes down, all licensing is disabled.
- Licensing load is concentrated to one machine.

STARTING THE LICENSE SERVER

Invoke each license server:

- For Server 1: `lmgrd -c` *`vendor_ABC_license_dir_list`*
- For Server 2: `lmgrd -c` *`vendor_DEF_license_dir_list`*
- For Server 3: `lmgrd -c` *`vendor_XYZ_license_dir_list`*

Where *`vendor_nnn_license_list`* is a license-file list as described in Section 3.5, “Managing Multiple License Files.” Each `lmgrd` starts the vendor daemon referred to in its license file(s).

3.4 One Machine with One License Server and Multiple License Files

In this scenario, one `lmgrd` process runs on the server machine and serves one or more vendor daemons, each with one or more license files; the license files usually are in the same directory. The standard filename extension for license files is `.lic`. The number of vendor daemons is not limited by FLEX*lm*. Figure 3-3 illustrates this scenario.

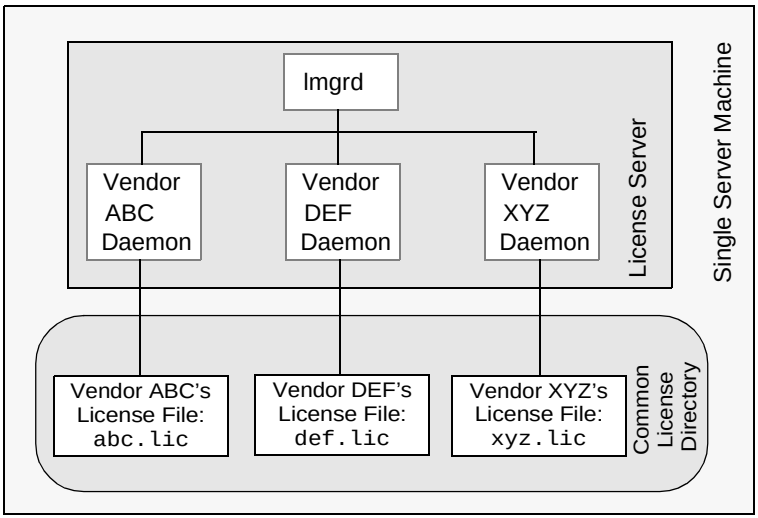


Figure 3-3: One lmgrd, Multiple License Files

ADVANTAGES

- The license files can be maintained separately.
- Reduced administrative overhead.

DISADVANTAGES

- One lmgrd serves all vendor daemons. If `lmgrd` goes down, all licensing is disabled.
- If the machine goes down, all licensing is disabled.
- Output from all vendor daemons goes into one common debug log unless separate debug logs are specified with DEBUGLOG in each vendor's options file. Having one common debug log makes it harder to debug a single vendor daemon's problem.
- Maximizes licensing load to one machine and one `lmgrd` process.

STARTING THE LICENSE SERVER

Invoke the license manager daemon once on the server machine.

```
lmgrd -c common_license_directory
```

`lmgrd` processes all files with the `.lic` extension in *`common_license_directory`* and starts all vendor daemons referred to in those files; so, there is no need to enumerate each license file name on the `lmgrd` command line.

SEE ALSO

- Section 3.5, "Managing Multiple License Files"
- Section H.1.2, "Capturing Debug Log Output for a Particular Vendor Daemon"

FLEXLM VERSION NOTES

- Ability for one `lmgrd` to process multiple license files introduced in v6.0 `lmgrd`.

3.5 Managing Multiple License Files

You can manage multiple license files that are on the same server machine via a license-file list. A license-file list is specified two ways:

- By using the `-c` option to `lmgrd`

```
lmgrd -c license_file_list [other lmgrd options]
```

- By defining the `LM_LICENSE_FILE` environment variable within the scope of the `lmgrd` process' environment.

  For example, in the C shell:

  ```
  setenv LM_LICENSE_FILE license_file_list
  ```

  In the Korn and Bourne shells:

  ```
  LM_LICENSE_FILE=license_file_list
  export LM_LICENSE_FILE
  ```

  On Windows 95/98/ME, add the following line to `C:\autoexec.bat`:

  ```
  SET LM_LICENSE_FILE=license_file_list
  ```

  On Windows NT/2000/XP, use the System Control Panel to change the global environment, adding *`license_file_list`* to `LM_LICENSE_FILE`.

Install the license files in convenient locations on the server machine and then define the license-file list.

Wherever *`license_file_list`* is specified it consists of a list of one or more of the following components:

- the full path to the license file
- a directory containing one or more license files with a `.lic` extension

---

Note: Use a colon (" : ") to separate the license file names on UNIX and on Windows use a semicolon (" ; ").

`lmgrd` builds up an internal license-file list when it starts up by parsing each license-file list component in the order listed.

Some scenarios where a license-file list is used include those described in Section 3.2, "Multiple Machines," Section 3.3, "One Machine with Multiple License Servers," or Section 3.4, "One Machine with One License Server and Multiple License Files.":

SEE ALSO

- Section 2.1.1, "Setting the Path with an Environment Variable"
- Section 4.3.1, "Redundancy via License-File List"
- Appendix D, "FLEXlm Environment Variables"

3.6 Additional Considerations

3.6.1 Combining license files

If you have two or more products whose licenses are intended for the same machine, as specified by their SERVER lines, you may be able to combine the license files into a single license file. The license files for the models described in Section 3.3, “One Machine with Multiple License Servers,” and Section 3.4, “One Machine with One License Server and Multiple License Files,” could be combined if they met certain criteria. Figure 3-4 shows one possible scenario using a combined license file.

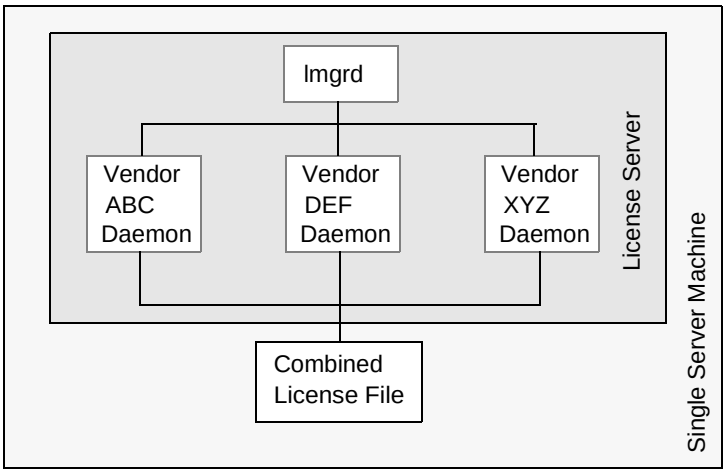


Figure 3-4: One lmgrd, One License File

ADVANTAGES

- A single license file to administer.
- Low administrative overhead.

DISADVANTAGE

- It is complicated to decide how to combine license lines from multiple vendors into one file, initially and over time.

STARTING THE LICENSE SERVER

Invoke the license manager daemon once on the server machine.

```
lmgrd -c combined_license_file
```

Criteria for Combining License Files

Your product's license file(s) define the license server(s) by host name and hostid in the SERVER line(s) in the license file. License files are candidates for combining under the following conditions:

- The number of SERVER lines in each file is the same.
- The hostid field of each SERVER line in one file *exactly* matches the hostid field of each SERVER line in the other file.

Some possible reasons license files may not be compatible are:

- License files are set up to run on different server machines, so hostids are different.
- One file is set up for single server (has only one SERVER line), the other is set up for a three-server redundant license server (has multiple SERVER lines).
- Hostids for the same machine use different hostid types. For example, the SERVER line in one license file uses INTERNET= for its hostid type and the other file uses the ethernet MAC address for its hostid type.

If your license files are compatible as described above, then you have the option of combining license files as summarized in Figure 3-4 and below in "How to Combine License Files." Note that you are not required to combine compatible license files. There is no performance or system-load penalty for not combining the files.

How to Combine License Files

If your license files are compatible, use any text editor to combine them. To combine license files, read all of the compatible license files into one file, then edit out the extra SERVER lines so that only one set of SERVER lines remains. Save the resulting data, and you have your combined license file. Figure 3-5 shows an example of combining license files.

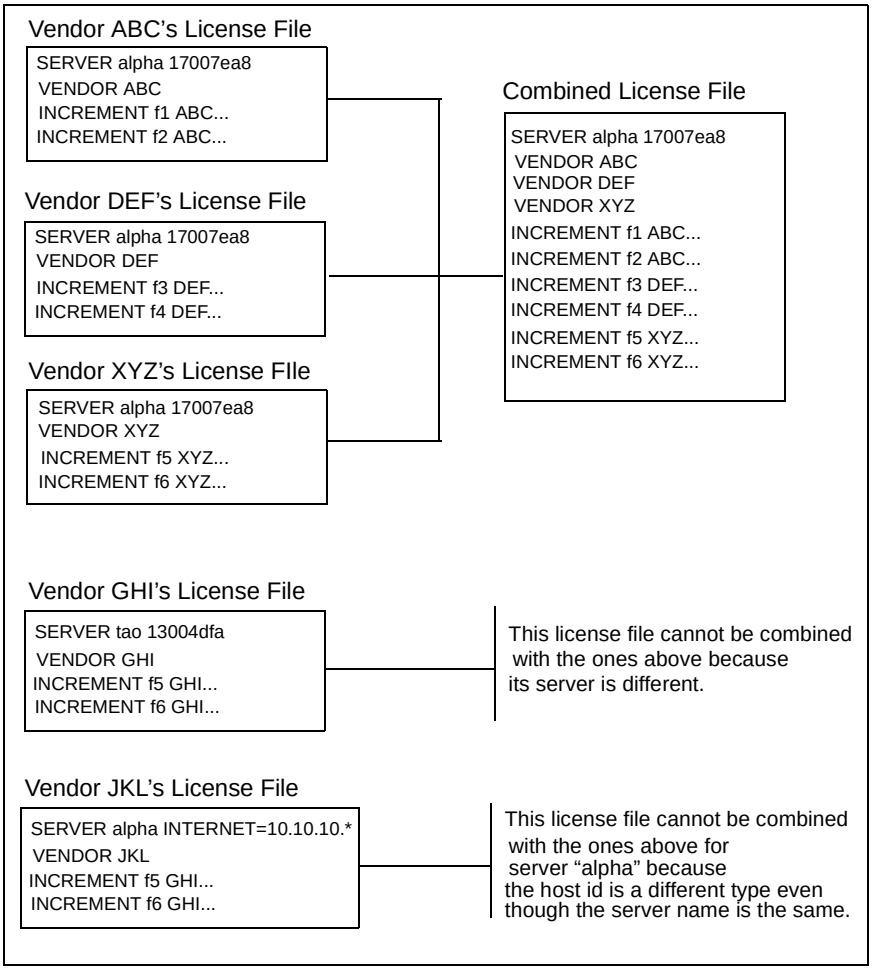


Figure 3-5: Combining License Files

3.6.2 Version Component Compatibility

When one lmgrd process manages multiple vendor daemons, it may be the case that those vendor daemons do not use the same version of FLEX*lm*. FLEX*lm* is designed to handle this situation. There are two basic compatibility rules for FLEX*lm*:

- A newer `lmgrd` can be used with an older vendor daemon, but a newer vendor daemon might not work properly with an older `lmgrd`.
- A newer vendor daemon (or `lmgrd`) can be used with an older FLEX*lm*-licensed application, but a newer FLEX*lm*-licensed application might not work properly with an older vendor daemon.
- A newer version of the FLEX*lm* utilities can be used with an older vendor daemon or `lmgrd`.

From these two compatibility rules come the simple rules for selecting which version of administration tools to use:

- Always use the newest version of `lmgrd` and the newest version of each vendor daemon.
- Use the newest FLEX*lm* utilities.

For specific FLEX*lm*-licensed applications, use either the new or the old version (of course, the vendor daemon for that application must be at least as new as the application itself).

Note: Use the `lmver` utility to determine the version of lmgrd, the vendor daemon, the utilities, and the licensed application.

SEE ALSO

- Section 7.15, “lmver,” for more information on the lmver utility.
- Section I.1, “Version Compatibility and Components,” for further information on version compatibility.

Chapter 4

Selecting Server Machines

This chapter helps you decide which machines to use as license server machines.

4.1 Resources Used by the Server

This section discusses the resources used by the license server. When you select a server machine, you may need to take into account the system limits on these resources. For small numbers of licenses (under about 100), most of these system limits are not a problem on any workstation.

4.1.1 Sockets

When using TCP/IP ports, a single vendor daemon supports as many users as the per-process system limit for file descriptors. When no more file descriptors are available to a daemon, additional vendor daemons are spawned to allow for extra file descriptors. When using UDP, there is no limit to the number of end users per vendor daemon process, because they can share a single socket in the vendor daemon (UDP has other drawbacks, and TCP is preferred). If there are more than 1000 concurrent clients being supported by a single vendor daemon, then it's probably good to split the license file into more than one file, from different servers, to lighten the networking traffic (which requires the vendor to agree to issue new licenses). Licensed applications check out licenses from multiple servers using a license-file list via the `LM_LICENSE_FILE` environment variable.

Each FLEX*lm*-licensed application connected to a license server uses one socket. The total number of sockets used by the license server programs is slightly larger than the total number of simultaneous clients.

On older SCO systems, the default number of sockets may be set fairly low; if you choose to run a server on such a machine, you may need to reconfigure your kernel to have more sockets.

The number of sockets available for Windows 95/98/ME clients is about 60. In general, NT/2000/XP is preferred for server systems, where there is no such limit, and the operating system is better designed for server processes.

4.1.2 CPU Time

For small numbers of clients, the license servers use very little CPU time. The servers might have only a few seconds of CPU time after many days.

For a large number of clients (who are each exchanging heartbeat messages with the server), or for high checkout/checkin activity levels (hundreds per second), the amount of CPU time consumed by the server may start to become significant, although, even here, CPU usage is normally not high. In this case, you may need to ensure that the server machine you select has enough CPU cycles to spare.

4.1.3 Disk Space

The only output files created by the license servers are the debug and report log files. The report log files are used to generate accurate usage reports by SAM*report*. If you have a lot of license activity, these log files grow very large. You need to consider where to put these files and how often to rotate and archive them. The license administrator has the option to suppress log file output if disk space is at a premium.

It is recommended that the log files are local files on the server machine(s) to avoid networking dependencies.

SEE ALSO

- Section 2.1.1, “Setting the Path with an Environment Variable”
- Section 6.2, “Starting the License Manager Daemon on UNIX Platforms”
- Appendix G, “The Report Log File”
- Appendix H, “The Debug Log File”

4.1.4 Memory

The FLEX*lm* daemons use little memory. On SunOS, `lmgrd` uses approximately2 MB and the vendor daemons use approximately 2 MB each, although memory usage increases in the vendor daemon with the size of the license file, size of the options file, and the number of concurrent users.

4.1.5 Network Bandwidth

FLEX*lm* sends relatively small amounts of data across the network. Each transaction, such as a checkout or checkin, is typically satisfied with less than 1 KB of data transferred. This means that FLEX*lm* licensing can be effectively run over slow networks (such as dial-up SLIP lines) for small numbers of clients.

For a large number of FLEX*lm*-licensed applications (hundreds), each of which exchange heartbeat messages with the vendor daemon, the network bandwidth used may start to become significant. In this case, run the FLEX*lm*-licensed application and server on the same local area network, which may require splitting licenses between two files for two servers. Users can use a license-file list in the `LM_LICENSE_FILE` environment variable to have effective access to both servers.

SEE ALSO

- Section 2.1, “Specifying Location of the License File”

4.2 Remote Mounted Disks

Macrovision recommends that you do not use remote mounted disks when you run the license server. In other words, it is recommended that `lmgrd`, the vendor daemons, the license file, and the debug and report log files are all on locally mounted disks. If any of these files is on a remote mounted disk, you double the points of failure which could lead to a temporary loss of all of your licenses. When all files are mounted locally, the licenses are available as long as the server machine is up; but when the files are on a different machine, then the loss of either the license server machine or the file server machine causes the licenses to be unavailable.

4.3 Redundant License Servers

If you wish to use redundant servers, select stable systems as server machines; in other words, do not pick systems that are frequently rebooted or shut down for one reason or another. Redundant license server machines are any supported server machines.

FLEX*lm* supports two methods of redundancy:

- via a license-file list in the `LM_LICENSE_FILE` environment variable
- via a set of three redundant license servers

With `LM_LICENSE_FILE` list redundancy, each one of a group of license servers serves a subset of the total licenses. The end user sets `LM_LICENSE_FILE` to a list of license files, where each license file refers to one of the license servers. The application then tries each server in the list, in order, until it succeeds or gets to the end of the list.

With three-server redundancy, if any two of the three license servers are up and running (two out of three license servers is referred to as a *quorum*), the system is functional and serves its total complement of licenses.

SEE ALSO

- Section 3.5, “Managing Multiple License Files.”

4.3.1 Redundancy via License-File List

This is best explained by example. If ten licenses are desired for both “f1” and “f2,” the vendor issues two sets of licenses with a count of 5 for each of “f1” and “f2.” The server machines (unlike three-server redundancy) can be physically distant.

The license files look like:

License 1 for “chicago”

```
SERVER chicago 17007ea8 1700
VENDOR sampled /etc/mydaemon
FEATURE f1 sampled 1.000 01-jan-2005 5 26C7DD9C0186
FEATURE f2 sampled 1.000 01-jan-2005 5 8CE46C57041D
```

License 2 for “tokyo”

```
SERVER tokyo 17a07e08 1700
VENDOR sampled /etc/mydaemon
FEATURE f1 sampled 1.000 01-jan-2005 5 16BE40E1D98D
FEATURE f2 sampled 1.000 01-jan-2005 5 6DB6F3E402DF
```

The user in Chicago could set `LM_LICENSE_FILE` to:

```
1700@chicago:1700@tokyo
```

The user in Tokyo could set `LM_LICENSE_FILE` to:

```
1700@tokyo:1700@chicago
```

Remember to separate the license file names with a colon (“ : ”) on UNIX and with a semicolon (“ ; ”) on Windows. The application attempts the first server in the list, and if that fails for any reason, the second server is tried.

4.3.2 Three-Server Redundancy

These three-server redundant servers need to have excellent communications on a reliable network and need to be located on the same subnet. The three servers must be located physically close to each other. This form of redundancy requires that the servers exchange heartbeats periodically, and poor communications can cause poor performance. Avoid configuring redundant servers with slow communications or dial-up links.

Three-server redundancy is designed to provide hardware failover protection only and does not provide load-balancing. Use `LM_LICENSE_FILE` list, instead, if load-balancing is desired. This is because with three-server redundancy, only one of the three servers is “master,” capable of issuing licenses. Since all clients must contact the “master,” all clients must have reliable networking to a single machine.

4.3.3 Comparing Three-Server to License-File List

ARE THERE ANY DRAWBACKS TO USING THE LICENSE-FILE LIST FOR REDUNDANCY?

Yes. By default, once a *license job* has successfully checked out a license from one host, all subsequent checkouts must be satisfied from the same host. If the application requires more than one license, this could result in a license denial when the license is available on another server. An application bypasses this restriction if it is coded with the use of multiple FLEX*lm* license jobs. Only your application vendor knows if their application is programmed in this manner.

If the application supports license queueing, all licenses are queued only from the first host on the list rather than the request moving to another server on the list.

Finally, if one server becomes unavailable, some licenses are unavailable.

WHEN IS IT RECOMMENDED TO USE A LICENSE-FILE LIST FOR REDUNDANCY RATHER THAN THREE-SERVER REDUNDANT SERVERS?

- When there’s less system administration available to monitor license servers.
- When load-balancing is needed for FLEX*lm*-licensed applications located far apart, e.g., London and Tokyo, make servers available locally, with remote servers available as backup.
- License-file list is more forgiving if you lose quorum.
- License-file list is not limited to three servers (any number work).

- Clients do not require reliable networking to a single machine with license-file list, so this is recommended where networking itself requires redundancy.

4.4 Counted vs. Uncounted Licenses

The license file determines whether a license server is needed. If all the FEATURE (or INCREMENT) lines have a license count of 0 (unlimited) or “uncounted”, then no server is needed. This type of license is called uncounted. Alternatively, if any FEATURE lines have a non-zero license count, then a server is required to count those licenses. If a vendor wants to use FLEX*lm* without a server, they must issue uncounted licenses.

The license server is able to serve uncounted licenses as well. This is done so that:

- transactions can be logged into the report log for all license requests, which can then be reported on by SAM*report*
- options file constraints can be applied to the licenses

To have uncounted licenses served, include a SERVER line in the license file, and put the USE_SERVER line immediately after the SERVER line. The vendor daemon serves the uncounted licenses, and the USE_SERVER line indicates to applications that requests must go to the license server for authorization.

FLEXLM VERSION NOTES

- “uncounted” keyword introduced in v6 FLEX*lm* client library.

Chapter 5

The Options File

The options file allows the license administrator to control various operating parameters of FLEX*lm*. Users are identified by their user name, host name, display, IP address, or PROJECT (which is set with the `LM_PROJECT` environment variable).

Specifically, the license administrator can:

- Allow the use of features
- Deny the use of features
- Reserve licenses
- Restrict the number of licenses available
- Control the amount of information logged about license usage
- Enable a report log file

Options files allow you, as the license administrator, to be as secure or open with licenses as you like.

Lines in the options file are limited to 2048 characters. The "\" character is a continuation character in options file lines.

FLEXLM VERSION NOTES

- PROJECT identification (set by `LM_PROJECT`) in options file introduced in v7.0 vendor daemon.

5.1 Creating an Options File

To create an options file:

1. Use the appropriate options listed in Section 5.2, "Options File Syntax," to create the options file using any text editor.

2. Locate the options file anywhere; however, it is recommended that the options file be placed in the same directory as the license file.
3. Add the path to the options file in the license file as the fourth field on the VENDOR line for the application's vendor daemon. For example:

   ```
   VENDOR sampled /etc/sampled \
         [options=]/sample_app/sampled/licenses/sampled.opt
   ```

   enables the `sampled` vendor daemon to look at the specified options file.

   If the path is omitted, the vendor daemon automatically looks for a file according to the following criteria:

   - the name the file is *`vendor`*`.opt`, where *`vendor`* is the vendor daemon name
   - it is placed it in the same directory as the license used by `lmgrd`, it is automatically used at server startup

FLEXLM VERSION NOTES

- The default options file name, *`vendor`*`.opt`, introduced in v6 vendor daemon.

5.2 Options File Syntax

Below is an overview of the options file syntax. See Section 5.5, "Options File Examples," for examples and additional information.

Each line of the file controls one option. Table 5-1 lists the option keywords.

Table 5-1: Option Keywords

Option Keyword	Description
`BORROW_LOWWATER`	Set the number of BORROW licenses that cannot be borrowed.
`DEBUGLOG`	Writes debug log information for this vendor daemon to the specified file (v8.0+ vendor daemon).
`EXCLUDE`	Deny a user access to a feature.

Table 5-1: Option Keywords (Continued)

Option Keyword	Description
EXCLUDE_BORROW	Deny a user the ability to borrow BORROW licenses.
EXCLUDEALL	Deny a user access to *all* features served by this vendor daemon.
GROUP	Define a group of users for use with any options.
HOST_GROUP	Define a group of hosts for use with any options (v4.0+).
INCLUDE	Allow a user to use a feature.
INCLUDE_BORROW	Allow a user to borrow BORROW licenses.
INCLUDEALL	Allow a user to use *all* features served by this vendor daemon.
LINGER	Allow a user to extend the linger time for a feature beyond its checkin.
MAX	Limit usage for a particular feature/group—prioritizes usage among users.
MAX_OVERDRAFT	Limit overdraft usage to less than the amount specified in the license.
NOLOG	Turn off logging of certain items in the debug log file.
REPORTLOG	Specify that a report log file suitable for use by the SAM*report* license usage reporting tool be written.
RESERVE	Reserve licenses for a user or group of users/hosts.
TIMEOUT	Specify idle timeout for a feature, returning it to the free pool for use by another user.

Table 5-1: Option Keywords (Continued)

Option Keyword	Description
`TIMEOUTALL`	Set timeout on all features.

FLEXlm Version Notes

- BORROW_LOWWATER options keyword introduced in v8.0 vendor daemon.
- EXCLUDE_BORROW options keyword introduced in v8.0 vendor daemon.
- INCLUDE_BORROW options keyword introduced in v8.0 vendor daemon.

Comments

Include comments in your options file by starting each comment line with a pound sign “#.” Everything in an options file is case sensitive. Be sure that user names and feature names, for example, are entered correctly.

Feature Specification

The feature name can be modified with an optional keyword-value pair to fully qualify it. This notation is used for distinguishing a particular group of licenses when there are multiple FEATURE lines for a single feature. The following syntax is used:

```
feature:keyword=value
```

For example:

```
f1:VERSION=2.0
```

specifies the version 2.0 pool of licenses for feature “f1”.

Note: A colon (:) is a valid feature name character. If colons are in your feature names, specify a group of licenses with the following alternative syntax using quotation marks and spaces:

```
"feature keyword=value"
```

The following option keywords are used as feature name modifiers to denote a specific group of licenses:

- VERSION=
- HOSTID=
- EXPDATE=
- KEY=
- SIGN=
- ISSUER=
- NOTICE=
- VENDOR_STRING= (if configured by the vendor as a pooling component)
- dist_info=
- user_info=
- asset_info=

If the USER_BASED or HOST_BASED keywords appear in a feature line, this feature specification syntax must be used to qualify the feature.

Using a package name in place of a feature name applies the option to all of the components in the package.

TYPE SPECIFICATION

The following option keywords restrict who may use licenses or where licenses may be used: EXCLUDE, EXCLUDEALL, EXCLUDE_BORROW, INCLUDE, INCLUDEALL, INCLUDE_BORROW, MAX, and RESERVE. These options take a `type` argument, which specifies whether the restriction is based on USER, HOST, DISPLAY, INTERNET, or PROJECT:

- USER—UNIX user name of the user executing the FLEX*lm*-licensed application.
- HOST—machine host name or IP address where the application is executing

  The IP-addresses can contain wildcards.
- DISPLAY—display where the application is displayed

  On UNIX, DISPLAY is `/dev/tty`*`xx`* (which is always `/dev/tty` when an application is run in the background) or the X-Display name. On Windows, it is the system name or, in the case of a terminal server environment, the terminal server client name.

- INTERNET—IP address of the machine where the application is executing (wildcards can be used in the IP address)
- PROJECT—`LM_PROJECT` environment variable set by the user who is executing the FLEX*lm*-licensed application

On Windows (without terminal server), the HOST and DISPLAY names are both set to the Window's system name. For licenses that allow checkouts from a terminal server (TS_OK keyword in the feature line), the USER, HOST, and DISPLAY names can be different from one another.

The types listed above take a single member. For example:

```
EXCLUDE coolsoft USER joe
```

To specify a list of users or hosts, first define the list using the GROUP or HOST_GROUP option lines, then use the GROUP or HOST_GROUP type to specify the group name. For example:

```
GROUP stars joe barbara susan
EXCLUDE coolsoft GROUP stars
```

FLEXLM VERSION NOTES

- IP address as a HOST specification introduced in v8 vendor daemon.
- LM_PROJECT environment variable introduced in V5 FLEX*lm* client library and vendor daemon.
- Colons in feature names introduced in v8 vendor daemon.

5.2.1 BORROW_LOWWATER

`BORROW_LOWWATER` *`feature`*`[:`*`keyword=value`*`]` *`n`*

Sets the number of licenses for a BORROW feature that cannot be borrowed.

`feature`	Name of feature being affected.
`keyword=value`	Feature name modifier to denote a group of licenses. See the section, "Feature Specification," for details.
`n`	Number of licenses that cannot be borrowed via license borrowing.

For example, if a feature “f1” has a count of 10 and borrowing is enabled in the application and on the FEATURE line:

```
FEATURE f1 ... 10 ... BORROW SIGN=...
```

the following line in the options file allows only 7 licenses to be borrowed.

```
BORROW_LOWWATER f1 3
```

5.2.2 DEBUGLOG

```
DEBUGLOG [+]debug_log_path
```

Specifies a location for the debug log output from the vendor daemon associated with this options file. Preceding the *debug_log_path* with a + character appends logging entries, otherwise the file is overwritten each time the daemon is started. Note that this affects output from only the vendor daemon associated with this options file. The debug log output of `lmgrd` and any other vendor daemons in the same license file is not captured in this file.

SEE ALSO:

- Section 7.13, “lmswitch”
- Appendix H, “The Debug Log File”

FLEXLM VERSION NOTES

- Debug log output restricted to that of just the vendor daemon introduced in V8 vendor daemon.

5.2.3 EXCLUDE

```
EXCLUDE feature[:keyword=value] type {name | group_name}
```

Excludes a user or pre-defined group of users, etc., from the list of who is allowed to use the feature. EXCLUDE supersedes INCLUDE; conflicts between the EXCLUDE list and the INCLUDE list are resolved by the EXCLUDE taking precedence.

feature	Name of the feature being affected.
keyword=value	Feature name modifier to denote a group of licenses. See the section, “Feature Specification,” for details.

type	One of USER, HOST, DISPLAY, INTERNET, PROJECT, GROUP, or HOST_GROUP. See the section, “Type Specification,” for details.
name	Name of an item of type *type* to reserve licenses for.
group_name	Name of the group to exclude.

To exclude the user “hank” from the list of users able to use feature “f1”:

```
EXCLUDE f1 USER hank
```

5.2.4 EXCLUDE_BORROW

```
EXCLUDE_BORROW feature[:keyword=value] type \
               {name | group_name}
```

Excludes a user or pre-defined group of users, etc., from the list of who is allowed to borrow licenses for this BORROW feature. EXCLUDE_BORROW supersedes INCLUDE_BORROW; conflicts between the EXCLUDE_BORROW list and the INCLUDE_BORROW list are resolved by the EXCLUDE_BORROW taking precedence.

feature	Name of the feature being affected.
keyword=value	Feature name modifier to denote a group of licenses. See the section, “Feature Specification,” for details.
type	One of USER, HOST, DISPLAY, INTERNET, PROJECT, GROUP, or HOST_GROUP. See the section, “Type Specification,” for details.
name	Name of an item of type *type* to reserve licenses for.
group_name	Name of the group to exclude.

To exclude the user “fred” from the list of users able to borrow feature “f1” assuming the feature has the BORROW attribute:

```
EXCLUDE_BORROW f1 USER fred
```

5.2.5 EXCLUDEALL

```
EXCLUDEALL type {name | group_name}
```

Excludes a user or pre-defined group of users, etc., from the list of who is allowed to use all features served by this vendor daemon.

type	One of USER, HOST, DISPLAY, INTERNET, PROJECT, GROUP, or HOST_GROUP. See the section, “Type Specification,” for details.
name	Name of an item of type *type* to reserve licenses for.
group_name	Name of the group to exclude.

To exclude any user on the machine “chaos” from using all features served by this vendor daemon:

```
EXCLUDEALL HOST chaos
```

5.2.6 GROUP

```
GROUP group_name user_list
```

Defines a group of users for use in INCLUDE, INCLUDEALL, EXCLUDE, EXCLUDEALL, and RESERVE option lines.

group_name	Name of the group being defined.
user_list	List of user names in that group.

To define the group “Hackers” consisting of “bob,” “howard,” and “james”:

```
GROUP Hackers bob howard james
```

Multiple GROUP lines for the same group name add all the specified users into the group.

Note: USER_GROUP is an alias for GROUP.

5.2.7 HOST_GROUP

```
HOST_GROUP group_name host_list
```

Defines a group of hosts for use in INCLUDE, INCLUDEALL, EXCLUDE, EXCLUDEALL, and RESERVE option lines. Multiple HOST_GROUP lines add all the specified hosts into the group.

group_name	Name of the group being defined.
host_list	List of host names in that group.

To define the host group “Pacific” consisting of “tokyo,” “seattle,” and “auckland”:

```
HOST_GROUP Pacific tokyo seattle auckland
```

Anywhere a host name can be used in an options file, an IP-address can be used instead.

5.2.8 INCLUDE

```
INCLUDE feature[:keyword=value] type {name | group_name}
```

Includes a user or pre-defined group of users, etc., in the list of who is allowed to use licenses for this feature. Anyone not in an INCLUDE statement is not allowed to use that feature. EXCLUDE supersedes INCLUDE; conflicts between the EXCLUDE list and the INCLUDE list are resolved by the EXCLUDE taking precedence.

feature	Name of the feature being affected.
keyword=value	Feature name modifier to denote a group of licenses. See the section, “Feature Specification,” for details.
type	One of USER, HOST, DISPLAY, INTERNET, PROJECT, GROUP, or HOST_GROUP. See the section, “Type Specification,” for details.
name	Name of an item of type *type* to reserve licenses for.
group_name	Name of the group to include.

To include user “bob” in the list of users able to use feature “f1”:

```
INCLUDE f1 USER bob
```

Note: INCLUDE is required for USER_BASED or HOST_BASED features. The system administrator specifies which users are allowed to use the product, via INCLUDE, and the license limits the number of users that are INCLUDEd.

5.2.9 INCLUDE_BORROW

```
INCLUDE_BORROW feature[:keyword=value] type \
               {name | group_name}
```

Includes a user or pre-defined group of users, etc., in the list of who is allowed to borrow the BORROW feature. Anyone not in an INCLUDE_BORROW statement is not allowed to borrow licenses. EXCLUDE_BORROW supersedes INCLUDE_BORROW; conflicts between the EXCLUDE_BORROW list and the INCLUDE_BORROW list are resolved by the EXCLUDE_BORROW taking precedence.

feature	Name of the feature being affected.
keyword=value	Feature name modifier to denote a group of licenses. See the section, “Feature Specification,” for details.
type	One of USER, HOST, DISPLAY, INTERNET, PROJECT, GROUP, or HOST_GROUP. See the section, “Type Specification,” for details.
name	Name of an item of type *type* to reserve licenses for.
group_name	Name of the group to include.

To include user “tom” in the list of users able to borrow feature “f1”:

```
INCLUDE_BORROW f1 USER tom
```

5.2.10 INCLUDEALL

```
INCLUDEALL type {name | group_name}
```

Includes a user or pre-defined group of users, etc. in the list of who is allowed to use all features served by this vendor daemon. Anyone not in an INCLUDEALL statement is not allowed to use these features.

type	One of USER, HOST, DISPLAY, INTERNET, PROJECT, GROUP, or HOST_GROUP. See the section, “Type Specification,” for details.
name	Name of an item of type *type* to reserve licenses for.
group_name	Name of the group to include.

To allow the user “jane” to use all features served by this vendor daemon:

```
INCLUDEALL USER jane
```

5.2.11 LINGER

```
LINGER feature[:keyword=value] seconds
```

A lingering license stays checked out for a specified period of time beyond its checkin or licensed application exit, whichever comes first. This option extends the default linger time configured by the vendor in the licensed application.

Note: The vendor must have added enabled this feature in the licensed application for it to work. Contact your software vendor to find out if this feature is implemented.

---

feature	Name of the feature.
keyword=value	Feature name modifier to denote a group of licenses. See the section, “Feature Specification,” for details.

seconds	Number of seconds the license lingers. The vendor sets a minimum value. If you specify a value for *seconds* that is smaller than the vendor's minimum, the minimum is used.

To set the linger value for feature "f1" to one hour (3600 seconds):

```
LINGER f1 3600
```

The actual linger time varies somewhat since the vendor daemon checks all lingering licenses just once per minute. If, however, a new license request is made that would otherwise be denied, a check of the lingering licenses is made immediately to attempt to satisfy the new request.

5.2.12 MAX

```
MAX num_lic feature[:keyword=value] type {name | group_name}
```

Limits usage for a group or user.

num_lic	Usage limit for this user or group.
feature	Feature this limit applies to.
keyword=value	Feature name modifier to denote a group of licenses. See the section, "Feature Specification," for details.
type	One of USER, HOST, DISPLAY, INTERNET, PROJECT, GROUP, or HOST_GROUP. See the section, "Type Specification," for details.
name	Name of an item of type *type* to reserve licenses for.
group_name	Name of the group to limit.

5.2.13 MAX_OVERDRAFT

```
MAX_OVERDRAFT feature[:keyword=value] num_lic
```

Limits OVERDRAFT license usage below the OVERDRAFT allowed by the license file.

feature	Feature this limit applies to.
keyword=value	Feature name modifier to denote a group of licenses. See the section, “Feature Specification,” for details.
num_lic	Usage limit for this user or group.

5.2.14 NOLOG

```
NOLOG { IN | OUT | DENIED | QUEUED }
```

Suppresses logging the selected type of event in the debug log file.

To turn off logging of checkins:

```
NOLOG IN
```

To turn off logging of checkouts and queued requests two separate NOLOG lines are required:

```
NOLOG DENIED
NOLOG QUEUED
```

Note: License administrators use this option to reduce the size of the debug log file. However, it can reduce the usefulness of the debug log in debugging license server problems.

See also Section 7.13, “lmswitch.”

5.2.15 REPORTLOG

```
REPORTLOG [+]report_log_path
```

REPORTLOG specifies the report log file for this vendor daemon. It is recommended preceding the *report_log_path* with a + character to append logging entries, otherwise the file is overwritten each time the daemon is started.

On Windows, pathnames which include spaces have to be enclosed in double quotes.

Note: SAM*report*, a separate product available from Macrovision, is used to process FLEX*lm* report log files. SAM*report* processes only report log files, not debug log files.

REPORTING ON PROJECTS WITH LM_PROJECT

The SAM*report* report writer reports on "projects." A project is set up by having all users working on the same project set their `LM_PROJECT` environment variable (or registry on Windows) to a string that describes the project. SAM*report* groups usage by project, as defined by what `LM_PROJECT` was set to when the application was run.

SEE ALSO

- Appendix D, "FLEXlm Environment Variables"
- Appendix G, "The Report Log File"

5.2.16 RESERVE

```
RESERVE num_lic feature[:keyword=value] type
        {name | group_name}
```

Reserves licenses for a specific user.

num_lic	Number of license to reserve for this user or group.
feature	Feature this reservation applies to.
keyword=value	Feature name modifier to denote a group of licenses. See the section, "Feature Specification," for details.
type	One of USER, HOST, DISPLAY, INTERNET, PROJECT, GROUP, or HOST_GROUP. See the section, "Type Specification," for details.
name	Name of an item of type *type* to reserve licenses for.
group_name	Name of group to reserve licenses for.

To reserve one license of feature “f1” for user “mel”:

```
RESERVE 1 f1 USER mel
```

If you want to reserve a license for *each* of several users or groups, you must use a separate RESERVE line for each user or group.

Note: Any licenses reserved for a user are dedicated to that user. Even when that user is not actively using the license it is unavailable to other users. However, a RESERVEd license does not cause usage to be reported by SAM*report* if the license is not actually in use.

5.2.17 TIMEOUT

```
TIMEOUT feature[:keyword=value] seconds
```

Sets the time after which an inactive license is freed and reclaimed by the vendor daemon.

Note: The vendor must have enabled this feature in the licensed application for it to work. Contact your software vendor to find out if this feature is implemented.

feature	Name of the feature.
keyword=value	Feature name modifier to denote a group of licenses. See the section, “Feature Specification,” for details.
seconds	Number of seconds after which inactive license is reclaimed. The vendor sets a minimum value.If you specify a value for *seconds* that is smaller than the vendor’s minimum, the minimum is used.

To set the timeout for feature “f1” to one hour (3600 seconds):

```
TIMEOUT f1 3600
```

TIMEOUT checks in the licenses if the FLEX*lm*-licensed application has been inactive for a period longer than the specified time period. The daemon declares a process inactive when it has not received heartbeats from it; whereas, an active FLEX*lm*-licensed application sends heartbeats.

A TIMEOUT line must be present in the options file in order to take advantage of the vendor-enabled timeout feature.

5.2.18 TIMEOUTALL

```
TIMEOUTALL seconds
```

Same as TIMEOUT, but applies to all features.

5.3 How the Vendor Daemon Uses the Options File

When the vendor daemon is started by `lmgrd`, the vendor daemon reads its options file. There is only one options file per vendor daemon and each vendor daemon needs its own options file. For any changes in an options file to take effect, the vendor daemon must read its options file. The `lmreread` utility causes the vendor daemon to reread its options file.

FLEXLM VERSION NOTES

- The `lmreread` utility enhanced in v8.0 vendor daemon so that it causes the vendor daemon to reread the options file. If you are using earlier versions of FLEX*lm*, the vendor daemon must be stopped and restarted in order for the options file to be reread.

5.4 Rules of Precedence in Options Files

Rules of precedence take effect when INCLUDE and EXCLUDE statements are combined in the same options file and control access to the same features. The following define the precedence when both types of statements appear together:

- If there is only an EXCLUDE list, everyone who is not on the list is allowed to use the feature.
- If there is only an INCLUDE list, only those users on the list is allowed to use the feature.
- If neither list exists, everyone is allowed to use the feature.
- The EXCLUDE list is checked before the INCLUDE list; someone who is on both lists is not allowed to use the feature.

Once you create an INCLUDE or EXCLUDE list, everyone else is *implicitly* outside the group. This feature allows you, as an administrator, the ability to control licenses without having to *explicitly* list each user that you wish to allow or deny access to. In other words, there are two approaches; you either:

- Give most users access and list only the exceptions, or
- Severely limit access and list only the those users that have access privileges

5.5 Options File Examples

The following information gives some examples of options files intended to illustrate ways to effectively control access to your licenses.

5.5.1 Simple Options File Example

```
RESERVE 1 compile USER robert
RESERVE 3 compile HOST mainline
EXCLUDE compile USER lori
NOLOG QUEUED
```

This options file:

- Reserves one license for the feature “compile” for the user “robert.”
- Reserves three licenses for the feature “compile” for anyone on the system with the host name “mainline.”
- Prevents the user “lori” from using the “compile” feature on any machine on the network.
- Causes QUEUED messages to be omitted from the debug log file.

The sum total of the licenses reserved must be less than or equal to the number of licenses specified in the FEATURE line. In the example above, there must be a minimum of four licenses on the “compile” FEATURE line. If fewer licenses are available, only the first set of reservations (up to the license limit) is used.

If this data were in file `/a/b/sampled/licenses/sampled.opt`, then modify the license file VENDOR line as follows:

```
VENDOR sampled /etc/sampled \
                /sample_app/sampled/licenses/sampled.opt
```

5.5.2 Limiting Access for Multiple Users

Each INCLUDE, INCLUDEALL, INCLUDE_BORROW, EXCLUDE, EXCLUDEALL, EXCLUDE_BORROW, MAX, and RESERVE line must have a single user name (or group) listed. To affect more than one user name create a GROUP. For example to exclude “bob,” “howard,” and “james” from using the feature called “toothbrush,” create the following options file:

```
EXCLUDE toothbrush USER bob
EXCLUDE toothbrush USER howard
EXCLUDE toothbrush USER james
```

However, there is an easier way. Create a GROUP and exclude the list of users from using the feature. Like the previous example, the following options file excludes “bob,” “howard,” and “james” from using the feature called “toothbrush”:

```
# First define the group "Hackers"
GROUP Hackers bob howard james
# Then exclude the group
EXCLUDE toothbrush GROUP Hackers
```

Now when you want to allow or deny access to any feature to that group, you have an alias list to make it simple.

Use HOST_GROUP to allow, deny, or reserve licenses for multiple hosts. For example, to exclude all users logged in on the hosts “fred” and “barney” from using a feature called “f1,” add these lines to your options file:

```
HOST_GROUP writers fred barney
EXCLUDE f1 HOST_GROUP writers
```

SEE ALSO

- Section 5.2.7, “HOST_GROUP,” for more information about defining groups

5.5.3 EXCLUDE Example

```
#First Define the group "painters"
GROUP painters picasso mondrian klee
EXCLUDE spell GROUP painters
EXCLUDE spell USER bob
EXCLUDE spell INTERNET 123.123.123.*
```

This options file:

- Prevents the users “picasso,” “mondrian,” and “klee” from using the feature “spell” on any machine on the network.

- Prevents the user “bob” from using the feature “spell” on any machine on the network.
- Prevents any user logged into a host with an IP address in the range 123.123.123.0 through 123.123.123.255 from using the feature “spell.”
- Allows any other user, as long as they are not on the excluded IP addresses, *and* they are not a member of the “painters” GROUP, *and* they are not “bob,” to use feature “spell” (by implication).

Note that “bob” could have been added to the group “painters.” However, “painters” might be used for some other purpose in the future so the license administrator chose to handle “bob” as a special case here. In this case, the two EXCLUDE statements concatenate to create a list of four users.

5.5.4 INCLUDE Example

```
INCLUDE paint USER picasso
INCLUDE paint USER mondrian
INCLUDE paint HOST bigbrush
```

This options file:

- Allows the user “picasso” to use the feature “paint” on any machine on the network.
- Allows the user “mondrian” to use the feature “paint” on any machine on the network.
- Allows any user, as long as they are on the host “bigbrush,” to use feature “paint.”
- Denies access to the feature “paint” to anyone except “picasso,” “mondrian,” or anyone from the host “bigbrush” (by implication).

Chapter 6

The License Manager Daemon

The *license manager daemon*, `lmgrd`, is one of two components which comprise the license server (the other being the vendor daemon). It handles the initial contact with FLEX*lm*-licensed applications, passing the connection on to the appropriate vendor daemon. The purpose of the license manager daemon, `lmgrd`, is to:

- Start and maintain all the vendor daemons listed in the VENDOR lines of the license file.
- Refer application checkout (or other) requests to the correct vendor daemon.

A newer `lmgrd` can be used with an older vendor daemon or FLEX*lm*-licensed application, but a newer vendor daemon or FLEX*lm*-licensed application might not work properly with an older `lmgrd`. Always use the newest version of `lmgrd` as possible; it is available for download from www.macrovision.com.

6.1 lmgrd Command-Line Syntax

`lmgrd` is the main daemon for FLEX*lm*. When you invoke `lmgrd`, it looks for a license file which contains information about vendors and features and starts those vendor daemons.

Usage is:

```
lmgrd [-c license_file_list] [-l [+]debug_log_path]
      [-2 -p] [-local] [-x lmdown]
      [-x lmremove] [-z ] [-v] [-help]
```

where:

`-c` *`license_file_list`* Use the specified license file(s).

`-l [+]`*`debug_log_path`*	Write debugging information to file *`debug_log_path`*. This option uses the letter `l`, not the numeral 1. Prepending *`debug_log_path`* with the + character appends logging entries. See Appendix H, “The Debug Log File,” for more information on this file.
`-2 -p`	Restricts usage of `lmdown`, `lmreread`, and `lmremove` to a FLEX*lm* administrator who is by default root. If there a UNIX group called “lmadmin,” then use is restricted to only members of that group. If root is not a member of this group, then root does not have permission to use any of the above utilities. If `-2 -p` is used when starting `lmgrd`, no user on Windows can shut down the license server with `lmdown`.
`-local`	Restricts the `lmdown` command to be run only from the same machine where `lmgrd` is running.
`-x lmdown`	Disable the `lmdown` command (no user can run `lmdown`). If `lmdown` is disabled, stop `lmgrd` via `kill` *`pid`* (UNIX) or stop the `lmgrd` and vendor daemon processes through the Windows Task Manager or Windows service. On UNIX, be sure the `kill` command does not have a `-9` argument. (v4.0+ `lmgrd`)
`-x lmremove`	Disable the `lmremove` command (no user can run `lmremove`). (v4.0+ `lmgrd`)

`-z`	Run in foreground. The default behavior is to run in the background. If `-l` *`debug_log_path`* is present, then no windows are used, but if no `-l` argument specified, separate windows are used for `lmgrd` and each vendor daemon.
`-v`	Displays `lmgrd` version number and copyright and exits.
`-help`	Displays usage information and exits.

6.2 Starting the License Manager Daemon on UNIX Platforms

If any licenses in the license file are counted (license count > 0), the license manager daemon, and hence the license server, must be started before the FLEX*lm* licensed application can be used.

The license manager daemon, lmgrd, is started either manually on the command line or automatically at system startup. Both methods are discussed in the following sections.

Note: Start `lmgrd` only on the server node specified on the SERVER line in the license file.

If you are running three-server redundant license servers, maintain a separate copy of the license file (as well as the `lmgrd` and the vendor daemons binaries) on each server node. If you do not do this, you lose all the advantages of having redundant servers, since the file server holding these files becomes a single point of failure.

6.2.1 Manually

Start `lmgrd` from the UNIX command line using the following syntax:

```
lmgrd -c license_file_list -L [+]debug_log_path
```

where

- *`license_file_list`* is one or more of the following:
  - the full path to a single license file

- a directory, where all files named `*.lic` in that directory are used

- *`debug_log_path`* is the full path to the debug log file

  Prepending *`debug_log_path`* with the + character appends logging entries.

Start `lmgrd` by a user other than “root”, since processes started by root can introduce security risks. If `lmgrd` must be started by the root user, use the `su` command to run `lmgrd` as a non-privileged user:

```
su username -c "lmgrd -c license_file_list -l \
                        debug_log_path"
```

where *`username`* is a non-privileged user. You must ensure that the vendor daemons listed in the license file have execute permissions for *`username`*. The paths to all the vendor daemons in the license file are listed on each VENDOR line.

6.2.2 Automatically

On UNIX, edit the appropriate boot script, which may be `/etc/rc.boot`, `/etc/rc.local`, `/etc/rc2.d/Sxxx`, `/sbin/rc2.d/Sxxxx`, etc. Include commands similar to the following. See the notes following for a full explanation.

```
/bin/su daniel -c 'echo starting lmgrd > \
/home/flexlm/v5.12/hp700_u9/boot.log'

/bin/nohup /bin/su daniel -c 'umask 022; \
/home/flexlm/v5.12/hp700_u9/lmgrd -c \
/home/flexlm/v5.12/hp700_u9/license.dat >> \
/home/flexlm/v5.12/hp700_u9/boot.log'

/bin/su daniel -c 'echo sleep 5 >> \
/home/flexlm/v5.12/hp700_u9/boot.log'

/bin/sleep 5

/bin/su daniel -c 'echo lmdiag >>\
/home/flexlm/v5.12/hp700_u9/boot.log'

/bin/su daniel -c '/home/flexlm/v5.12/hp700_u9/lmdiag -n -c\
/home/flexlm/v5.12/hp700_u9/license.dat >> \
/home/flexlm/v5.12/hp700_u9/boot.log'

/bin/su daniel -c 'echo exiting >>\
/home/flexlm/v5.12/hp700_u9/boot.log'
```

Please note the following about how this script was written:

- All paths are specified in full, because no paths are assumed at boot time.
- Because no paths are assumed, the vendor daemon must be in the same directory as `lmgrd`, or the VENDOR lines in the license file must be edited to include the full path to the vendor daemon.
- The `su` command is used to run `lmgrd` as a non-root user, "daniel." It is recommended that `lmgrd` not be run as "root," since it is a security risk to run any program as "root" that does not require root permissions. `lmgrd` does not require root permissions.
- Daniel has a `csh` login, so all commands executed as "daniel" must be in `csh` syntax. All commands not executed as "daniel" must be in `/bin/sh` syntax, since that is what is used by the boot scripts.
- The use of `nohup` and `sleep` are required on some operating systems, notably HP-UX and Digital UNIX. These are not needed on Solaris and some other operating systems, but are safe to use on all.
- `lmdiag` is used as a diagnostic tool to verify that the server is running and serving licenses.

Note: This does not start the daemon until you reboot your license server machine.

6.3 Starting the License Manager Daemon on Windows

6.3.1 Manually from the Command Line

`Start lmgrd` as an application from a Windows command shell using the following syntax:

```
C:\flexlm> lmgrd -c license_file_list -L [+]debug_log_path
```

where

- *`license_file_list`* is one or more of the following:
  - the full path to a single license file
  - a directory, where all files named `*.lic` in that directory are used
- *`debug_log_path`* is the full path to the debug log file

  Prepending *`debug_log_path`* with the + character appends logging entries.

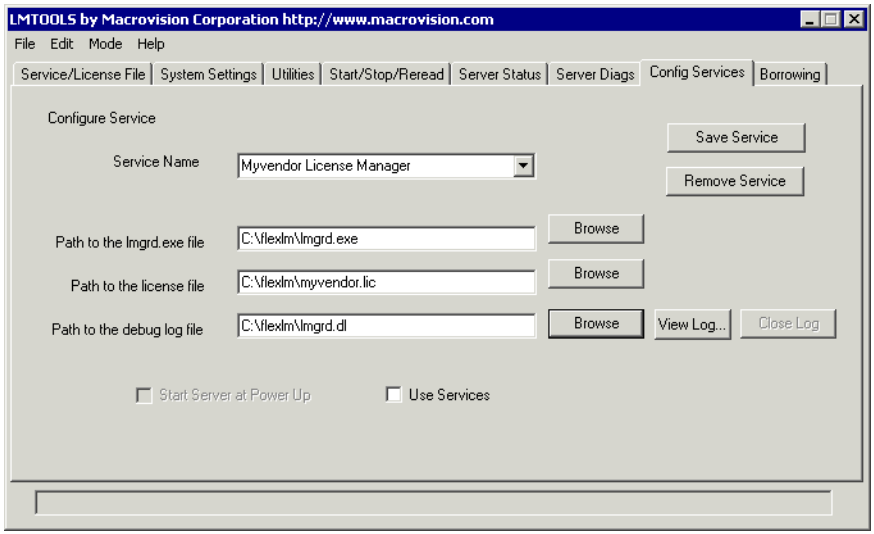
Spaces in pathnames require double quotes around the path.

On Windows, `lmgrd` can be installed as a service to allow it to be started and stopped through a user interface and run in the background.

6.3.2 Configuring the License Manager as a Windows Service

To configure a license server as a service, you must have Administrator privileges:

1. Run LMTOOLS.
2. Click the Configuration using Services radio button, then click the Config Services tab.
3. In the Service Name, type the name of the service that you want to define, for example, Myvendor License Manager.
4. In the Path to the lmgrd.exe file field, enter or browse to `lmgrd.exe` for this license server.
5. In the Path to the license file field, enter or browse to the license file for this license server.
6. In the Path to the debug log file, enter or browse to the debug log file that this license server writes. Prepending the debug log file name with the + character appends logging entries.



7. To save the new Myvendor License Manager service, click the Save Service button.

6.3.3 Manually from LMTOOLS

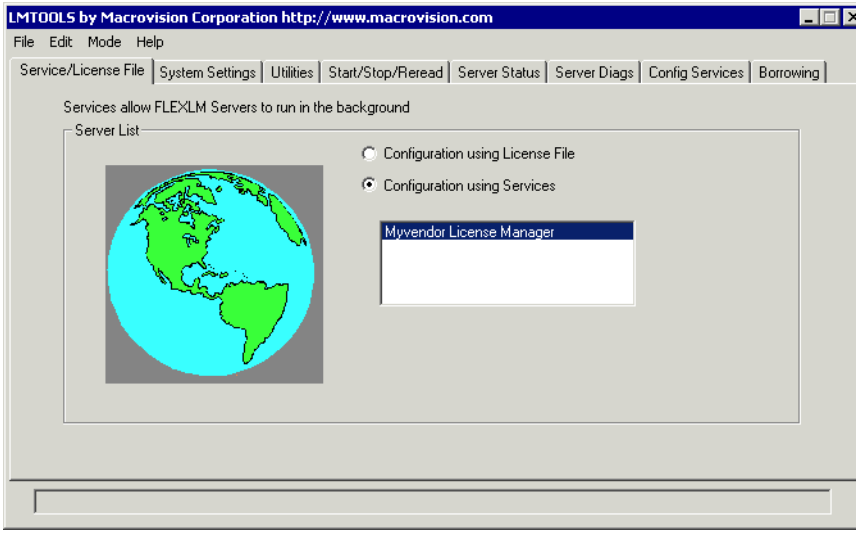
A graphical user interface to the license manager tools is provided called LMTOOLS. Some of the functions LMTOOLS performs include:

- starting, stopping, and configuring FLEX*lm* license servers
- getting system information, including hostids
- getting server status

In order to control the operation of `lmgrd` from the LMTOOLS user interface, you first must configure it as a license manager service. Follow the procedure in Section 6.3.2, “Configuring the License Manager as a Windows Service,” before proceeding.

Once the license manager service is configured, lmgrd is started by starting the service from the LMTOOLS interface:

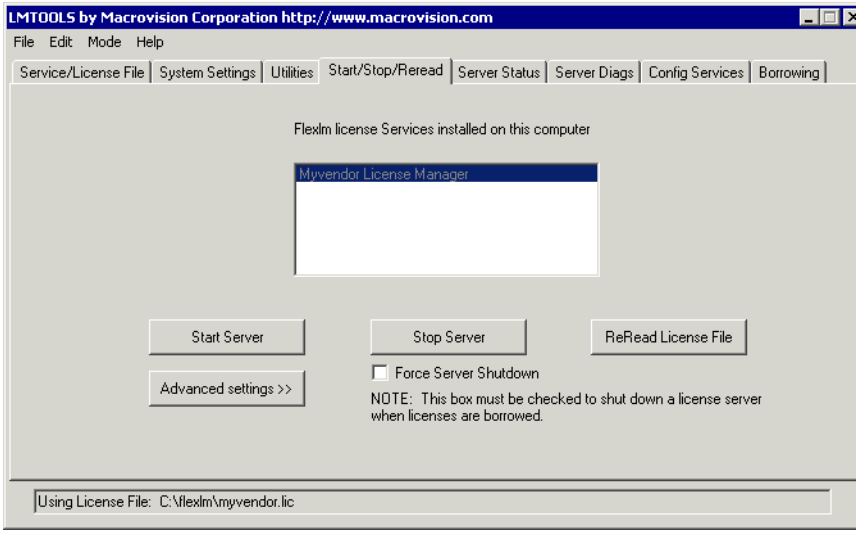
1. Start LMTOOLS.



   LMTOOLS appears with the Service/License File tab open.

2. Click the Configuration using Services radio button.
3. Select the service name from the list presented in the selection box. In this example, the service name is Myvendor License Manager.

4. Click the Start/Stop/Reread tab.



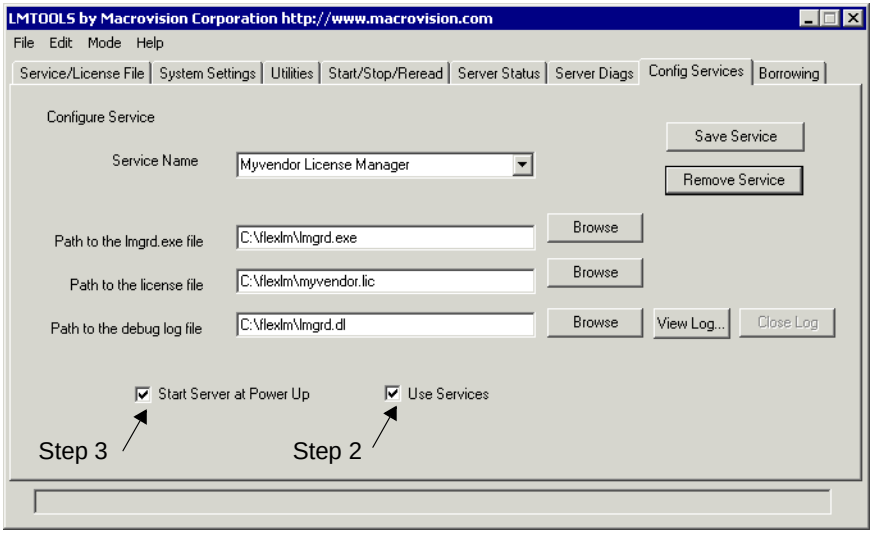
5. Start Myvendor License Manager by clicking the Start Server button.

Myvendor License Manager license server starts and writes its debug log output to `c:\flexlm\lmgrd.dl`.

6.3.4 Automatically at System Start Up

In order for lmgrd, and hence the license server, to start up automatically at system start up time, you first must configure it as a license manager service. Follow the procedure in Section 6.3.2, “Configuring the License Manager as a Windows Service,” before proceeding. Then continue:

1. With LMTOOLS started and the desired service name selected, click the Config Services tab.



2. Make this license manager a Windows service: check the Use Services check box (otherwise, it becomes a FLEX*lm* service).
3. Configure it to start at system startup time by checking the Start Server at Power Up check box.

From now on, when the machine is rebooted, this license manager starts automatically as a Windows service.

Chapter 7

License Administration Tools

FLEX*lm* provides utilities for the license administrator to help manage the licensing activities on the network. Always use the newest version of the utilities as possible; they are available for download from www.macrovision.com.

Table 7-1 lists these utilities.

Table 7-1: License Administration Utilities

Utility	Description
`lmborrow`	Supports license borrowing.
`lmdiag`	Diagnoses license checkout problems.
`lmdown`	Gracefully shuts down selected license daemons (both `lmgrd` and all vendor daemons) on the license server machine (or on all three machines in the case of three-server redundant servers).
`lmhostid`	Reports the hostid of a system.
`lminstall`	Converts license files between different formats.
`lmnewlog`	Moves existing report log information to a new file name and starts a new report log file with existing file name.
`lmpath`	Allows users direct control over license file path settings.
`lmremove`	Releases a hung license to the pool of free licenses.
`lmreread`	Causes the license daemon to reread the license file and start any new vendor daemons.

Table 7-1: License Administration Utilities (Continued)

Utility	Description
`lmstat`	Displays the status of a license server.
`lmswitch`	Controls debug log location and size.
`lmswitchr`	Switches the report log to a new file name.
`lmver`	Reports the FLEX*lm* version of a library or binary file.

FLEXLM VERSION NOTES

- The `lmpath` utility introduced in the v7.0 FLEX*lm* utilities.
- The `lmborrow` utility introduced in the v8.0 FLEX*lm* utilities.
- The `lmswitch` utility introduced in v8.0 vendor daemon.
- The `lmswitchr` utility introduced in v8.0 vendor daemon.

7.1 Running Administration Tools

All FLEX*lm* utilities are packaged as a single executable called `lmutil`. `lmutil` is either installed as individual commands (either by creating links to the individual command names, or making copies of `lmutil` as the individual command names), or as a wrapper that runs the individual command as `lmutil` *`command`*. For example, `lmutil lmstat`, or `lmutil lmdown`.

On Windows systems, the `lmutil` *`command`* form of the commands are available. There is also a graphical user interface available for these commands —see Section 7.16, “License Administration Tools—LMTOOLS for Windows.”

7.2 Universal lmutil Arguments

The following are valid arguments for most `lmutil` utilities:

`-c` *`license_file_path`*	Most `lmutil` utilities need to know the path to the license file. This is specified with a `-c` *`license_file_path`* argument, or by setting the `LM_LICENSE_FILE` environment variable. Otherwise, the default location is used. The utilities also honor all *`VENDOR`*`_LICENSE_FILE` environment variables. Some utilities take more than one license file path in a license-file list separated by colons on UNIX and semi-colons on Windows. Pathnames which include spaces have to be enclosed in double quotes.
`-help`	Displays usage information and exits.
`-v`	Displays the FLEX*lm* version of the utility and exits.
`-verbose`	Displays longer description for all errors found.

FLEXLM VERSION NOTES

- *`VENDOR`*`_LICENSE_FILE` environment variable honored in utilities starting with v7.0 FLEX*lm* utilities.
- `-verbose` option introduced in v6.0 of the FLEX*lm* utilities.

7.3 lmborrow

`lmborrow` supports borrowing of licenses that contain the BORROW attribute. It must be run on the machine where licenses are borrowed. It is used to perform the following:

- Initiating borrowing by setting the borrow period
- Clearing the borrow period
- Determining borrow status
- Returning a borrowed license early

INITIATING BORROWING

To initiate borrowing, the user sets the borrow period by running `lmborrow` from the command line or through LMTOOLS:

```
lmborrow {vendor | all} enddate [time]
```

where:

`vendor`	The vendor daemon name that serves the licenses to be borrowed, or `all` specifies all vendor daemons in that license server.
`enddate` `[`*`time`*`]`	Date the license is to be returned in *`dd-mmm-yyyy`* format. *`time`* is optional and is specified in 24-hour format (`hh:mm`) in the FLEX*lm*-licensed application's local time. If *`time`* is unspecified, the checkout lasts until the end of the given end date.

For example:

```
lmborrow sampled 20-aug-2001 13:00
```

This has the effect of setting `LM_BORROW` with the borrow period in either the registry (Windows) or in `$HOME/.flexlmrc` (UNIX).

To borrow licenses for the desired vendor, *on the same day and the same machine* that the user runs `lmborrow`, run the application(s) to check out the license(s). If you run the application(s) more than once that day, no duplicate licenses are borrowed. No licenses are borrowed if the application is run on a day different than the date borrowing is initiated.

In addition to the `lmborrow` utility, there are other ways to initiate borrowing:

- Using the borrowing interface in application, if provided in the application.
- Setting the `LM_BORROW` environment variable directly.

See Section 8.4.1, “Initiating License Borrowing,” for more information on these other ways.

Clearing the Borrowed License Setting

To clear the `LM_BORROW` setting in the registry or `$HOME/.flexlmrc`, issue the command:

```
lmborrow -clear
```

Clearing the `LM_BORROW` setting stops licenses from being borrowed until borrowing is initiated again. A user might run `lmborrow -clear` after he has borrowed licenses for features that are used offline if—before disconnecting from the network—he wants to run an application that checks out additional features, served by *`vendor,`* that are not meant to be borrowed. Clearing `LM_BORROW` does *not* change the status for already-borrowed licenses.

Determining Borrowed License Status

To print information about borrowed features, issue the following command on the machine from which they are borrowed:

```
lmborrow -status
```

The borrowing system does not have to be connected to the network to determine the status.

Returning a Borrowed License Early

To return a borrowed license early, first reconnect the borrowing system back to the network and then, from the same machine that initiated the borrowing, issue the command:

```
lmborrow -return [-c license_file_list]
                 [-c display] feature
```

where:

`-c` *`license_file_list`*	Use the specified license file(s). In some configurations, the license file needs to be specified in order to return the license file early.
`-d` *`display`*	Used to specify the display from which the borrow was initiated. Required if your current display is different than what was used to initiate the borrow. On Windows, it is the system name or, in the case of a terminal server environment, the terminal server client name. On UNIX, it is in the form `/dev/tty`*`xx`* or the X-Display name.
`feature`	The name of the borrowed feature to be returned early. Use `lmborrow -status` to get a list of borrowed feature names.

This option may or may not be allowed by your vendor. Check directly with your vendor to determine if early borrowed-license return is supported.

Returning the license early has the effect of clearing the LM_BORROW setting for the vendor daemon that serves the returned license.

If the borrowing system is not placed back on the network before attempting the early return, the license is not returned and LM_BORROW is kept intact. Additionally, an error message is issued to the end user with notification that the system needs to be connected to the network.

FLEXLM VERSION NOTES

- Early borrowed-license return introduced in v8.3 FLEX*lm* utilities.

7.4 lmdiag

`lmdiag` allows you to diagnose problems when you cannot check out a license.

Usage is:

```
lmdiag [-c license_file_list] [-n] [feature[:keyword=value]]
```

where:

`-c` *`license_file_list`*	Diagnose the specified file(s).
`-n`	Run in non-interactive mode; `lmdiag` does not prompt for any input in this mode. In this mode, extended connection diagnostics are not available.
`feature`	Diagnose this feature only.
`keyword=value`	If a license file contains multiple lines for a particular feature, select a particular line for `lmdiag` to report on. For example: `lmdiag f1:HOSTID=12345678` attempts a checkout on the line with the hostid "12345678." *`keyword`* is one of the following: VERSION, HOSTID, EXPDATE, KEY, VENDOR_STRING, ISSUER

If no *`feature`* is specified, `lmdiag` operates on all features in the license file(s) in your list. `lmdiag` first prints information about the license, then attempts to check out each license. If the checkout succeeds, `lmdiag` indicates this. If the checkout fails, `lmdiag` gives you the reason for the failure. If the checkout fails because `lmdiag` cannot connect to the license server, then you have the option of running "extended connection diagnostics."

These extended diagnostics attempt to connect to each TCP/IP port on the license server machine, and detects if the port number in the license file is incorrect. `lmdiag` indicates each TCP/IP port number that is listening, and if

it is an `lmgrd` process, `lmdiag` indicates this as well. If `lmdiag` finds the vendor daemon for the feature being tested, then it indicates the correct port number for the license file to correct the problem.

SEE ALSO

- Section C.2, “FLEXLM_DIAGNOSTICS.”

7.5 lmdown

The `lmdown` utility allows for the graceful shutdown of selected license daemons (both `lmgrd` and selected vendor daemons) on all machines.

Usage is:

```
lmdown -c license_file_list [-vendor vendor_daemon] [-q]
          [-all] [-force]
```

where:

`-c` *`license_file_list`*	Use the specified license file(s). Note that specifying `-c` *`license_file_list`* is always recommended with `lmdown`
`-vendor` *`vendor_daemon`*	Shut down only this vendor daemon. `lmgrd` continues running. Requires v6.0 `lmdown` and `lmgrd`.
`-q`	Don’t prompt or print a header. Otherwise `lmdown` asks “Are you sure? [y/n]: .”
`-all`	If multiple servers are specified, automatically shuts down all of them. `-q` is implied with `-all`.
`-force`	If licenses are borrowed, `lmdown` runs only from the machine where the license server is running, and then only if the user adds `-force`.

If `lmdown` encounters more than one server (for example if `-c` specifies a directory with many `*.lic` files) and `-all` is not specified, a choice of license servers to shut down is presented.

Note: On UNIX, do *not* use `kill -9` to shut down the license servers. On Windows, if you must use the Task Manager to kill the FLEX*lm* service, be sure to end the `lmgrd` process first, then all the vendor daemon processes.

To stop and restart a single vendor daemon, use `lmdown -vendor` *`vendor`*, then use `lmreread -vendor` *`vendor`* to restart the vendor daemon.

When shutting down a three-server redundant license server, there is a one-minute delay before the servers shut down. `lmdown` shuts down all three license servers of a set of redundant license servers. If you need to shut down one of a set of redundant license servers (not recommended because you are left with two points of failure), you must kill both the `lmgrd` and vendor daemon processes on that license server machine.

You can protect the unauthorized execution of `lmdown` when you start up the license manager daemon, `lmgrd.` Shutting down the servers causes users to lose their licenses.

SEE ALSO

- Section 6.1, “lmgrd Command-Line Syntax,” for details about securing access to `lmdown`
- Section 7.11, “lmreread.”

FLEXLM VERSION NOTES

- `-all` option introduced in the v7.0 FLEX*lm* lmdown utility.
- -force option introduced in the v8.0 FLEX*lm* lmdown utility.

7.6 lmhostid

The `lmhostid` utility returns the FLEX*lm* hostid of the current platform. Invoked without any arguments, `lmhostid` displays the default hostid type for current platform. Otherwise, the hostid corresponding to the requested *`type`* is displayed, if supported on the current platform.

Usage is:

```
lmhostid [-n] [-type]
```

Where:

`-n`	Only the hostid, itself, is returned as a string, which is appropriate to use with HOSTID= in the license file. Header text is suppressed.
`-type`	One of the following hostid types. If not specified, the default hostid for the current platform is displayed. See Appendix A, “Expected FLEXlm Hostids,” for a list of the default types.

Platform Dependent Hostids

`-ether`	Ethernet address.
`-string`	String id.
`-vsn`	Volume serial number. (Windows platforms only)
`-flexid`	Parallel or USB FLEX*id* hardware key identification. (Windows platforms only)
`-long`	32-bit hostid.

Platform Independent Hostids

`-user`	Current user name.
`-display`	Current display name. On Windows, it is the system name or, in the case of a terminal server environment, the terminal server client name. On UNIX, it is in the form `/dev/tty`*xx* or the X-Display name.
`-hostname`	Current host name.
`-internet`	IP address of current platform in the form `###.###.###.###`.

The output of this command looks as follows:

```
lmhostid - Copyright (c) 1989, 2002 Macrovision Corporation
The FLEXlm hostid of this machine is "69021c89"
```

SEE ALSO

- Appendix A, “Hostids for FLEXlm-Supported Machines.”

7.7 lminstall

Introduced in v6.0, `lminstall` is designed primarily for typing in decimal format licenses to generate a readable format license file.

Usage is:

```
lminstall [-i in_lic_file ] [-maxlen n] [-e err_file] \
          [-o out_lic_file] \
          [-overfmt {2 | 3 | 4 | 5 | 5.1 | 6 | 7 | 7.1 | 8}] \
          [-odecimal]
```

Normally, to convert from decimal to readable format, `lminstall` is used with no arguments; you are prompted for the name of the output license file. The default file name is today’s date in *yyyymmdd*`.lic` format. Move this file to the application’s default license file directory, if specified by the software vendor. Otherwise, use the `LM_LICENSE_FILE` or *VENDOR*`_LICENSE_FILE` environment variables to specify the directory where the `*.lic` files are located.

To finish entering, type `q` on a line by itself or enter two blank lines.

When an input file is specified with no output file specified, output goes to stdout; if neither input nor output file is specified, `lminstall` assumes that input comes from stdin and prompts the user for an output file name.

`lminstall` is also used to convert licenses from readable to decimal format, and between different versions of FLEX*lm* license formats.

To convert from readable to decimal:

```
lminstall -i in_lic_file -o out_lic_file -odecimal
```

To convert to FLEX*lm* v5.1 format:

```
lminstall -i in_lic_file -o out_lic_file -overfmt 5.1
```

To enforce a maximum line length of, for example, 50 characters:

```
lminstall -maxlen 50
```

Conversion errors are reported as necessary and can be written to a file by specifying `-e` *`err_file`*. `lminstall` has a limit of 1000 lines of input.

7.8 lmnewlog

The `lmnewlog` utility switches the report log file by moving the existing report log information to a new file, then starting a new report log with the original report log file name. If you rotate report logs with `lmnewlog` instead of `lmswitchr`, you do not have to change the file name in the REPORTLOG line of the vendor's options file. Requires a v7.1+ vendor daemon.

Usage is:

```
lmnewlog [-c license_file_list] feature renamed_report_log
```

or:

```
lmnewlog [-c license_file_list] vendor renamed_report_log
```

where:

`-c` *`license_file_list`*	Use the specified license file(s).
`feature`	Any feature in this license file.
`vendor`	Vendor daemon in this license file.
`renamed_report_log`	New file path where existing report log information is to be moved.

7.9 lmpath

The `lmpath` utility allows direct control over FLEX*lm* license path settings. It is used to add to, override, or get the current license path settings.

Usage is:

```
lmpath {-add | -override} {vendor | all} license_file_list
```

where:

`-add`	Prepends *`license_file_list`* to the current license-file list or creates the license-file list, if it doesn't exist, initializing it to *`license_file_list`*. Duplicates are discarded.

`-override`	Overrides the existing license-file list with *`license_file_list`*. If *`license_file_list`* is the null string, `""`, the specified list is deleted. • `lmpath -override all ""` Deletes the value of LM_LICENSE_FILE. • `lmpath -override` *`vendor`* `""` Deletes the value of *VENDOR*_LICENSE_FILE.
`vendor`	A vendor daemon name. Effects the value of *VENDOR*_LICENSE_FILE.
`all`	Refers to all vendor daemons. Effects the value of LM_LICENSE_FILE.
`license_file_list`	A colon-separated list on UNIX and a semi-colon-separated list on Windows. If *`license_file_list`* is the null string, `""`, then the specified entry is deleted.

Note: `lmpath` works by setting the FLEX*lm* registry entry on Windows or `$HOME/.flexlmrc` on UNIX.

To display the current license path settings, use:

```
lmpath -status
```

The following is displayed:

```
lmpath - Copyright (C) 1989-2002 Macrovision Corporation
Known Vendors:
_____________
demo:   ./counted.lic:./uncounted.lic
_____________
Other Vendors:
_____________
/usr/local/flexlm/licenses/license.lic
```

Note that where the path is set to a directory, all the `*.lic` files are listed separately.

7.10 lmremove

The `lmremove` utility allows you to remove a single user's license for a specified feature. If the application is active, it re-checks out the license shortly after it is freed by `lmremove`.

Usage is:

```
lmremove [-c license_file_list] feature user user_host display
```

or

```
lmremove [-c license_file_list] -h feature server_host \
            port handle
```

where:

`-c` *`license_file_list`*	Specify license file(s).
`feature`	Name of the feature checked out by the user.
`user`	Name of the user whose license you are removing, as reported by `lmstat -a`.
`user_host`	Name of the host the user is logged into, as reported by `lmstat -a`.
`display`	Name of the display where the user is working, as reported by `lmstat -a`.
`server_host`	Name of the host on which the license server is running.
`port`	TCP/IP port number where the license server is running, as reported by `lmstat -a`.
`handle`	License handle, as reported by `lmstat -a`.

The *`user`*, *`user_host`*, *`display`*, *`server_host, port`*, and *`handle`* information must be obtained from the output of `lmstat -a`.

`lmremove` removes all instances of *`user`* on *`user_host`* and *`display`* from usage of *`feature`*. If the optional `-c` *`license_file_list`* is specified, the indicated file(s) is used as the license file.

The `-h` variation uses the *`server_host`*, *`port`*, and license *`handle`*, as reported by `lmstat -a`. Consider this example `lmstat -a` output:

```
joe nirvana /dev/ttyp5 (v1.000) (cloud9/7654 102), start Fri 10/29 18:40
```

In this example, the user is “joe,” the user host is “nirvana,” the display is “/dev/typp5,” the server host is “cloud9,” the TCP/IP port is “7654,” and the license handle is “102.”

To remove this license, issue one of the following commands:

```
lmremove f1 joe nirvana /dev/ttyp5
```

or

```
lmremove -h f1 cloud9 7654 102
```

When removing by handle, if licenses are grouped as duplicates, all duplicate licenses are also removed. If license lingering is set and `lmremove` is used to reclaim the license, `lmremove` starts, but does not override, the license’s linger time.

You can protect the unauthorized execution of `lmremove` when you start up the license manager daemon, `lmgrd,` because removing a user’s license is disruptive.

SEE ALSO

- Section 6.1, “lmgrd Command-Line Syntax,” for details about securing access to `lmremove`

7.11 lmreread

The `lmreread` utility causes the license manager daemon to reread the license file and start any new vendor daemons that have been added. In addition, all currently running vendor daemons are signaled to reread the license file and their end-user options files for changes in feature licensing information or option settings. If report logging is enabled, any report log data still in the vendor daemon’s internal data buffer is flushed. `lmreread` recognizes changes to server machine host names, but cannot be used to change server TCP/IP port numbers.

If the optional vendor daemon name is specified, only the named daemon rereads the license file and its end-user options file (in this case, `lmgrd` does not reread the license file).

Usage is:

```
lmreread [-c license_file_list] [-vendor vendor] [-all]
```

where:

`-c license_file_list`	Use the specified license file(s).
`-vendor vendor`	Only this one vendor daemon rereads the license file. `lmgrd` restarts the vendor daemon if necessary.
`-all`	If more than one `lmgrd` is specified, instructs all `lmgrd`s to reread.

To stop and restart a single vendor daemon, use `lmdown -vendor vendor`, then use `lmreread -vendor vendor`, which restarts the vendor daemon.

Note: If you use the `-c license_file_list` option, the license file(s) specified are read by `lmreread`, not by `lmgrd`; `lmgrd` rereads the file it read originally.

You can protect the unauthorized execution of `lmreread` when you start up the license manager daemon, `lmgrd`.

SEE ALSO

- Section 6.1, “lmgrd Command-Line Syntax,” for details about securing access to `lmreread`

FLEXLM VERSION NOTES

- Ability for vendor daemon to participate in rereading of its option file introduced in v8.0 vendor daemon

7.12 lmstat

The `lmstat` utility helps you monitor the status of all network licensing activities, including:

- Daemons that are running
- License files
- Users of individual features
- Users of features served by a specific vendor daemon
- BORROW licenses borrowed

`lmstat` prints information that it receives from the license server; therefore, it does not report on unserved licenses such as uncounted licenses. To report on an uncounted license, the license must be added to a served license file and the application must be directed to use the license server for that license file (via `@host`, `port@host` or USE_SERVER). Queued users and licenses shared due to duplicate grouping are also not returned by `lmstat`.

Usage is:

```
lmstat [-a] [-c license_file_list] [-f [feature]]
       [-i [feature] [-s[server] [-S [vendor]]
       [-t timeout_value]
```

where:

`-a`	Displays all information.
`-c license_file_list`	Uses the specified license file(s).
`-f [feature]`	Displays users of `feature`. If `feature` is not specified, usage information for all features is displayed.
`-i [feature]`	Displays information from the FEATURE/INCREMENT line for the specified `feature`, or all features if `feature` is not specified.
`-s [server]`	Displays status of all license files listed in `$VENDOR_LICENSE_FILE` or `$LM_LICENSE_FILE` on `server`, or on all servers if `server` is not specified.

`-S [vendor]`	Lists all users of *vendor*'s features.
`-t timeout_value`	Sets connection timeout to *timeout_value*. This limits the amount of time `lmstat` spends attempting to connect to *server*.

The output of `lmstat -a` looks similar to:

```
License server status: 27000@myhost1
License file(s) on myhost:
install_dir/flexlm/v9.0/sun4_u5/counted.lic:
myhost: license server UP (MASTER) v9.0
Vendor daemon status (on myhost1):

demo: UP v9.0
Feature usage info:
Users of f1:  (Total of 4 licenses issued;  Total of 1 license in use)
  "f1" v1.0, vendor: demo
  floating license
    daniel myhost2 19.16.18.26 (v1.0) (myhost1/27000 102), start Fri
         5/3 7:29
```

where:

daniel	*user*	User name.
myhost2	*user_host*	Host where user is running.
19.16.18.26	*display*	Display where user is running.
v1.0	*version*	Version of feature.
myhost1	*server_host*	Host where license server is running.
27000	*port*	TCP/IP port on *server_host* where license server is running.
102	*handle*	License handle.
start Fri 5/3 7:29	*checkout_time*	Time that this license was checked out.

The *user*, *user_host*, *display*, *server_host, port*, and *handle* information is used when removing licenses with `lmremove`.

Note: `lmstat -a` is a potentially expensive command. With many active users, this command generates a lot of network activity.

FLEXLM VERSION NOTES

- Ability to list all active users, using the `-A` option, available in FLEX*lm* `lmstat` utility, v8.3 and earlier.

7.13 lmswitch

The `lmswitch` utility switches the debug log file written by a particular vendor daemon by closing the existing debug log for that vendor daemon and starting a new debug log for that vendor daemon with a new file name. It also starts a new debug log file written by that vendor daemon if one does not already exist.

Usage is:

```
lmswitch [-c license_file_list] vendor new_debug_log
```

where:

`-c` *license_file_list*	Use the specified license file(s).
vendor	Vendor daemon in this license file.
new_debug_log	Path to new debug log file.

By default, debug log output from `lmgrd` and all vendor daemons started by that `lmgrd` get written into the same debug file. `lmswitch` allows companies to keep separate log files for different vendors and control the size of their debug log file.

If debug log output is not already directed to a separate file for this vendor daemon, `lmswitch` tells the vendor daemon to start writing its debug log output to a file, *`new_debug_log`*. If this vendor daemon is already writing to its own debug log, `lmswitch` tells the vendor daemon to close its current debug log file and start writing its debug log output to *`new_debug_log`*.

Note: The effect of `lmswitch` continues only until the vendor daemon is shut down or its options file is reread via `lmreread`. When the vendor daemon is restarted or its options file is reread, it looks for a DEBUGLOG line in the options file to determine whether or not to write its debug log output into its own file and, if so, what file to write.

SEE ALSO:

- Section 5.2.2, “DEBUGLOG”
- Section 7.11, “lmreread”
- Appendix H, “The Debug Log File”

7.14 lmswitchr

The `lmswitchr` utility switches the report log file by closing the existing report log and starting a new report log with a new file name. It also starts a new report log file if one does not already exist.

Usage is:

```
lmswitchr [-c license_file_list] feature new_report_log
```

or with v5.0+ vendor daemon:

```
lmswitchr [-c license_file_list] vendor new_report_log
```

where:

`-c` *`license_file_list`*	Use the specified license file(s).
`feature`	Any feature in this license file.
`vendor`	Vendor daemon in this license file.
`new_report_log`	Path to new report log file.

If report logging is not enabled for the vendor daemon, `lmswitchr` tells it to start writing its report log output to *`new_report_log`*. If report logging is already enabled for the vendor daemon, `lmswitchr` tells the vendor daemon to close its report log file and start writing its new report log output to *`new_report_log`*.

Note: The effect of `lmswitchr` continues only until the vendor daemon is shut down or its options file is reread via `lmreread`. When the vendor daemon is restarted or its options file is reread, it looks for a REPORTLOG line in the options file to determine whether or not to write report log output to a file and, if so, what file to write.

SEE ALSO:

- Section 5.2.15, “REPORTLOG”
- Section 7.8, “lmnewlog”
- Section 7.11, “lmreread”
- Appendix G, “The Report Log File”

7.15 lmver

The `lmver` utility reports the FLEX*lm* version of a library or binary file.

Usage is:

```
lmver filename
```

where *`filename`* is one of the following:

- the name of an executable file built with FLEX*lm*
- `lmgrd`
- a license administration tool
- a vendor daemon

For example if you have an application called “spell,” type:

```
lmver spell
```

7.16 License Administration Tools—LMTOOLS for Windows

For the 32-bit Windows platforms, a graphical user interface to the license manager tools is provided called LMTOOLS. Always use the newest version of LMTOOLS as possible; it is available for download from www.macrovision.com.

Some of the functions LMTOOLS performs include:

- starting, stopping, and configuring FLEX*lm* license servers
- getting system information, including hostids
- getting server status

LMTOOLS has two modes in which to configure a license server:

- Configuration using a license file
- Configuration using services

7.16.1 Configuration Using License File

Operations are performed on a particular license file. The file can be either local or remote. In this mode, you cannot start the `lmgrd` process, but you can do everything else. To configure this mode, perform the following:

1. Invoke LMTOOLS.
2. Click the Configuration using License File radio button.
3. Enter one or more the license file names or *`port@host`* specifications.

7.16.2 Configuration Using Services

Operations are performed on a service, which allows starting `lmgrd` processes local to the system on which LMTOOLS is running. On NT/2000/XP, you configure a Windows Service. On Windows 95/98, you configure a FLEX*lm* service that behaves similarly to a Windows service.

For details on configuring services, see Section 6.3.2, “Configuring the License Manager as a Windows Service.”

Chapter 8

Mobile Licensing

End users often want to use applications on computers that do not have a continuous connection to a FLEX*lm* license server. These situations include:

- Working on a laptop
- Using a computer both at work and at home
- Working from several different computers not connected to a license server

FLEX*lm* supports licenses that allow one of several kinds of mobile licensing:

- Node-locked to a laptop
- Node-locked to a FLEX*id* (Windows only)
- Node-locked to a FLEX*id* with FLOAT_OK keyword (Windows only)
- License borrowing with BORROW keyword
- Node-locked to a user name
- Fulfilled from a prepaid license pool

License rehosting is the consequence of an end user wanting to move a license without using one of these methods. This means a new node-locked license file is generated by the vendor for each new client computer. Rehosting incurs administrative overhead because the vendor is involved for each move.

8.1 Node-Locked to a Laptop Computer

If a license is to be used exclusively on one laptop computer, that license is simply node-locked to an address associated with that computer. The license file resides on the laptop computer.

8.2 Node-locked to a FLEX*id* (Windows Only)

If a license is to be moved between different Windows machines, it is node-locked to a FLEX*id* (a dongle that connects to a parallel or USB port). This license is moved between machines by installing a copy of the license file on

each machine and moving the FLEX*id* from one machine to another. Since the license is tied to the FLEX*id*, only the machine with the FLEX*id* has use of the license.

8.3 Node-Locked to a FLEX*id* with FLOAT_OK (Windows Only)

This method of license mobility has an advantage over simply using a license node-locked to a FLEX*id*, because the FLEX*id* is attached to a license server machine and its license floats on the network. Licenses with a FLOAT_OK keyword that are node-locked to a FLEX*id* are supported only where both the FLEX*lm*-licensed application and the license server are running on Windows.

A vendor issues a license file with a FEATURE line node-locked to a FLEX*id* and containing the FLOAT_OK keyword and a FLEX*id* for that FEATURE line. One FEATURE line containing the FLOAT_OK keyword and one FLEX*id* is needed for each instance of a license that is mobile. When the FLEX*id* is attached to a license server, the license floats on the network. When the FLEX*id* is removed from the license server, the license is available only on the standalone computer.

This method supports parallel or USB FLEX*id*s. Because it is simpler to attach multiple USB dongles to a computer, USB FLEX*id*s may be preferable.

8.3.1 Initiating FLEX*id* with FLOAT_OK

A vendor issues the end user a FLEX*id*, a FLEX*id* driver installer, and a license file that contains a FEATURE line node-locked to that FLEX*id* containing the FLOAT_OK keyword. An end user then:

1. Installs the license file on the license server machine
2. Attaches all of the FLEX*id*s to the license server machine
3. Installs the FLEX*id* driver on the license server machine
4. Starts the license server or rereads the license file

While the FLEX*id*s are attached to the license server machine, the node-locked licenses associated with them float on the network. Each of the FLOAT_OK uncounted node-locked FEATURE lines has a count of *one* while it is available on the network.

To transfer a license from the pool of floating licenses to a disconnected computer, the end user:

1. Copies the license file containing the FLOAT_OK node-locked FEATURE line from the license file on the license server machine to a license file on the client computer in the location where the licensed application expects to find its license file.
2. Moves the FLEX*id* matching the node-locked FEATURE line from the license server machine to the client computer. When the FLEX*id* is removed from the license server machine, this license is unavailable on the network.
3. Installs the FLEX*id* drivers on the client computer, if they are not already installed.
4. Disconnects the client computer from the network. Now the license is available on the computer with the FLEX*id*, even though that computer is disconnected from the network.

8.3.2 Returning a FLEX*id* with FLOAT_OK License

To return the license to the license server machine so it floats on the network again, the end user:

1. Removes the FLEX*id* from the client machine and replaces it on the license server machine.
2. Rereads the license file for the license server that serves the floating version of the license by running `lmreread`. When the FLEX*id* is returned to the license server machine, the FLOAT_OK license does not float on the network again until `lmreread` is run.

8.3.3 FLEX*id* with FLOAT_OK Example

The following is a sample license file issued to a customer site. It is shipped with two FLEX*id*s: `FLEXID=7-b28520b9` and `FLEXID=7-b2857678`.

```
SERVER myhost ANY
VENDOR sampled
FEATURE f1 sampled 1.0 permanent uncounted FLOAT_OK \
        HOSTID=FLEXID=7-b28520b9 SIGN=123456789012
FEATURE f1 sampled 1.0 permanent uncounted FLOAT_OK \
        HOSTID=FLEXID=7-b2857678 SIGN=ABCDEF123456
```

The customer installs the license file and the two FLEX*id*s on the license server machine. When attached to the license server machine, each uncounted FLOAT_OK license floats on the network and allows a single use. Therefore, up to two users can use “f1” on the customer’s network, except on the license server machine itself, where the license use is disallowed.

If an end user wants to work at home, the end user installs a license file that contains the FEATURE line node-locked to `FLEXID=7-b28520b9` (this only needs to be done once), transfers the FLEX*id* `FLEXID=7-b28520b9` from the license server machine to the client computer, and installs the FLEX*id* driver on the client computer (this also only needs to be done once). The end user disconnects the client computer from the network and uses the transferred FLOAT_OK license on the client computer. The license server allows only the single remaining FLOAT_OK license to float on the network.

After returning the FLEX*id* to the license server machine, the end user (or the system administrator) runs `lmreread` so the returned license can float again.

FLEXLM VERSION NOTES

- FLOAT_OK keyword introduced in v8.0 FLEX*lm* client library, license manager daemon, and vendor daemon. All components must be v8.0+ in order to use FLOAT_OK.

8.4 License Borrowing with BORROW

If a license is to be used on a computer that is intermittently connected to a license server, that license can be issued as a floating license with the BORROW keyword. A BORROW license can be borrowed from a license server via a special checkout and used later to run an application on a computer that is no longer connected to the license server. License borrowing must be enabled by a vendor before an end user can borrow licenses.

With license borrowing, a vendor issues a floating license with a FEATURE line that contains the BORROW keyword. An end user specifies the expiration date a borrowed license is to be returned and runs the application while connected to the network which writes borrowing information on the client computer. The license server keeps the borrowed license checked out. The FLEX*lm*-licensed application automatically uses the local borrowing data to do checkouts during the borrow period. If enabled by the vendor, borrowed licenses can be returned early, that is, before the borrow period expires. Upon the earlier of either the expiration of the borrow period or the early return of a borrowed license, the local borrowing data no longer authorizes checkouts and the license server returns the borrowed license to the pool of available licenses. No clock synchronization is required between the license server machine and the machine running the FLEX*lm*-licensed application.

8.4.1 Initiating License Borrowing

If a vendor has enabled license borrowing by issuing a license file that contains a FEATURE line with the BORROW keyword, an end user initiates license borrowing in one of three ways:

- Using the borrowing interface in application, if provided in the application
- Running the `lmborrow` utility to set `LM_BORROW`
- Setting the `LM_BORROW` environment variable directly

Application Interface

The user initiates license borrowing this way only if the application provides a borrowing interface. Information about this is supplied by the vendor.

Running the lmborrow Utility

`lmborrow` is one of the `lmutil`/LMTOOLS utilities. To initiate borrowing, the user runs `lmborrow` from the command line or through LMTOOLS:

```
lmborrow {vendor|all} enddate [time]
```

where *`vendor`* is the vendor daemon that serves the licenses to be borrowed, or `all` specifies all vendor daemons in the license server. *`enddate`* is the date the license is to be returned in *`dd-mmm-yyyy`* format. *`time`* is optional and is specified in 24-hour format (*`hh`*:*`mm`*) in the FLEX*lm*-licensed application's local time. If *`time`* is unspecified, the checkout lasts until the end of the given end date.

For example:

```
lmborrow sampled 20-aug-2001 13:00
```

Setting the LM_BORROW Environment Variable Directly

The `lmborrow` utility is a user interface to set `LM_BORROW` in either the registry (Windows) or in `$HOME/.flexlmrc` (UNIX). `LM_BORROW` can also be set directly as an environment variable:

```
today:{vendor|all}:enddate[:time]
```

where:

`today`	Today's date in *`dd-mmm-yyyy`* format. Any checkouts done on this date create local borrow information. If a checkout is done on a different date than this date, no local borrowing information is created.

vendor	Vendor daemon that serves the licenses to be borrowed, or `all` specifies all vendor daemons in the license server.
enddate	Date the license is to be returned in *dd-mmm-yyyy* format.
time	Optional. *time* is specified in 24-hour format (*hh*:*mm*) in the FLEX*lm*-licensed application's local time. If *time* is unspecified, the checkout lasts until the end of the given end date.

For example:

```
LM_BORROW=15-aug-2001:sampled:20-aug-2001:13:00
```

In this example, one or more licenses served by the `sampled` vendor daemon are borrowed on August 15, 2001, and are scheduled to be returned at 1 pm on August 20, 2001.

8.4.2 Borrowing a License

To borrow a license for a desired feature, *on the same day and the same machine* that the end user runs `lmborrow` or sets `LM_BORROW` (and while still connected to the network), the end user runs the application to check out and borrow the license. If the end user runs the application more than once that day, no duplicate license is borrowed. No license is borrowed if the application is run on a day different than the date borrowing was set to be initiated.

For example, say that today you want to borrow a license for the PageWizard feature for a week. The PageWizard feature is served by the `sampled` vendor daemon. Today, while you are connected to the network, run `lmborrow` or set `LM_BORROW` directly. For example:

```
lmborrow sampled enddate
```

Today, after you run `lmborrow`, while you are connected to the network, run the application that checks out a license for the PageWizard feature. After the license is checked out, close the application and disconnect your machine from the network. The license that you just checked out stays checked out from the license server until the borrow period expires—that license now is used on your disconnected machine until the borrow period expires. The borrowed

license cannot be returned before the end of the borrow period. Once checked out, it remains checked out for the full borrow period. The borrow period cannot be renewed until the period has expired.

Clearing the Borrow Period

Once you have borrowed all the licenses that you need for the current borrow period (defined by the `LM_BORROW` environment variable), prevent licenses for any additional features from being borrowed by running `lmborrow -clear`. This clears the LM_BORROW setting in the registry (Windows) or `$HOME/.flexlmrc` (UNIX). `lmborrow -clear` does *not* clear the local information about licenses you have already borrowed.

Checking Borrow Status

To print information about borrowed features, issue the following command on the machine from which they are borrowed:

```
lmborrow -status
```

The borrowing system does not have to be connected to the network to determine the status.

Returning a Borrowed License Early

To return a borrowed license before the borrow period expires, first reconnect the borrowing system back to the network and then, from the same machine that initiated the borrowing, issue the command:

```
lmborrow -return [-c license_file_list] feature
```

This option may or may not be allowed by your vendor. Check directly with your vendor to determine if early borrowed-license return is supported.

Returning the license early has the effect of clearing the LM_BORROW setting for the vendor daemon that serves the returned license.

8.4.3 Support for License Borrowing

See the following sections for more information about the utilities and end-user options that support license borrowing:

- Section 7.3, “lmborrow”
- Section 7.5, “lmdown”
- Section 7.12, “lmstat”
- Section 5.2.1, “BORROW_LOWWATER”
- Section 5.2.4, “EXCLUDE_BORROW”

- Section 5.2.9, “INCLUDE_BORROW”

FLEXLM VERSION NOTES

- BORROW keyword introduced in v8.0 FLEX*lm* client library, license manager daemon, and vendor daemon. All components must be v8.0+ in order to use BORROW.

8.5 Node-locked to a User Name

If a license is to be used exclusively by one user on different machines, that license can be node-locked to the user’s user name. The license file is copied to the different machines on which the user might work; the user’s user name must be identical on each machine. For this method to be useful, individual user names in an organization need to be unique.

8.6 Fulfilled from a Prepaid License Pool

In this method, the end user buys a prepaid number of license-days from the vendor. The end user can then fulfill a license using a partial amount of the total license-days for the given borrow period, node-locked to a particular machine. For example, in preparation for a business trip (or even during a business trip), the end user fulfills a license that expires in 5 days that is node-locked to their laptop. Each fulfillment can be node-locked to a different machine (or even multiple times to the same machine), thus allowing mobility of license usage within the pre-paid number of license-days.

This model is like pay-per-use because each fulfillment is made from a decreasing number license-days. It is different than other pay-per-use models because, once node-locked to a machine, that machine is allowed unlimited use of the application until the license expires. This short-term license cannot be returned early; once fulfilled, those license-days cannot be refunded. Other pay-per-use models charge based on the number of times the application is used.

Appendix A

Hostids for FLEX*lm*-Supported Machines

FLEX*lm* uses different machine identifications for different machine architectures. For example, all Sun Microsystems machines have a unique hostid, whereas all DEC machines do not. For this reason, the ethernet address is used on some machine architectures as the hostid. An ethernet address is a 6-byte quantity, with each byte specified as two hexadecimal digits. Specify all twelve hex digits when using an ethernet address as a hostid. For example, if the ethernet address is “8:0:20:0:5:ac,” specify “0800200005ac” as the hostid.

A.1 Hostid Formats

Numeric, 32-bit hostids are normally used in hexadecimal format. On some systems, the system command returns the ID in decimal format. Use a “#” before the hostid to indicate a decimal number. For example, if the system command returns “2005771344,” FLEX*lm* accepts “#2005771344.” Alternatively, convert the decimal value to hexadecimal.

A.2 Expected FLEX*lm* Hostids

The `lmhostid` utility prints the exact hostid that FLEX*lm* expects to use on any given machine. The following table lists alternate methods to obtain the required hostid for each machine architecture. FLEX*lm* also supports a group of special hostids and vendor-defined hostids.

Hardware Platform	Hostid	Type this command on the license server:	Example
AIX (RS/6000, PPC)	32-bit hostid	`uname -m` (returns 000276513100), then remove last two digits, and use remaining last eight digits	02765131
DEC Alpha	ethernet address	`netstat -i`	080020005532
HP (32-bit and 64-bit platforms, non-Itanium)	32-bit hostid	`uname -i` and convert to hex, or prepend with #	778DA450 or #2005771344
HP (64-bit Itanium	machine identification	getconf \ CS_PARTITION_ IDENT then prefix with “ID_STRING=”	ID_STRING=9c766319-db72-d411-af62-0060b05e4c05
Linux	ethernet address	`/sbin/ifconfig eth0` and remove colons from HWaddr 00:40:05:16:E5:25	00400516E525
SCO	Hostid String	`uname -x` (Serial is SCO00354), then prefix with “ID_STRING=”	ID_STRING=SCO00354

Hardware Platform	Hostid	Type this command on the license server:	Example
SGI	32-bit hostid	`/etc/sysinfo -s`, convert to hex, or prefix #	69064C3C or #1762020412
SUN	32-bit hostid	`hostid`	170a3472
Windows	ethernet address	`lmutil lmhostid`	00B0A9DF9A32
	Disk serial number	DIR C: (look for "Volume Serial Number is", and remove "-")	DISK_SERIAL_NUM= 3e2e17fd
	FLEX*id* parallel or USB port hardware key (dongle)	`lmhostid -flexid`	FLEXID=7-b28520b9
		FLEX*id*s are made available by your vendor. Your vendor can also provide you with an installer that installs drivers for all FLEX*id*s.	

A.3 Special FLEX*lm* Hostids

FLEX*lm* contains a number of special hostid types which apply to all platforms. These hostid types are valid to use in both SERVER lines and FEATURE lines, wherever a hostid is required. These are:

`ANY`	Locks the software to any machine (i.e., does not lock anything).
`DEMO`	Similar to ANY, but only for use with uncounted FEATURE lines.
`COMPOSITE=` *`composit_hostid`*	Locks the software to a composite hostid. A composite hostid is a hashed 12-character hexidecimal value formed by combining the values of one or more simple hostids types, as defined by the software vendor..

`DISPLAY=`*`display`*	Locks the software to display *`display`*. On UNIX, *`display`* is `/dev/tty`*`xx`* (which is always `/dev/tty` when an application is run in the background) or the X-Display name. On Windows, it is the system name or, in the case of a terminal server environment, the terminal server client name. (v8+ licensed applications only)
`HOSTNAME=`*`host`*	Locks the software to computer host name *`host`*.
`ID=`*`n`*	Functionally equivalent to the “ANY” hostid—it runs on any machine. The difference is that the license is unique and is used to identify the customer. This hostid is used to lock the license server (on the SERVER line) or the licensed application (on the FEATURE/INCREMENT line). The number can have dashes included for readability—the dashes are ignored. Examples: ID=12345678 is the same as ID=1234-5678 is the same as ID=1-2-3-4-5-6-7-8
`INTERNET=` *`###.###.###.###`*	Locks the software to an Internet IP address, or group of IP addresses. Wildcards are allowed. For example, 198.156.*.* means any host with a matching internet IP address. The main use is to limit usage access by subnet, implying geographic area. For this purpose, it is used on the FEATURE/INCREMENT line as a hostid lock.
`USER=`*`user`*	Locks the software to user name *`user`*.

EXAMPLES

```
FEATURE f1 demo 1.0 1-jan-2005 uncounted \
        HOSTID=FLEXID=6-a6300015f SIGN=AB28E0011DA1
```

or

```
FEATURE f1 demo 1.0 1-jan-2005 uncounted \
        HOSTID=INTERNET=10.10.10.* SIGN=EB78201163B0
```

Appendix B

License File Format

License files usually begin with a SERVER line (or three lines for three-server redundant servers) followed by one or more VENDOR lines, followed by one or more FEATURE or INCREMENT lines. In some cases the license file requires no SERVER line and no VENDOR line.

You can modify these elements in the license file:

- Host names on the SERVER line(s)
- TCP/IP port numbers on the SERVER line(s)
- Paths on the VENDOR line(s)
- Options file paths on the VENDOR line(s)
- Optional TCP/IP port numbers on the VENDOR line(s) (for firewall support only)
- USE_SERVER line
- Values in `keyword=value` pairs on FEATURE lines, if `keyword` is specified in lowercase

Use the "\" line-continuation character to break up long lines.

8-bit Latin-based characters are fully supported in license files, options files, log files, and FLEX*lm*-licensed application environments.

See Section 4.4, "Counted vs. Uncounted Licenses," for more information on SERVER and VENDOR line requirements.

FLEXLM VERSION NOTES

- USE_SERVER introduced in the v5.0 FLEX*lm* client library.
- Line-continuation character is required with the pre-v7.0 FLEX*lm* client library.
- 8-bit Latin-based character support introduced in the v8.0 FLEX*lm* client library.

B.1 License File Syntax

B.1.1 Sample License File

This is an example of a license file for a single vendor with two features.

```
SERVER my_server 17007ea8 1700
VENDOR sampled
FEATURE f1 sampled 1.000 01-jan-2005 10 SIGN=9BFAC0316462
FEATURE f2 sampled 1.000 01-jan-2005 10 SIGN=1B9A308CC0F7
```

The license file above allows the license server “my_server” with the hostid “17007ea8” to serve ten floating licenses for each feature, “f1” and “f2,” to any user on the network.

B.1.2 SERVER Lines

The SERVER line specifies the host name and hostid of the license server and the TCP/IP port number of the license manager daemon (`lmgrd`). Normally a license file has one SERVER line. Three SERVER lines mean that you are using a three-server redundant license server. The absence of a SERVER line means that every FEATURE and INCREMENT line in the license file is uncounted.

The hostids from the SERVER lines are computed into the license key or signature on every FEATURE and INCREMENT line. For this reason, make sure you keep SERVER lines together with any FEATURE/INCREMENT lines as they were sent from the vendor.

The format of the SERVER line is:

```
SERVER host hostid [port]
```

where:

Field	Description
host	The system host name or IP address. String returned by the UNIX `hostname` or `uname -n` command. On NT/2000/XP, `ipconfig /all`; on Windows 95/98/ME, `winipcfg /all` return the host name.
hostid	Usually the string returned by the `lmhostid` command. This is changed only by your software supplier.

Field	Description
port	TCP/IP port number to use. A valid number is any unused port number between 0 and 64000. On UNIX, choose a port >1024, since those <1024 are privileged port numbers. If no TCP/IP port number is specified, one of the default ports in the range of 27000 and 27009 is used.

Example:

```
SERVER my_server 17007ea8 21987
```

SEE ALSO

- Section B.1.5, “FEATURE/INCREMENT Lines,” for more information about uncounted features.
- Chapter 4, “Selecting Server Machines,” for more information about redundant servers.

FLEXLM VERSION NOTES

- IP address specification for *host* introduced in v5.0 `lmgrd`, vendor daemon, and FLEX*lm* client library.
- *port* specification is required with a pre-v6.0 `lmgrd`, vendor daemon, and FLEX*lm* client library.

B.1.3 VENDOR Lines

The VENDOR line specifies the daemon name and path. `lmgrd` uses this line to start the vendor daemon, and the vendor daemon reads it to find its options file. The format of the VENDOR line is shown below.

```
VENDOR vendor [vendor_daemon_path]\
                [[options=]options_file_path] [[port=]port]
```

where:

Field	Description
vendor	Name of the vendor daemon used to serve some feature(s) in the file. This name cannot be changed by the administrator.

Field	Description
vendor_daemon_path	Optional path to the executable for this daemon. Generally the license administrator is free to install the daemon in any directory. (It is recommended, however, that it be installed in a local directory on the license server machine.) If omitted, `lmgrd` looks for the vendor daemon binary in • the current directory • the path specified in `lmgrd's` $PATH environment variable • in the directory where `lmgrd` is located If *vendor_daemon_path* is blank, then any options or TCP/IP port number specifications require the `options=` and `port=` strings.
options_file_path	Full path to the end-user options file for this daemon. FLEX*lm* does not require an options file. If omitted, the vendor daemon, by default, looks for a file called *vendor*`.opt` (where *vendor* is the vendor daemon name) located in the same directory as the license file.
port	Vendor daemon TCP/IP port number. The default, if *port* is not specified, is chosen by the operating system at run-time. Sites with Internet firewalls need to specify the TCP/IP port number the daemon uses. If a TCP/IP port number is specified on the VENDOR line, there may be a delay restarting the vendor daemon until all the clients have closed their connections to the vendor daemon.

SEE ALSO

- Chapter 5, “The Options File,” for further information regarding options file contents.

FLEXLM VERSION NOTES

- *vendor_daemon_path* required in pre-v6.0 vendor daemon.
- *options_file_path* required in pre-v6.0 vendor daemon.
- VENDOR lines are known as DAEMON lines in the pre-v6.0 `lmgrd` and vendor daemon.

  v6.0+:

  ```
  VENDOR sampled
  ```

  pre-v6.0:

  ```
  DAEMON sampled /etc/sampled \
         /etc/sampled/licenses/sampled.opt
  ```
- The `options=` keyword introduced in the v5.0 vendor daemon.

B.1.4 USE_SERVER Line

USE_SERVER takes no arguments and has no impact on the server. When the application sees USE_SERVER, it ignores everything in the license file except preceding SERVER lines and transfers checkout validation to the vendor daemon.

USE_SERVER is recommended since it improves performance when a license server is used. For uncounted features, USE_SERVER is used to force logging of usage by the daemons.

B.1.5 FEATURE/INCREMENT Lines

A FEATURE line describes the license required to use a product. An INCREMENT line can be used in place of a FEATURE line, as well as to incrementally add licenses to a prior FEATURE or INCREMENT line in the license file.

Only the first FEATURE line for a given feature is processed by the vendor daemon. If you want to have additional copies of the same feature (for example, to have multiple node-locked, counted features), then you must use multiple INCREMENT lines. INCREMENT lines form license groups, or *pools*, based on the following fields:

- feature name
- version

- DUP_GROUP
- FLOAT_OK
- HOST_BASED
- HOSTID
- PLATFORM
- USER_BASED
- VENDOR_STRING (if configured by the vendor as a pooling component)

If two lines differ by any of these fields, a new group of licenses, called a *license pool*, is created in the vendor daemon, and this group is counted independently from other license pools with the same feature name. A FEATURE line does not give an additional number of licenses, whereas an INCREMENT line always gives an additional number of licenses.

The basic FEATURE/INCREMENT line format is:

```
{FEATURE|INCREMENT} feature vendor feat_version exp_date \
            num_lic SIGN=sign [optional_attributes]
```

The six fields after the FEATURE/INCREMENT line keyword are required and have a fixed order. They are defined by the vendor and cannot be changed. Table B-1 presents these fields in the order they must appear.

Table B-1: FEATURE/INCREMENT Line Required Fields

Field	Description
feature	Name given to the feature by the vendor.
vendor	Name of the vendor daemon; also found in the VENDOR line. The specified daemon serves this feature.
feat_version	Version of this feature that is supported by this license.
exp_date	Expiration date of license in the format dd-mmm-yyyy, e.g., 07-may-2005. **Note:** If *exp_date* is the string “permanent” or the year is 0 (or 00, 000, 0000) then the license never expires.

Table B-1: FEATURE/INCREMENT Line Required Fields (Continued)

Field	Description
num_lic	Number of concurrent licenses for this feature. If the *num_lic* is set to the string "uncounted" or 0, the licenses for this feature are uncounted and no `lmgrd` is required but a hostid on the FEATURE line is required. See Section 4.4, "Counted vs. Uncounted Licenses."
`SIGN=`*sign*	SIGN= signature to authenticate this FEATURE line.

Table B-2 lists attributes that may appear in a FEATURE or INCREMENT line. They are supplied at the discretion of the vendor to provide particular licensing behavior. If present in the FEATURE or INCREMENT line, they must remain there and cannot be altered by the end user. These attributes have a *keyword=value* syntax where *keyword* is in uppercase.

In places where *value* is a string surrounded with double quotes ("..."), the string can contain any characters except a quote.

Table B-2: Vendor Supplied Attributes

Attribute	Description
`BORROW[=n]`	Enables license borrowing for a particular FEATURE/INCREMENT line. *n* is the number of hours that the license is borrowed. The default borrow period is 168 hours, or one week.

Table B-2: Vendor Supplied Attributes (Continued)

Attribute	Description

`DUP_GROUP=...`	The syntax is:
	`DUP_GROUP=NONE|SITE|[UHDV]`
	`U = DUP_USER`
	`H = DUP_HOST`
	`D = DUP_DISPLAY`
	`V = DUP_VENDOR_DEF`
	Any combination of UHDV is allowed, and the DUP_MASK is the OR of the combination. For example, DUP_GROUP=UHD means the duplicate grouping is (DUP_USER|DUP_HOST|DUP_DISPLAY), so for a user on the same host and display, additional uses of a feature do not consume additional licenses.

`FLOAT_OK` `[=`*`server_hostid`*`]`	Enables mobile licensing via FLEX*id* with FLOAT_OK for a particular FEATURE/INCREMENT line. This FEATURE/INCREMENT line must also be node-locked to a FLEX*id*. When `FLOAT_OK=`*`server_hostid`* is specified on a FEATURE line: • The *`server_hostid`* must refer to the same host that appears on the SERVER line of the license file. • The license server runs only on the machine with the hostid that `lmhostid` returns equal to the *`server_hostid`* specified with FLOAT_OK.
`HOSTID=` *`"hostid1`* *`[hostid2 ...`* *`hostidn]"`*	Id of the host to which the feature line is bound. *`hostid`* is determined with the `lmhostid` utility. This field is required for uncounted licenses; but can be used for counted licenses as well. See Appendix A, “Hostids for FLEXlm-Supported Machines,” for more information.

Table B-2: Vendor Supplied Attributes (Continued)

Attribute	Description
`HOST_BASED[=n]`	Host names must be specified in INCLUDE statements in the end-user options file, and the number of hosts is limited to *num_lic*, or the number specified in *=n*.
`ISSUED=` *dd-mmm-yyyy*	Date issued.
`ISSUER="..."`	Issuer of the license.
`LINGER=n`	The vendor-defined lingering interval for this license. Use LINGER in the options file to extend this time. See Section 5.2.11, “LINGER,” for more information.
`NOTICE="..."`	A field for intellectual property notices.
`OVERDRAFT=n`	The overdraft policy allows your vendor to specify a number of additional licenses which users are allowed to use, in addition to the licenses they have purchased. This allows your users to not be denied service when in a “temporary overdraft” state. Usage above the license limit is reported by the SAM*report* reporting tool.
`PLATFORMS="..."`	Usage is limited to the listed platforms.
`SN=`*serial_num*	Serial number, used to identify FEATURE or INCREMENT lines.
`START=` *dd-mmm-yyyy*	Start date.
`SUITE_DUP_GROUP=...`	Similar to DUP_GROUP, but affects only the enabling FEATURE line for a package suite. It limits the total number of users of the package to the number of licenses, and allows the package to be shared among the users that have the SUITE checked out.

Table B-2: Vendor Supplied Attributes (Continued)

Attribute	Description
`SUPERSEDE= "f1 f2 ..."`	If this appears, all licenses issued before the date specified in `ISSUED=` are *superseded* by this line and become ineffective.
`TS_OK`	FLEX*lm* detects when a node-locked uncounted license is running under Windows Terminal Server. To run on Terminal Server remote machines, `TS_OK` must be added to the FEATURE line. Without `TS_OK`, a user running on a Terminal Server client is denied a license.
`USER_BASED[=n]`	Users must be specified in INCLUDE statements in the end-user options file, and the number of users are limited to `num_lic`, or the number specified in `=n`.
`VENDOR_STRING= "..."`	Vendor-defined string, enclosed in double quotes.

The following attributes listed in Table B-3 are optional and are under control of the end user. These attributes have a `keyword=value` syntax where `keyword` is in lowercase.

Table B-3: End-User Attributes

Attribute	Description
`asset_info= "..."`	Additional information provided by the license administrator for asset management.
`dist_info= "..."`	Additional information provided by the software distributor.
`user_info= "..."`	Additional information provided by the license administrator.
`vendor_info= "..."`	Additional information provided by the software vendor.

Examples:

```
FEATURE sample_app sampled 2.300 31-dec-2005 20 \
        SIGN=123456789012
INCREMENT f1 sampled 1.000 permanent 5 \
        HOSTID=INTERNET=195.186.*.* NOTICE="Licensed to \
        Sample corp" SIGN=901234567890
```

FLEXLM VERSION NOTES

- Pre-v7.1 FEATURE/INCREMENT line format uses *license_key*:

  {FEATURE|INCREMENT} *feature vendor feat_version exp_date num_lic license_key* [optional_atributes]

  The pre-v7.1 format is understood by the current FLEXlm client library and vendor daemon.
- The SIGN= keyword introduced in the v7.1 FLEXlm client library and vendor daemon.
- For v7.1 through v8.0 FLEXlm client libraries and vendor daemons, the FEATURE/INCREMENT line must have a SIGN= signature and, for backward compatibility with pre-v8.1, can contain a *license_key*:

  {FEATURE|INCREMENT} *feature vendor feat_version exp_date* \
  num_lic [*license_key*] SIGN=*sign* \
  [optional_attributes]
- *license_key* obsoleted in v8.1 FLEXlm client library and vendor daemon
- The keyword “permanent” for *exp_date* introduced in v6 FLEX*lm* client library.
- The keyword “uncounted’ for *num_lic* introduced in v6 FLEX*lm* client library.
- BORROW keyword introduced in v8.0 FLEX*lm* client library and vendor daemon.
- FLOAT_OK keyword introduced in v8.0 FLEX*lm* client library and vendor daemon.
- TS_OK keyword introduced in v8.0 FLEX*lm* client library and vendor daemon.

B.1.6 PACKAGE Lines

The purpose of the PACKAGE line is to support two different licensing needs:

- To license a product SUITE, or
- To provide a more efficient way of distributing a license file that has a large number of features, which largely share the same FEATURE line arguments.

A PACKAGE line, by itself, does not license anything—it requires a matching FEATURE/INCREMENT line to license the whole package. A PACKAGE line is shipped by your software vendor with a product, independent of any licenses. Later, when you purchase a license for that package, one or more corresponding FEATURE/INCREMENT lines enable the PACKAGE line.

Example:

```
PACKAGE package vendor [pkg_version] COMPONENTS=pkg_list \
        [OPTIONS=SUITE] [SUPERSEDE[="p1 p2 ..."] ISSUED=date]
        SIGN=pkg_sign
```

Table B-4 lists the PACKAGE line fields. They must appear in the order listed.

Table B-4: PACKAGE Line Fields

Field	Description
package	Name of the package. The corresponding FEATURE/INCREMENT line must have the same name.
vendor	Name of the vendor daemon that supports this package.
pkg_version	Optional field specifying the package version. If specified, the enabling FEATURE/INCREMENT line must have the same version.

Table B-4: PACKAGE Line Fields (Continued)

Field	Description
`COMPONENTS=\` *`pkg_list`*	List of package components. The format is: *`feature`*`[:`*`version`*`[:`*`num_lic`*`]]` Packages must consist of at least one component. Version and count are optional, and if left out, their values come from the corresponding FEATURE/INCREMENT line. *`num_lic`* is only legal if `OPTIONS=SUITE` is not set—in this case the resulting number of licenses is *`num_lic`* on the COMPONENTS line multiplied by the number of licenses in the FEATURE/INCREMENT line. Examples: `COMPONENTS="comp1 comp2 comp3 comp4"` `COMPONENTS="comp1:1.5 comp2 comp3:2.0:4"`
`OPTIONS=SUITE`	Optional field. Used to denote a package suite. If set, the corresponding feature of the same name as the package is checked out in addition to the component feature being checked out. If not set, then the corresponding feature of the same name as the package is removed once the package is enabled; it is not checked out when a component feature is checked out.
`OPTIONS=` `SUITE_RESERVED`	Optional field. If set, reserves a set of package components. Once one package component is checked out, all the other components are reserved for that same user.
`SUPERSEDE` `[="`*`p1 p2 ...`*`"]`	Optional field. Used in conjunction with ISSUED date. Replaces all PACKAGE lines for the same package name with ISSUED dates previous to *`dd-mmm-yyyy`*.
`ISSUED=` *`dd-mmm-yyyy`*	Optional field. Used in conjunction with SUPERSEDE. Replaces all PACKAGE lines for the same package name with ISSUED dates previous to *`dd-mmm-yyyy`*.

Table B-4: PACKAGE Line Fields (Continued)

Field	Description
`SIGN=\` *`pkg_sign`*	License key or SIGN= signature.

Examples:

```
PACKAGE suite sampled 1.0 SIGN=3B24B2F508CB \
        COMPONENTS="comp1 comp2" OPTIONS=SUITE
FEATURE suite sampled 1.0 1-jan-0 5 SIGN=4193E6ABCCCB
```

This is a typical `OPTIONS=SUITE` example. There are two features, “comp1” and “comp2,” which are each version 1.0, each with five non-expiring licenses available. When “comp1” or “comp2” is checked out, “suite” is also checked out.

```
PACKAGE suite sampled 1.0 SIGN=2CBF44FCB9C1 \
        COMPONENTS="apple:1.5:2 orange:3.0:4"
FEATURE suite sampled 1.0 1-jan-2005 3 SIGN=321E78A17EC1 SN=123
```

In this example, the component version overrides the feature version, and the number of licenses available for any component is the product of the three licenses for “suite” and the number of licenses for that component. The result is equivalent to:

```
FEATURE c1 sampled 1.5 1-jan-2005 6 SIGN=0D3AD5F26BEC SN=123
FEATURE c2 sampled 3.0 1-jan-2005 12 SIGN=EB16C5AE61F0 SN=123
```

FLEXLM VERSION NOTES

- Ability to store PACKAGE lines in separate files introduced in v6 FLEX*lm* client library.
- *`pkg_version`* field required in pre-v7.1 FLEX*lm* client library.

B.1.7 UPGRADE Lines

```
UPGRADE feature vendor from_feat_version to_feat_version \
exp_date num_lic [options ... ] SIGN=sign
```

All the data is the same as for a FEATURE or INCREMENT line, with the addition of the *from_feat_version* field. An UPGRADE line removes up to the number of licenses specified from any old version (>= *from_feat_version*) and creates a new version with that same number of licenses.

For example, the two lines:

```
INCREMENT f1 sampled 1.000 1-jan-2005 5 SIGN=9BFAC0316462
UPGRADE f1 sampled 1.000 2.000 1-jan-2005 2 SIGN=1B9A308CC0F7
```

provide three v1.0 licenses of “f1” and two v2.0 licenses of “f1.”

An UPGRADE line operates on the closest preceding FEATURE or INCREMENT line with a version number that is >= *from_feat_version*, and < *to_feat_version*.

Note: UPGRADE lines do not work for node-locked, uncounted licenses.

B.2 Decimal Format

Licenses can be represented in decimal format. Decimal has the advantage that it’s simpler to type in, and often the licenses are much shorter.

A simple demo license in readable format:

```
FEATURE f1 sampled 1.00 1-jan-2005 0 key1 HOSTID=DEMO
```

and its decimal equivalent:

```
sampled-f1-00737-55296-1825
```

If needed, decimal lines can be mixed with readable format lines in a license file. Use the `lminstall` command to convert decimal licenses to readable format.

SEE ALSO

- Section 7.7, “lminstall,” for additional information on the `lminstall` command.

FLEXLM VERSION NOTES

- The decimal format was introduced in the v6 FLEX*lm* client library and vendor daemon.

B.3 License File Order

The order of the lines in a license file is not critical. They are sorted when they are processed so that in most cases the optimal result is achieved. However, pre-v7.0 versions of FLEX*lm* licensed applications and license servers implicitly impose an ordering to license file lines. Note the following suggestions for ordering lines in the license file:

- Place FEATURE lines before INCREMENT lines for the same feature.

  The rule regarding FEATURE lines is that only the first counted FEATURE line is observed by the license server, and that if there is a FEATURE line and INCREMENT lines, the FEATURE line must appear first.

- Where multiple counted FEATURE lines exist for the same feature, make sure the desired FEATURE line appears first.

  All but the first is ignored.

- Place node-locked, uncounted lines before floating lines for the same FEATURE.

  Otherwise, it is possible the floating license is consumed instead of the node-locked license, resulting in denial for other users.

- The placement of a USE_SERVER line affects behavior.

  A USE_SERVER line is recommended. Normally, the USE_SERVER line is placed immediately after the SERVER line. However, place any uncounted licenses not served by SERVER before the USE_SERVER line. Make sure each user that needs the uncounted license has direct access to a current copy of the file. The advantage to placing USE_SERVER right after the SERVER line is users don’t need up-to-date copies of the license file.

Appendix C

Troubleshooting Guide

This appendix documents areas of FLEX*lm* that have given customers difficulty in the past.

C.1 General Debugging Hints

The following are tips for debugging:

- When you start the license server (`lmgrd`) be sure that you direct the output into a local log file where you can examine it. The log file often contains useful information. Examine it when you have a problem, and be prepared to answer questions about it when you talk to a support person.
- If the license server appears to have started correctly (which you can determine from the log file), try running `lmstat -a` and `lmdiag` to see if that program has the same problem as your application.
- If your application is FLEX*lm* v4.1 or later (v5 or later on Windows), you can use the `FLEXLM_DIAGNOSTICS` environment variable. Set `FLEXLM_DIAGNOSTICS` to 1, 2, or 3. A setting of 3 gives more information than 2, 2 gives more information than 1 (in particular, the feature name that was denied). See Section C.2, "FLEXLM_DIAGNOSTICS," for more information.
- When you talk to a support person, be prepared with answers to the following questions:
  - What kind of machine is your license server running on?
  - What version of the operating system?
  - What machine and operating system is the application running on?
  - What version of FLEX*lm* does the FLEX*lm*-licensed application use?

    Use the `lmver` script, or, on UNIX, execute the following command on your `lmgrd`, vendor daemon, and application:

    `strings` *`binary_name`* `| grep Copy`

    Alternatively, `lmgrd -v` gives the `lmgrd` version, and this works with the vendor daemon also.

- What error or warning messages appear in the log file?
- Did the server start correctly?

  Look for a message such as:

  `server xyz started for: feature1 feature2.`
- What is the output from running `lmstat -a`?
- Are you running other products which are also licensed by FLEX*lm*?
- Are you using a combined license file or separate license files?
- Are you using a three-server redundant license server (multiple SERVER lines in your license file)?

C.2 FLEXLM_DIAGNOSTICS

Note: Available only with applications using FLEX*lm* v4.1 or higher for UNIX, and v5.0 or higher with Windows. Also, applications may choose not to provide this functionality.

`FLEXLM_DIAGNOSTICS` is an environment variable that causes the application to produce diagnostic information when a checkout is denied. The format of the diagnostic information may change over time.

To set `FLEXLM_DIAGNOSTICS`, on UNIX:

```
(csh): setenv FLEXLM_DIAGNOSTICS 1
(sh): FLEXLM_DIAGNOSTICS=1; export FLEXLM_DIAGNOSTICS
```

On Windows 95/98/ME, add the following line to `C:\autoexec.bat`:

```
SET FLEXLM_DIAGNOSTICS=1
```

On NT/2000/XP, use the System Control Panel applet to change the global environment, adding `FLEXLM_DIAGNOSTICS` to 1.

On UNIX, the diagnostic output goes to stderr.

On Windows, if the application is v5.11 or higher, the output is a file in the current directory called `flex`*`pid`*`.log`, where *`pid`* is the application's process ID. If the application is v5.0, the output file is called `flex_err.log`.

C.2.1 Level 1 Content

If `FLEXLM_DIAGNOSTICS` is set to 1, then the standard FLEX*lm* error message is be presented, plus a complete list of license files that the application tried to use. For example:

```
setenv FLEXLM_DIAGNOSTICS 1
FLEXlm checkout error: Cannot find license file (-1,73:2) No such file
or directory
license file(s):  /usr/myproduct/licenses/testing.lic license.lic
```

C.2.2 Level 2 Content

If `FLEXLM_DIAGNOSTICS` is set to 2, then, in addition to level 1 output, the checkout arguments are presented. For example:

```
setenv FLEXLM_DIAGNOSTICS 2
FLEXlm checkout error: No such feature exists (-5,116:2) No such file or
directory
license file(s):  /usr/myproduct/licenses/testing.lic license.lic
lm_checkout("f1", 1.0, 1, 0x0, ..., 0x4000)
```

Note that the error message actually contains two separate problems, which both occurred during the checkout:

- There's no such feature in the license it did find
- It was unable to find the other license file, which is what produces the message "No such file or directory"

Following is a description of the arguments to lm_checkout()

`lm_checkout(`*`feature, version, num_lic, queue_flag, ..., dupgroup_mask`*`)`

where:

`feature`	The requested feature.
`version`	The requested version. The license file must contain a version >= the requested version.
`num_lic`	Number of licenses requested. Usually 1.
`queue_flag`	If 0, no queueing If 1, queue for license ("blocking" queue) If 2, queue for licenses, but return to application ("non-blocking" queue)
`dupgroup_mask`	Indicates duplicate grouping, also called license sharing. User, host, and display are as shown by `lmstat -a`.

C.2.3 Level 3 Content (FLEX*lm* v6.0+ only)

If `FLEXLM_DIAGNOSTICS` is set to 3, then, in addition to level 1 and 2 output, if a checkout is successful, information is printed explaining how the license was granted:

```
setenv FLEXLM_DIAGNOSTICS 3
app
Checkout succeeded: f0/14263EAEA8E0
License file: ./servtest.lic
No server used
app2
Checkout succeeded: f1/BC64A7B120AE
License file: @localhost
License Server: @localhost
app3
Checkout succeeded: f1/BC64A7B120AE
License file: servtest.lic
License Server: @speedy
```

Note that the feature name and license key are printed, along with the license file location (or host name if `@host` were used) and host name of the server, where applicable.

C.3 FLEX*lm* Troubleshooting List

C.3.1 Problem Description Format

Each problem is presented in three parts:

Symptom:	*A description of the problem.*
Cause:	A discussion of what causes the problem described in the “Symptom” section.
Solution:	Instructions on how to solve the problem.

You can scan through the list of problems to find any which appear to relate to your concerns. In order to solve your problem, you may have to use all or some of the solutions listed here.

C.3.2 Hostid Problems

Symptom: *When I run the license manager on my machine, it tells me it is the wrong hostid.*

Cause: The vendor daemon checks the hostid listed on the SERVER line in the license file; if it does not match the hostid of the machine on which it is running, this message is printed. Possible causes include: 1) you are trying to run the license server on a different machine from the machine the file was made for; 2) the hostid of the machine you are running on changed (for example, the dongle hostid was moved (Windows) or the CPU board was replaced); 3) the hostid in the license file was modified.

Solution: Verify that the hostid of the machine on which the vendor daemon (or node-locked FLEX*lm*-licensed application) is being run matches the hostid specified in the license file (on the SERVER line for the vendor or on the FEATURE line for a node-locked FLEX*lm*-licensed application). You can run the `lmhostid` utility to see what FLEX*lm* thinks the hostid is. You may not modify the hostid in the license file. If the hostid of your server machine changes, you must get a new license file from your software vendor.

C.3.3 Connection Problems

Symptom: *The FLEXlm-licensed application (or `lmstat`) can't connect to the server to check out a license.*

Cause: The FLEX*lm* routines in the application are unable to make a TCP connection to the server and TCP/IP port specified in the license file. Possible reasons for this are: 1) the wrong license file is being referenced by the FLEX*lm*-licensed application; 2) the server machine specified in the license file is down; 3) the vendor daemon specified in the license file is not running; 4) the host name in the license file is not recognized by the system; 5) the network between the machine running the FLEX*lm*-licensed application and the server machine is down; 6) You are mixing FLEX*lm* v1.5 (or earlier) and v2.1 (or later) vendor daemons in one license file and have run `lmgrd` without the `-b` command line option; 7) TCP is not running on your machine.

Solution: The `lmdiag` utility is designed primarily to debug this problem, so first, try `lmdiag`. Verify that the application is using the proper license file. Verify that specified server machine is up and reachable by executing another command that uses TCP, such as `telnet`, from the FLEX*lm*-licensed application to the server. Verify that the vendor daemon is running (you can use `ps` on the server to look for it). Examine the license log file to see if any problems are reported, particularly messages indicating that the vendor daemon has quit. Run `lmstat -a` from the server machine to verify that the vendor daemon is alive. Run `lmstat -a` from the machine running the FLEX*lm*-licensed application to verify the connection from it to vendor daemon across the network. Try using `telnet` *`host port`* where *`host`* and *`port`* are the same as on the SERVER line in your license file.

C.3.4 Other FLEX*lm*-Licensed Application Problems

Symptom: *When I run my application* FLEX*lm-licensed application (or vendor daemon), I get the error* `bad code` *or* `inconsistent encryption code`*.*

Cause: Possible causes for this are: 1) the license file was modified (either the hostid on a SERVER line or anything on the FEATURE line was changed); 2) the vendor used the wrong version of his license creation utility to generate your license file (or there is a bug in that process).

Solution: You may not modify the license file (except for specific fields, see Chapter 2, “License File Basics”). If you need to change something in your license file, you must get a new license file from your software vendor.

Symptom: *When the second user tries to check out a license, the vendor daemon prints an error concerning* `Parameter mismatch` *in the debug log file and refuses the license. The application may give the error:* `Duplicate selection mismatch for this feature.`

Cause: The most likely cause of this problem is that you are simultaneously trying to run two different versions of the FLEX*lm*-licensed application, and the software vendor has not specifically set up the new version for this kind of compatibility. Check the debug log file for a comm version mismatch warning message; this indicates that someone is running an older FLEX*lm*-licensed application than the license server daemon, `lmgrd`.

Solution: Run only the new version of the application (or only the old version) or get a copy of a v8.0+ vendor daemon.

C.3.5 Other Server Problems

Symptom: *I'm having trouble restarting my license servers on Solaris.*

Cause: The default Solaris TCP/IP port delay of four minutes may cause problems when restarting a license server, especially a license server that is using a hard-coded TCP/IP port number (for example, each of the three-server redundant servers) rather than selecting from a set of default port numbers.

Solution: If you have a problem restarting license servers on Solaris, a root user can reset the TCP/IP port delay to 2.4 seconds (2400 milliseconds). As a long-term solution, it is recommended that one of the following lines be put in a boot script because the port delay is reset every time the machine reboots:

Solaris 2.6-:

```
/usr/sbin/ndd -set /dev/tcp \
                tcp_close_wait_interval 2400
```

Solaris 2.7+:

```
/usr/sbin/ndd -set /dev/tcp \
                tcp_time_wait_interval 2400
```

Symptom: *The license server keeps reporting "lost lock" errors in the log file and exiting.*

Cause: The lock file (normally placed in `/usr/tmp` on UNIX (`/var/tmp` in FLEX*lm* v8.1), `C:\flexlm` on Windows NT/2000/XP) is being removed by someone else. There could be another daemon running, or the license administrator (or a script he set up) could have deleted the file.

Solution: Check to see if there is more than one copy of the daemon running. On UNIX use a command like `ps -aux | grep vendor` to search for it. Check for more than one `lmgrd` running as well, since it restarts your vendor daemon when it is killed. If more than one `lmgrd` is running, kill them all (using the `kill` command, not `kill -9`, on UNIX or LMTOOLS on Windows NT/2000/XP), then kill any remaining vendor daemons (on UNIX, try a simple `kill`, if that fails then try `kill -9` the `lmgrd` and all vendor daemons) and start one fresh copy of `lmgrd`. On UNIX, check to see if there is a shell script running that cleans out `/tmp` (or `/usr/tmp` or `/var/tmp`). If so, try modifying it so that it does not delete zero-length files.

Appendix D

FLEX*lm* Environment Variables

Environment variables are not required in order to use FLEX*lm*-licensed applications. Environment variables are normally used for debugging or for changing license default location.

D.1 How to Set Environment Variables

FLEX*lm* environment variables are set in two different ways:

- The normal `set` or `setenv` commands (or the System Control Panel on Windows NT/2000/XP)
- The registry (Windows v6.0+) or in `$HOME/.flexlmrc` (UNIX v7.0+), which functions like the registry for FLEX*lm* on UNIX.

D.1.1 Registry

On Windows, the FLEX*lm* registry location is:

```
HKEY_LOCAL_MACHINE\Software\FLEXlm License Manager
```

On UNIX, the equivalent information is stored in `$HOME/.flexlmrc`. In this file, the syntax is *`variable=value`*.

D.1.2 Precedence

If the variable is `LM_LICENSE_FILE` or *`VENDOR`*`_LICENSE_FILE`, then both the environment and the registry are used, with the environment used first, and the registry appended to the path.

If it's a different variable, then if the environment set, only that is used, otherwise the registry is used. That is, the registry is only used if the environment is not set.

D.2 Environment Variables

Variable	Use (FLEX*lm* version introduced)
`FLEXLM_BATCH`	Windows only: prevents interactive pop-ups from appearing. Set to 1 if a batch application. (Version 7.0+ clients)
`FLEXLM_DIAGNOSTICS`	Used for debugging where applications don't print FLEX*lm* error message text. Set to 1, 2, or 3, depending on the amount of diagnostic information desired. See Section C.2, "FLEXLM_DIAGNOSTICS." (Version 5.0+ clients)
`FLEXLM_TIMEOUT`	Windows only: Sets the timeout value a FLEX*lm*-licensed application uses when attempting to connect to a license server port in the range 27000-27009. Values are in microseconds, within the range 0 through 2147483647. The default setting is 100000 microseconds.
`LM_LICENSE_FILE` or *`VENDOR`*`_LICENSE_FILE`	Reset path to license file. Can be a license-file list, separated by " : " on UNIX and " ; " on Windows. If *`VENDOR`*`_LICENSE_FILE` used, *`VENDOR`* is the vendor daemon name used by this application. For example, Macrovision products use `GSI_LICENSE_FILE`. Can be a file name, or *`port@host`*. See also Section 2.1.1, "Setting the Path with an Environment Variable." (*`VENDOR`*`_LICENSE_FILE` requires v6.0+ clients.)
`LM_BORROW`	Used for initiating license borrowing and setting the borrow period. See Section 8.4.1, "Initiating License Borrowing," for more details.

`LM_PROJECT`	`LM_PROJECT`'s value is logged in the report log file and later reported on by SAM*report*. Limited to 30 characters. (v5.0+ client required.) This can also be used to RESERVE, INCLUDE, etc. licenses with PROJECT. For example: `RESERVE 1 f1 PROJECT airplane` v5.0+ clients and v7.0+ vendor daemon are required for this feature.

Appendix E

Frequently Asked Questions

E.1 License File Questions

E.1.1 I've received FLEX*lm* license files from two different companies. Do I have to combine them?

You don't have to combine license files. Each license file that has any counted lines (the license count field is > 0) requires a server. It's perfectly OK to have any number of separate license files, with different `lmgrd` server processes supporting each file. Moreover, since `lmgrd` is a lightweight process, for sites without system administrators, this is often the simplest (and therefore recommended) way to proceed. With v6.0+ `lmgrd`/`lmdown`/`lmreread`, you can restart/stop/reread a single vendor daemon (of any FLEX*lm* version). This makes combining licenses more attractive than previously. Also, if the application is v6.0+, using `dir/*.lic` for license file management behaves like combining licenses without physically combining them.

E.1.2 When is it recommended to combine license files?

Many system administrators, especially for larger sites, prefer to combine license files to ease administration of FLEX*lm* licenses. It's purely a matter of preference.

E.2 FLEX*lm* Versions

E.2.1 I have products from several companies at various FLEX*lm* version levels. Do I have to worry about how these versions work together?

If you're not combining license files from different vendors, the simplest thing to do is make sure you use the tools that are shipped by each vendor.

`lmgrd` always supports older versions of vendor daemons and applications, so it's *always* safe to use the latest version of `lmgrd` and the `lmutil`/LMTOOLS utilities. If you've combined license files from two vendors, you *must* use the latest version of `lmgrd`.

If you've received two versions of a product from the same vendor, you *must* use the latest vendor daemon they send you. An older vendor daemon with a newer FLEX*lm*-licensed application causes communication errors.

Please ignore letters appended to FLEX*lm* versions, for example, the "d" in v2.4d. The appended letter indicates a patch, and does NOT indicate any compatibility differences. In particular, some elements of FLEX*lm* didn't require certain patches, so a v2.4 `lmgrd` works successfully with a v2.4b vendor daemon. See also Section I.1, "Version Compatibility and Components."

E.2.2 I've received a new copy of a product from a vendor, and it uses a new version of FLEX*lm*. Is my old license file still valid?

Yes. Older FLEX*lm* license files are always valid with newer versions of FLEX*lm*.

E.2.3 I've received a new license file, and the format is different from the old one I had from the same vendor? Why? Are they compatible?

As of v3.0, FLEX*lm* has an optional new format for license files. FLEX*lm* products always understand older versions; therefore, the pre-v3.0 files are understood by every FLEX*lm* version. However, your old applications (pre-FLEX*lm* v3.0) are not be able to use the new license file. See also Appendix B, "License File Format."

E.3 Using FLEX*lm*

E.3.1 Does FLEX*lm* work across the Internet?

Yes. A server on the Internet serves licenses to anyone else on the Internet with access to those ports on that machine. This can be limited with the INTERNET= attribute on the FEATURE line, which limits access to a range of Internet addresses. You can also use the INCLUDE and EXCLUDE options in the daemon options file to allow (or deny) access to clients running on a range of Internet addresses.

E.3.2 Does FLEX*lm* work with Internet firewalls?

Many firewalls require that TCP/IP port numbers be specified to the firewall. FLEX*lm* v5 `lmgrd` and vendor daemons supports this.

E.3.3 If my FLEX*lm*-licensed application dies, does the server free the license?

Yes, unless the FLEX*lm*-licensed application's whole system crashes, or becomes disconnected from the network. Assuming communications is TCP, the license is automatically freed immediately. If communications are UDP, then the license is freed after the UDP timeout, which is set by each vendor, but defaults to 45 minutes. UDP communications is normally only set by the end user, so TCP is assumed. If the whole system crashes, then the license is not freed; use `lmremove` to free the license.

E.3.4 What happens when the license server dies?

FLEX*lm* applications send periodic heartbeats to the server to discover if it has died. What happens when the server dies is then up to the application:

- Continue periodic attempts to re-checkout the license when the server comes back up.
- Attempt to re-checkout a license a few times, and then, presumably with some warning, exit.
- In the case of GUI applications, present pop-ups to the user periodically letting them know the server is down and needs to be restarted.

E.3.5 How do you tell if a TCP/IP port is already in use?

99.44% of the time, if it's in use, it's because `lmgrd` is already running on the TCP/IP port—or was recently killed, and the port isn't freed yet. Assuming this is not the case, then use `telnet` *`host port`*— if it says "Can't connect," it's a free port.

E.3.6 Does FLEX*lm* require root permissions?

No. There is no part of FLEX*lm*, `lmgrd`, vendor daemon, or application, that requires root permissions. In fact, it is recommended that you do not run the license server (`lmgrd`) as "root," since root processes can introduce security risks. If `lmgrd` must be started from the root user (for example, in a system boot script), it is recommended that you use the `su` command to run `lmgrd` as a non-privileged user:

```
su username -c "lmgrd_path -c license_file_path -l debug_log_path"
```

where *`username`* is a non-privileged user, and the paths are the correct paths to `lmgrd`, the license file, and the debug log file. You must ensure that the vendor daemons listed in the license file have execute permissions for *`username`*. The paths to all the vendor daemons in the license file are listed on each VENDOR line.

E.3.7 Is it OK to run lmgrd as “root” (UNIX only)?

It is not prudent to run any command, particularly a daemon, as root on UNIX, as it may pose a security risk to the operating system. Therefore, it is recommended that `lmgrd` be run as a non-privileged user (not “root”). For example, if you are starting `lmgrd` from a boot script, use:

```
su username -c "umask 022; lmgrd..."
```

to run `lmgrd` as a non-privileged user.

E.3.8 Does FLEX*lm* licensing impose a heavy load on the network?

No, but partly this depends on the application, and end user’s use. A typical checkout request requires five messages and responses between the FLEX*lm*-licensed application and server, and each message is <150 bytes.

When a server is not receiving requests, it requires virtually no CPU time.

When an application, or `lmstat`, requests the list of current users, this can significantly increase the amount of networking FLEX*lm* uses, depending on the number of current users.

Also, prior to FLEX*lm* v5, use of *`port@host`* can increase network load, since the license file is downloaded from the server to the system running the FLEX*lm*-licensed application. For pre-v5 FLEX*lm*, it is recommended that *`port@host`* be limited to small license files (say, <50 features). In v5+, *`port@host`* actually *improves* performance.

E.3.9 Does FLEX*lm* work with NFS?

Yes. FLEX*lm* has no direct interaction with NFS. FLEX*lm* uses an NFS-mounted file like any other application.

E.3.10 Does FLEX*lm* work with ATM, ISDN, Token-Ring, etc.

In general, these have no impact on FLEX*lm*. FLEX*lm* requires TCP/IP. So long as TCP/IP works, FLEX*lm* works.

E.3.11 Does FLEX*lm* work with subnets, fully qualified names, multiple domains, etc.?

Yes, although this behavior was improved in v3.0, and v6.0.

When a license server and a FLEX*lm*-licensed application are located in different domains, fully qualified host names have to be used. A “fully qualified host name” is of the form *`machine.domain`*, where *`machine`* is the local host name (usually returned by the `hostname` command or `uname -n`) and *`domain`* is the Internet domain name, e.g., “globes.com.”

To ensure success with FLEX*lm* across domains, do the following:

1. Make the sure the fully qualified host name is the name on the SERVER line of the license file.
2. Make sure ALL FLEX*lm*-licensed application machines, as well as the server machine, are able to `telnet` to that fully qualified host name. For example, if the host is locally called “speedy,” and the domain name is “corp.com,” local systems are able to logon to “speedy” via `telnet speedy`. But very often the fully qualified host name, `telnet speedy.corp.com,` fails locally. Note that this `telnet` command always succeeds on hosts in other domains (assuming everything is configured correctly), since the network resolves “speedy.corp.com” automatically.
3. Finally, there must be an alias for “speedy” so it’s also known locally as “speedy.corp.com.” This alias is added to the `/etc/hosts` file, or if NIS/Yellow Pages are being used, then it has to be added to the NIS database. This requirement goes away in version 3.0 of FLEX*lm*.

If all components (application, `lmgrd` and vendor daemon) are v6.0 or higher, no aliases are required; the only requirement is that the fully qualified domain name, or IP address, is used as a host name on the SERVER, or as a host name in the `LM_LICENSE_FILE` environment variable (*`port@host`* or *`@host`*).

E.3.12 Does FLEX*lm* work with NIS and DNS?

Yes. However, some sites have broken NIS or DNS, which causes FLEX*lm* to fail. In v5 of FLEX*lm*, NIS and DNS can be avoided to solve this problem. In particular, sometimes DNS is configured for a server that’s not currently available (e.g., a dial-up connection from a Windows system). Again, if DNS is configured, but the server is not available, FLEX*lm* fails.

On Windows, if a checkout seems to take three minutes and then fails, this is usually because the system is configured for a dial-up DNS server which is not currently available. The solution here is to turn off DNS.

Finally, host names must not have periods in the name. These are not legal host names, and they do not work with DNS; although, Windows systems allow you to define them.

E.3.13 We are using FLEX*lm* over a wide-area network. What can we do to improve performance?

With the most common uses of FLEX*lm*, traffic is negligible. In particular, checkout, checkin, and heartbeats use very little networking traffic. There are two items, however, which can send considerably more data:

- `lmstat -a`

  Use sparingly. Network traffic increases as the amount of concurrent users and features increases. As a rule of thumb, avoid using this command when there are more than 20 concurrent users or features.
- Prior to FLEX*lm* v5, the *`port@host`* mode of the `LM_LICENSE_FILE` environment variable should be avoided, especially when the license file has many features, or there are a lot of license files included in `LM_LICENSE_FILE`. The license file information is sent via the network, and can place a heavy load. Failures due to *`port@host`* generate the error `LM_SERVNOREADLIC (-61)`.

E.4 Windows Questions

E.4.1 What Windows Platforms are supported?

FLEX*lm* v9.0 supports Windows 95/98/98SE, NT, NT Server, 2000, ME, and XP.

E.4.2 Will the Server run on Windows 95?

Yes.

Appendix F

FLEX*lm* Error Codes

F.1 Error Message Format

FLEX*lm* error messages presented by applications have the following components:

- FLEX*lm* Error Number—a negative number starting at -1.
- FLEX*lm* Error Text—short sentence (< 80 characters) summarizing problem.
- FLEX*lm* Error Explanation (optional)—short paragraph (3-5 lines) explaining problem and possible solutions or workarounds.
- FLEX*lm* Minor Error Number—a positive number starting at 1. These numbers are unique error identifiers and are used by FLEX*lm* vendors for more advanced support assistance. Their meaning is not documented.
- System Error Number (optional)—a UNIX or Windows OS error code last set by the operating system.
- System Error Explanation (optional)—a short sentence (< 80 characters) explaining the system error.
- Other supporting information (optional)

Error messages were improved in v6. FLEX*lm* Error Explanation, and supporting information are only available in applications using v6.0+.

These error messages may occur in two formats available with FLEX*lm* or may appear in a format customized by the application.

F.1.1 Format 1 (short):

```
FLEXlm error text (-lm_errno, minor_num[:sys_errno]) [sys_error_text]
```

The system error information may be missing.

Example:

```
Can’t connect to license server (-15,12:61) Connection refused
```

F.1.2 Format 2 (long—version 6.0+):

```
FLEXlm error text
 FLEXlm error explanation
[Optional Supporting information]
FLEXlm error: -lm_errno, minor_num. [System Error: sys_errno]
["system_error_text"]
```

Example:

```
Cannot connect to license server
 The server (lmgrd) has not been started yet, or
 the wrong port@host or license file is being used, or the
 port or hostname in the license file has been changed.
Feature:       f1
Server name:   localhost
License path: @localhost:license.dat:./*.lic
FLEXlm error:  -15,12.  System Error: 61 "Connection refused"
```

F.2 Error Code Descriptions

The following table lists the most common errors produced by FLEX*lm*-licensed products.

Table 8-1: FLEXlm Error Codes

Error Code	Description
-1	Cannot find license file.
-2	Invalid license file syntax.
-3	No server for this feature.
-4	Licensed number of users already reached.
-5	No such feature exists.
-6	No TCP/IP port number in license file and FLEXlm service does not exist. (pre-v6 only)
-7	No socket connection to license manager service.
-8	Invalid (inconsistent) license key or signature. The license key/signature and data for the feature do not match. This usually happens when a license file has been altered.

Table 8-1: FLEXlm Error Codes (Continued)

Error Code	Description
-9	Invalid host. The hostid of this system does not match the hostid specified in the license file.
-10	Feature has expired.
-11	Invalid date format in license file.
-12	Invalid returned data from license server.
-13	No SERVER lines in license file.
-14	Cannot find SERVER host name in network database. The lookup for the host name on the SERVER line in the license file failed. This often happens when NIS or DNS or the hosts file is incorrect. Workaround: Use IP address (e.g., 123.456.789.123) instead of host name.
-15	Cannot connect to license server. The server (`lmgrd`) has not been started yet, or the wrong *`port@host`* or license file is being used, or the TCP/IP port or host name in the license file has been changed.
-16	Cannot read data from license server.
-17	Cannot write data to license server.
-18	License server does not support this feature.
-19	Error in select system call.
-21	License file does not support this version.
-22	Feature checkin failure detected at license server.
-23	License server temporarily busy (new server connecting).
-24	Users are queued for this feature.
-25	License server does not support this version of this feature.

Table 8-1: FLEXlm Error Codes (Continued)

Error Code	Description
-26	Request for more licenses than this feature supports.
-29	Cannot find ethernet device.
-30	Cannot read license file.
-31	Feature start date is in the future.
-32	No such attribute.
-33	Bad encryption handshake with daemon.
-34	Clock difference too large between client and server.
-35	In the queue for this feature.
-36	Feature database corrupted in daemon.
-37	Duplicate selection mismatch for this feature. Obsolete with v8.0+ vendor daemon.
-38	User/host on EXCLUDE list for feature.
-39	User/host not on INCLUDE list for feature.
-40	Cannot locate dynamic memory.
-41	Feature was never checked out.
-42	Invalid parameter.
-47	Clock setting check not available in daemon.
-52	FLEXlm vendor daemon did not respond within timeout interval.
-53	Checkout request rejected by vendor-defined checkout filter.
-54	No FEATURESET line in license file.
-55	Incorrect FEATURESET line in license file.
-56	Cannot compute FEATURESET data from license file.

Table 8-1: FLEXlm Error Codes (Continued)

Error Code	Description
-57[1]	socket() call failed.
-59	Message checksum failure.
-60	Server message checksum failure.
-61	Cannot read license file data from server.
-62	Network software (TCP/IP) not available.
-63	You are not a license administrator.
-64	lmremove request before the minimum lmremove interval.
-67	No licenses to borrow.
-68	License BORROW support not enabled.
-69	FLOAT_OK can't run standalone on SERVER.
-71	Invalid TZ environment variable.
-73	Local checkout filter rejected request.
-74	Attempt to read beyond end of license file path.
-75[1]	SYS$SETIMR call failed (VMS).
-76	Internal FLEXlm error—please report to Macrovision.
-77	Bad version number must be floating-point number with no letters.
-82	Invalid PACKAGE line in license file.
-83	FLEXlm version of client newer than server.
-84	USER_BASED license has no specified users - see server log.
-85	License server doesn't support this request.
-87	Checkout exceeds MAX specified in options file.

Table 8-1: FLEXlm Error Codes (Continued)

Error Code	Description
-88	System clock has been set back.
-89	This platform not authorized by license.
-90	Future license file format or misspelling in license file. The file was issued for a later version of FLEX*lm* than this program understands.
-91	ENCRYPTION_SEEDS are non-unique.
-92	Feature removed during lmreread, or wrong SERVER line hostid.
-93	This feature is available in a different license pool. This is a warning condition. The server has pooled one or more INCREMENT lines into a single pool, and the request was made on an INCREMENT line that has been pooled.
-94	Attempt to generate license with incompatible attributes.
-95	Network connect to `this_host` failed. Change `this_host` on the SERVER line in the license file to the actual host name.
-96	Server machine is down or not responding. See the system administrator about starting the server, or make sure that you're referring to the right host (see `LM_LICENSE_FILE` environment variable).
-97	The desired vendor daemon is down. 1) Check the `lmgrd` log file, or 2) Try `lmreread`.
-98	This FEATURE line can't be converted to decimal format.
-99	The decimal format license is typed incorrectly.
-100	Cannot remove a linger license.

Table 8-1: FLEXlm Error Codes (Continued)

Error Code	Description
-101	All licenses are reserved for others. The system administrator has reserved all the licenses for others. Reservations are made in the options file. The server must be restarted for options file changes to take effect.
-102	A FLEX*id* borrow error occurred.
-103	Terminal Server remote client not allowed.
-104	Cannot borrow that long.
-106	License server out of network connections. The vendor daemon can't handle any more users. See the debug log for further information.
-110	Dongle not attached, or can’t read dongle. Either the hardware dongle is unattached, or the necessary software driver for this dongle type is not installed.
-112	Missing dongle driver. In order to read the dongle hostid, the correct driver must be installed. These drivers are available at www.macrovision.com or from your software vendor.
-113	Two FLEXlock checkouts attempted. Only one checkout is allowed with FLEX*lock*-enabled applications.
-114	SIGN= keyword required, but missing from license. This is probably because the license is older than the application. You need to obtain a SIGN= version of this license from your vendor.
-115	Error in Public Key package.
-116	CRO not supported for this platform.
-117	BORROW failed.
-118	BORROW period has expired.

Table 8-1: FLEXlm Error Codes (Continued)

Error Code	Description
-119	lmdown and lmreread must be run on license server machine.
-120	Cannot lmdown the server when licenses are borrowed.
-121	FLOAT_OK license must have exactly one dongle hostid.
-122	Unable to delete local borrow info.
-123	Support for returning a borrowed license early is not enabled. The vendor must have enabled support for this feature in the vendor daemon. Contact the vendor for further details.
-124	An error occurred while returning a borrowed license to the server.
-125	Attempt to checkout just a PACKAGE. Need to also checkout a feature.
-126	Error initializing a composite hostid.
-127	A hostid needed for the composite hostid is missing or invalid.
-128	Error, borrowed license doesn't match any known server license.

1.Indicates errors due to an operating system failure.

Appendix G

The Report Log File

The license server produces both report log files and debug log files. The focus of this appendix is report log files. For information on debug log files see Appendix H, “The Debug Log File.”

The report log file contains feature usage information and is generated by the vendor daemon. However, a vendor daemon does not write report log output by default. Report log output is not human readable and is only used by the SAM*report* and FLEX*bill* products. Therefore, unless you are using either of these two Macrovision products (or intend to use them in the future), there is no reason to enable report logging.

G.1 Managing Report Log Output

As a vendor daemon runs for a period of time, the volume of report log output increases. If you have a lot of license activity, these log files grow very large. You need to consider where to put these files and how often to rotate and archive them. Therefore, it may be necessary to rotate or switch report log output into different files over time, each file containing license activity over a particular period of time.

Report log data is collected by the vendor daemon into an internal data buffer area before being flushed to the output file. The daemon’s internal buffer is flushed once a minute or whenever it gets full, whichever occurs first. To ensure the freshest data possible in the report log file, flush the buffer on demand with the `lmreread` command. Use standard file compression tools to reduce the size of a report log file when it is no longer being written.

To avoid corruption and for performance, it is suggested that the vendor daemon write its report log to a file on a disk local to the system running the vendor daemon. Each vendor daemon must write to its own report log file.

G.2 Enabling Report Log Output for a Vendor Daemon

There are two ways to enable report logging for a particular vendor daemon either before or after starting the license server.

- Add the REPORTLOG line to the options file for that vendor daemon. See Section 5.2.15, “REPORTLOG,” for more details.
- Invoke `lmswitchr` on the vendor daemon. See Section 7.14, “lmswitchr,” for more details.
- Invoke `lmnewlog` on the vendor daemon. Requires a v7.1+ vendor daemon. See Section 7.8, “lmnewlog,” for more details.

G.3 Redirecting Report Log Output for a Vendor Daemon

The report log output for a particular vendor daemon can be moved into separate files, each file representing activity over a different period of time. There are three ways in which to do this whether the vendor daemon is running or not:

- Change the REPORTLOG line in the vendor daemon’s options file and reread its options file by invoking `lmreread` (v8.0+ vendor daemon) or restart.
- Invoke `lmswitchr` on the vendor daemon. See Section 7.14, “lmswitchr,” for more details.
- Invoke `lmnewlog` on the vendor daemon. Requires a v7.1+ vendor daemon. See Section 7.8, “lmnewlog,” for more details.

Appendix H

The Debug Log File

The license server produces both debug log files and report log files. The focus of this appendix is debug log files. For information on report log files see Appendix G, “The Report Log File.”

A debug log file contains status and error messages useful for debugging the license server. A license server always generates debug log output. Some of the debug log output describes events specific to `lmgrd` and some of the debug log output describes events specific to each vendor daemon.

H.1 Managing Debug Log Output

As `lmgrd` and its vendor daemons run for a period of time, the volume of this output increases. As it gets older, the value of the debug log output decreases; therefore, it may be necessary for you to separate old debug log output from current output; either archive or delete the old output.

For performance, it is suggested that each debug log file be a on disk that is local to the machine that is running `lmgrd` and its vendor daemons. However, if the debug log file must be on a remotely-mounted disk and you find that the license server is too slow, start `lmgrd` with the `-nfs_log` option to improve performance.

See Section H.2, “Debug Log Messages,” for a description of the debug log output format.

H.1.1 Capturing Debug Log Output for a License Server

By default, `lmgrd` and the vendor daemons it manages write debug log output to standard out. To put this debug log output in a file, either redirect the output of the license server to a file or start `lmgrd` with the `-l` *`debug_log_path`* option.

H.1.2 Capturing Debug Log Output for a Particular Vendor Daemon

The debug log output from different vendor daemons controlled by the same license server can be written to their own files (v8.0+ vendor daemon). There are two ways to do this:

- Add the DEBUGLOG line to the options file for each vendor daemon. See Section 5.2.2, “DEBUGLOG,” for more details.
- Invoke `lmswitch` on the vendor daemon. See Section 7.13, “lmswitch,” for more details.

Note that `lmgrd` writes its own debug log output to standard out.

H.1.3 Redirecting Debug Log Output for a Running Vendor Daemon

It is possible to redirect the debug log output for a particular vendor daemon to a different file. There are two ways to do this:

- Change the DEBUGLOG line to the options file for the vendor daemon and reread its options file by invoking `lmreread`. See Section 5.2.2, “DEBUGLOG,” for more details.
- Invoke `lmswitch` on the vendor daemon. See Section 7.13, “lmswitch,” for more details.

H.1.4 Limiting Debug Log Output for a Vendor Daemon

By default, debug log output contains all events. To limit the events that are logged for a particular vendor daemon, add a NOLOG line to the options file of that vendor daemon. See Section 5.2.14, “NOLOG,” for more details. One of the reasons you may want to limit the events that are logged is to reduce the size of the debug log output.

H.2 Debug Log Messages

FLEX*lm* daemons generate debug log files in the following format:

```
hh:mm:ss (daemon) message
```

where:

hh:*mm*:*ss*	Time that the message was logged.
daemon	Either `lmgrd` or the vendor daemon name. In the case where a single copy of the daemon cannot handle all of the requested licenses, an optional “_” followed by a number indicates that this message comes from a forked daemon.

message The text of the message.

The debug log files can be used to:

- Diagnose configuration problems
- Diagnose daemon software errors

Note: A debug log file cannot be used for usage reporting with SAM*report*.

H.2.1 Informational Messages

Message	Description
Connected to *host*	This daemon is connected to its peer on *host*.
CONNECTED, master is *host*	The license daemons log this message when a quorum is up and everyone has selected a master.
DENIED: *num_lic feature* to *user*	*user* was denied access to *num_lic* licenses of *feature*.
EXITING DUE TO SIGNAL *nnn* EXITING with code *nnn*	All daemons list the reason that the daemon has exited.
EXPIRED: *feature*	*feature* has passed its expiration date.
IN: "*feature*" *user* (*num_lic* licenses)	*user* has checked in *num_lic* licenses of *feature*.

Message	Description
Lost connection to *host*	A daemon can no longer communicate with its peer on node *host*, which can cause the clients to have to reconnect, or cause the number of daemons to go below the minimum number, in which case clients may start exiting. If the license daemons lose the connection to the master, they kill all the vendor daemons; vendor daemons shut themselves down.
Lost quorum	The daemon lost quorum, so it processes only connection requests from other daemons.
MULTIPLE *vendor* servers running. Please kill, and restart license daemon.	The license manager daemon, `lmgrd`, has detected that multiple vendor daemons for *vendor* are running. Shutdown `lmgrd` and all *vendor* daemons with the `lmdown` utility and then restart `lmgrd`.
OUT: "*feature*" *user* (*num_lic* licenses)	*user* has checked out *num_lic* licenses of *feature*.
RESERVE *feature* for USER *user* RESERVE *feature* for HOST *host*	A license of *feature* is reserved for either *user* or *host*.
REStarted *vendor* (internet port *nnn*)	Vendor daemon *vendor* was restarted at TCP/IP port *nnn*.
Retrying socket bind (address in use)	The license servers try to bind their sockets for approximately six minutes if they detect "address in use" errors.
Selected (EXISTING) master *host*.	This license daemon has selected an existing master *host* as the master.

Message	Description
SERVER shutdown requested.	A daemon was requested to shut down via a user-generated `kill` command.
Server started on *`host`* for: "*`feature_list`*"	A (possibly new) server was started for the features listed.
Shutting down *`vendor`*	The license manager daemon is shutting down the vendor daemon *`vendor`*.
SIGCHLD received. Killing child servers.	A vendor daemon logs this message when a shutdown was requested by the license daemon.
Started *`vendor`*	The license manager daemon logs this message whenever it starts a new vendor daemon.
Trying to connect to *`host`*	The daemon is attempting a connection to *`host`*.

H.2.2 Configuration Problem Messages

Message	Description
`host`: Not a valid server host, exiting	This daemon was run on an invalid host name.
`host`: Wrong hostid, exiting	The hostid is wrong for *`host`*.
BAD CODE for *`feature`*	The specified feature name has a bad license key or signature. It was probably typed in wrong, or modified by the end user.
CANNOT OPEN options file *`file`*	The options file specified in the license file could not be opened.
Couldn't find a master	The daemons could not agree on a master.

Message	Description
License daemon: lost all connections	This message is logged when all the connections to a server are lost, which often indicates a network problem.
Lost lock, exiting Error closing lock file Unable to re-open lock file	The vendor daemon has a problem with its lock file, usually because of an attempt to run more than one copy of the daemon on a single node. Locate the other daemon that is running via a `ps` command, and kill it with `kill -9`.
No DAEMON line for *`vendor`*	The license file does not contain a DAEMON or VENDOR line for *`vendor`*.
No DAEMON lines, exiting	The license daemon logs this message if there are no DAEMON or VENDOR lines in the license file. Because there are no vendor daemons to start, there is nothing for the license daemon to do.
No features to serve!	A vendor daemon found no features to serve. This could be caused by a corrupted or incorrectly entered license file.
UNSUPPORTED FEATURE request: *`feature`* by *`user`*	The user has requested a feature that this vendor daemon does not support. This can happen for a number of reasons: the license file is bad, the feature has expired, or the daemon is accessing the wrong license file.
Unknown host: *`host`*	The host name specified on a SERVER line in the license file does not exist in the network database (probably `/etc/hosts`).

H.2.3 Daemon Software Error Messages

Message	Description
accept: message	An error was detected in the “accept” system call.
Can’t allocate server table space	A malloc error. Check swap space.
Connection to *host* TIMED OUT	The daemon could not connect to *host*.
Illegal connection request to *vendor*	A connection request was made to *vendor*, but this vendor daemon is not *vendor*.
read: error message	An error in a “read” system call was detected.
select: message	An error in a “select” system call was detected. This is usually a sign of a system networking failure.
Server exiting	The server is exiting. This is normally due to an error.

Appendix I

FLEX*lm* Versions

I.1 Version Compatibility and Components

In general, always use the latest `lmgrd` and `lmutil`/LMTOOLS, which are available from www.macrovision.com, and you'll automatically enjoy many of the enhancements available in the most recent versions of FLEX*lm*. However, some enhancements require a vendor daemon built with a newer version of FLEX*lm*, and yet others require a FLEX*lm*-licensed application built with a newer version of FLEX*lm*. Contact your software vendor for the most current version of their vendor daemon.

The rules about FLEX*lm* version compatibility can be summarized as:

Version of `lmutil`/LMTOOLS must be >=

Version of `lmgrd`, which must be >=

Version of vendor daemon, which must be >=

Version of FLEX*lm*-licensed application, which must be >=

Version of license file format

Except for the license file, you can use `lmver` to discover the version of all these components. For the vendor daemon, `lmgrd`, and `lmutil`, you can also use the `-v` argument to print the version.

I.2 How to Tell the License File Version

The following rules apply to individual FEATURE, INCREMENT or UPGRADE lines. It's possible to have a mix of versions in a single file. Only the features that a particular application checks out determine the version of the license for that feature.

Version 2	Blank quotes or a quoted string at the end of the FEATURE line.
>= Version 3	INCREMENT or UPGRADE line.

>= Version 4	OVERDRAFT, DUP_GROUP, INTERNET, or PACKAGE appear.
>= Version 5	SUPERSEDE, ISSUED, USER_BASED, HOST_BASED, or SN appear.
>= Version 6	START appears.
>= Version 7.1	SIGN= keyword appears.
>= Version 8	BORROW, FLOAT_OK, and TS_OK appear.

I.3 Version Summary

v1.0—1988

First FLEX*lm* Release, containing all the basic FLEX*lm* features

v1.5—February 1990

First widely used version including DEMO

v2.1—March 1991

- Improved TIMEOUT support
- Improved ethernet hostid support

v2.21—November 1991

- Added support for many platforms and some platform-specific improvements, such as hostid
- Hostid ANY added

v2.26—March 1992 (Used only by Sun)

- Added license lingering

v2.4—December 1992

- Added “use-all-feature-lines” capability for incremental license distribution
- Enhanced vendor customization routines
- Enhanced end-user options file
- Added new hostid types: USER, HOSTNAME, and DISPLAY
- Added `port@host` to locate license file —downloads license file from server

v2.61—March 1993 (Used only by Sun)

- Added INCREMENT and UPGRADE lines to license file

V3.0—MAY 1994

- INCREMENT and UPGRADE behavior changed and improved
- Added UDP protocol support
- Added `uname -i` hostid for HP
- Added multiple jobs for enhanced support of `LM_LICENSE_FILE` environment variable as a license-file list
- New, optional license file format with *`keyword=value`* syntax for optional new features, including: asset_info, ISSUER, and NOTICE, “ \ ” license file continuation character, 2048 character limit per feature

V4.0—DECEMBER 1994

- Removed use of floating point, for enhanced reliability
- FEATURE line additions: ck, OVERDRAFT, DUP_GROUP, INTERNET hostid
- PACKAGE line
- License Finder
- `lmdiag` and `FLEXLM_DIAGNOSTICS` for end-user diagnostics

V4.1—MAY 1995

- Performance improvements and new platform support

V4.1—PATCH RELEASE 6, OCTOBER 1995

- Windows patch release for Windows 95 with various performance improvements

V5.0—MARCH 1996

- Improved *`port@host`* behavior—FLEX*lm*-licensed application doesn’t read license file
- Automatic *`port@host`* via USE_SERVER line in license file
- Hostid lists—lock a feature to several hostids
- New FEATURE attributes: SN (serial number), USER_BASED, HOST_BASED, MINIMUM, SUPERSEDE, ISSUED (issued date), CAPACITY (charging based on system capacity)
- Optional avoidance of NIS and DNS via IP address instead of host name
- Improved report log file format
- Server, upon startup, notifies of licenses that expire within two weeks
- Improved end-user options file functionality

V5.11—FEBRUARY 1997

- SUPERSEDE lists, PLATFORMS= license attribute,

- new end-user options: MAX, TIMEOUTALL
- Windows control panel added
- Windows license generator `GENLIC` added

V5.12—APRIL 1997

- Performance improvements and new platform support

V6.0—SEPTEMBER 1997

- `lmgrd` can read multiple license files
- FLEX*lm* license directory support: `*.lic` automatically used
- License files require no editing for use at the end-user site
- Optional path on DAEMON/VENDOR line; `$PATH` environment variable used
- Decimal license format, with `lminstall` utility for typing in licenses
- FEATURE lines are shorter, easier to understand and type in
- PACKAGE lines can be shipped in separate files that never require user editing
- Default TCP/IP port numbers make SERVER line port number optional
- Default end-user options file path
- `this_host` host name supported on SERVER line
- *`VENDOR`*`_LICENSE_FILE` supported (e.g., `GSI_LICENSE_FILE`)
- `@`*`host`* supported where default port numbers are used
- Windows only: user prompted for license file or license server name
- License files are optionally case insensitive
- `lmdown` and `lmreread` accept `-vendor` *`vendor`* argument
- `START=`*`dd-mmm-yyyy`* optional license attribute

V6.1—JUNE 1998

- Performance improvements

V7.0—AUGUST 1999

- License Certificate Manager support for automatic license fulfillment
- Support for “try-before-you-buy” licensing
- License file handles inserted newlines from emailers
- License lines automatically optimally sorted
- Improved LMTOOLS interface for Windows

- `lmgrd`, when run at command line on Windows, runs in background by default
- Improved three-server redundancy reliability (v7.0 vendor daemon and `lmgrd`)
- `lmreread` and `lmdown` take `-all` argument to shut down or reread all `lmgrd`s
- Support registry (Windows) and `$HOME/.flexlmrc` (UNIX) for FLEX*lm* environment variables
- Automatically install license path in registry or `$HOME/.flexlmrc` after successful checkout
- Options support for `LM_PROJECT` with PROJECT
- Performance improvements, especially for Windows NT
- Intel Pentium III CPU-ID (v7.0d+, November 1999)

v7.1—August 2000

- Security enhancements
- SIGN= keyword in license
- `lmnewlog` utility (v7.1+ vendor daemon)

v7.2—December 2000

- Performance enhancements

v8.0—October 2001

- `lmborrow` (v8.0+ components), `lmpath` (v8.0+ vendor daemon), `lmswitch` (v8.0+ vendor daemon) utilities
- `lmreread` rereads end-user options file and SERVER host name
- License borrowing with BORROW keyword

v8.1—January 2002

- CRO Security enhancements

v8.2—August 2002

- Support added for Windows XP compliancy

v8.3—October 2002

- Support added for returning borrowed licenses early

v8.4—January 2003

- Support for reserved package suites

v9.0—March 2003

- Support for COMPOSITE= hostid type

Index

A
about this manual ix
ANY hostid 109
application, what happens when dies 145
asset_info 122

B
BORROW_LOWWATER 52
borrowing 102

C
combining license files 143
commands x
COMPOSITE
    hostid 109
concurrent license 29
configuring FLEXlm files 21
conventions x
converting license formats 87
creating options file 47

D
DAEMON line 115
debug log file
    format 160
debugging license server 129
DEBUGLOG 53
decimal format licenses 87
DEMO hostid 109
deployed FLEXlm files
    FLEXid drivers 20
    FLEXlock DLL 20
    lmgr8b.dll 20
    lmgrd 20
    lmtools.exe 20
    lmutil 20
    vendor daemon 20
diagnosing checkout problems
    troubleshooting
        checkouts 83
disabling
    lmdown 68
    lmremove 68
DISPLAY
    hostid 110
    type 51
dist_info 122
DUP_GROUP 120

E
enabling report log 60
environment variables
    FLEXLM_BATCH 140
    FLEXLM_DIAGNOSTICS 140
    FLEXLM_TIMEOUT 140
    LM_BORROW 140
    LM_LICENSE_FILE 140
    LM_PROJECT 141
    setting 139
    VENDOR_LICENSE_FILE 140
error code
    descriptions 150

format 149
EXCLUDE 53
EXCLUDE_BORROW 54
EXCLUDEALL 55
expiration date 118

F
feature
version 118
FEATURE line 117
asset_info 122
dist_info 122
DUP_GROUP 120
expiration date 118
feature version 118
FLOAT_OK 120
HOST_BASED 121
HOSTID 120
ISSUED 121
ISSUER 121
license count 119
NOTICE 121
OVERDRAFT 121
PLATFORMS 121
serial number 121
SIGN 119
signature 119
SN 121
START 121
SUPERSEDE 122
syntax 123
TS_OK 122
USER_BASED 122
user_info 122
vendor daemon name 118
vendor_info 122
VENDOR_STRING 122
Feature line
SUITE_DUP_GROUP 121
FLEXid with FLOAT_OK 100
flexid.exe 20
flexidsilent.exe 20
FLEXlm
across Internet 144
and ATM 146
and DNS 147
and domains 146
and firewalls 144
and ISDN 146
and NFS 146
and NIS 147
and subnets 146
and WAN performance 147
components 15
configuration 21
getting started checklist 22
installing client applications 22
network load 146
version compatibility 143
FLEXlm Programmers Guide x
FLEXlm Reference Manual x
FLEXLM_BATCH 140
FLEXLM_DIAGNOSTICS 130
level 1 130
level 2 131
level 3 132
FLEXLM_TIMEOUT 140
FLOAT_OK 120
floating license 29

G
GROUP type 55

H
HOST type 51
host, SERVER line 114
HOST_BASED 121
HOST_GROUP type 56
HOSTID 120
hostid

ANY 109
COMPOSITE 109
DEMO 109
DISPLAY 110
HOSTNAME 110
ID 110
INTERNET 110
SERVER line 114
special 109
table by platform 108
USER 110
HOSTNAME hostid 110

I

ID hostid 110
INCLUDE 56
INCLUDE_BORROW 57
INCLUDEALL 58
INCREMENT line 117
installing client applications 22
INTERNET
hostid 110
type 52
ISSUED 121
ISSUER 121

L

license
borrowing 102
concurrent 29
floating 29
mixed 30
mobile 99
network license 29
node-locked 29
license count 119
license directory 70, 71
license file
combining multiple 143
compatibility between different versions 39
DAEMON line 115
decimal format 127
expected location 18
FEATURE line 117
format 28
frequently asked questions 143
how to combine 38
INCREMENT line 117
LM_LICENSE_FILE 18
lminstall 87
order of lines 30, 128
overview 18
PACKAGE line 124
rereading after an update 91
SERVER lines 39
specifying location 25
types 29
UPGRADE line 127
USE_SERVER line 117
VENDOR line 115
with multiple servers 69
license manager daemon 17, 67
license pool 50, 118
license rehosting 99
license request process 21
license server
and Windows 95 148
debugging 129
deciding number of nodes 43
disk space used 42
install as Windows service 98
sockets used 41
what happens when dies 145
license-file list 35
license-file list redundancy 44
LINGER 58
LM_BORROW 140
LM_LICENSE_FILE 140
to reference multiple files 18

LM_PROJECT 141
reporting on project 61
use in options file 52
lmdiag
syntax 83
troubleshooting 83
lmdown
disabling 68
restricting access 68
syntax 84
lmgrd
and redundant servers 69
combining license files 143
compatibility between versions 40, 67
debug log file 160
memory usage 42
overview 17, 67
ports 145
root perms on UNIX 145
running as root 146
shutting down 84
starting 67, 69
starting automatically at boot time 23
starting debug log 68
syntax 67
use latest 167
lmhostid
syntax 85
lmhostid, syntax 85
lminstall
license file format 87
syntax 87
lmnewlog, syntax 88
lmremove
disabling 68
restricting access 68
syntax 90
lmreread
restricting access 68
syntax 91
lmstat
output for lmreread 93
syntax 93
lmswitch, syntax 95
lmswitchr, syntax 96
LMTOOLS 20, 97
lmutil
lmdiag 83
lmdown 84
lmhostid 85
lminstall 87
lmnewlog 88
lmremove 90
lmreread 91
lmstat 93
lmswitch 95
lmswitchr 96
lmver 97
lmver, syntax 97

M

MAX 59
MAX_OVERDRAFT 60
memory usage, daemons 42
mixed licenses 30
mobile licensing
borrowing 102
FLEXid with FLOAT_OK 100
node-locked to FLEXid 99
node-locked to laptop 99
node-locked to user name 106
prepaid license pool fulfillment 106

N

network bandwidth and FLEXlm 43
network license 29
node-locked license 29
NOLOG 60

NOTICE 121

O

office xi
options file
    BORROW_LOWWATER 52
    creating 47
    DEBUGLOG 53
    DISPLAY type 51
    examples 64
    EXCLUDE 53
    EXCLUDE_BORROW 54
    EXCLUDEALL 55
    GROUP type 55
    HOST type 51
    HOST_GROUP type 56
    INCLUDE 56
    INCLUDE_BORROW 57
    INCLUDEALL 58
    INTERNET type 52
    LINGER 58
    MAX 59
    MAX_OVERDRAFT 60
    NOLOG 60
    overview 23
    PROJECT type 52
    read by vendor daemon 63
    REPORTLOG 60
    required for HOST_BASED 121
    required for USER_BASED 122
    RESERVE 61
    rules of precedence 63
    TIMEOUT 62
    TIMEOUTALL 63
    type argument 51
    USER type 51
options file path 116
OPTIONS=SUITE 125
OPTIONS=SUITE_RESERVED 125
order of lines in license file 30, 128
OVERDRAFT 121

P

PACKAGE line 124
    OPTIONS=SUITE 125
    OPTIONS=SUITE_RESERVED 125
    syntax 124
package suite 125
PLATFORMS 121
port number
    server default range 115
    SERVER line 115
    VENDOR line 116
ports in use 145
preface ix
PROJECT type 52
prospects xi

R

redundant servers
    selecting server nodes 43
    separate license files 69
    SERVER lines 114
    TCP port delay 136
    three-server redundancy 45
    via license-file list 44
rehosting, license 99
remote disks, guidelines for using 43
report log file 42
reporting on project 61
REPORTLOG 60
RESERVE 61
restarting redundant servers 136
restricting access
    lmdown 68
    lmremove 68
    lmreread 68

S
SAMreport 61
SAMsuite Users Guide x
SAMwrap Users Guide x
SERVER line 114
    combining license files 39
    default port numbers 115
    host 114
    hostid 114
    port number 115
    redundant servers 114
    syntax 114
setting environment variables 139
setting TCP port delay 136
SIGN 119
signature 119
SN 121
sockets
    differences under SCO 41
    number used by license server 41
specifying location of license file 25
START 121
starting lmgrd 69
status of license server 93
SUITE_DUP_GROUP 121
SUPERSEDE 122
switching debug log
    lmswitch 95
switching report log
    lmnewlog 88
    lmswitchr 96

T
TCP port delay, setting 136
technical support x
term x
three-server redundancy 45
TIMEOUT 62
TIMEOUTALL 63
troubleshooting
    bad code error 135
    can't connect to server 134
    lost lock error 136
    parameter mismatch 135
    restarting redundant servers 136
    with FLEXLM_DIAGNOSTICS 130
    with lmdiag 83
    wrong hostid 133
TS_OK 122
typographic conventions x

U
UPGRADE line, syntax 127
USE_SERVER line 117
USER hostid 110
USER type 51
USER_BASED 122
user_info 122
user_info= 121

V
vendor daemon
    and redundant servers 69
    debug log file 160
    lmnewlog 88
    lmreread 91
    lmswitchr 96
    memory usage 42
    options file 48
    overview 17
    restarting 85
    sockets 41
    uncounted licenses 46
    VENDOR_LICENSE_FILE 140
    version compatibility 40, 67
vendor daemon name
    FEATURE line 118
    VENDOR line 115
vendor daemon path 116

VENDOR line 115
    options file path 116
    port number 116
    vendor daemon name 115
    vendor daemon path 116
vendor.opt 48, 116
vendor_info 122
VENDOR_LICENSE_FILE 27, 140
VENDOR_STRING 122

W

Windows 95, and license server 148