License Admin Manual

Release 16.1.0

Oct 15, 2024

Welcome to the documentation for Reprise Software!

For questions, comments, or issues with the documentation, please contact support@reprisesoftware.com.

Chapter 1

Administration Basics

The License Administration manual is for license administrators and end-users, and describes the setup and management of an RLM floating license server.

1.1 Introduction

1.1.1 About this Manual

This manual describes setup and maintenance of the RLM licensing system, and as such it is intended for license administrators and users at organizations which have purchased software that uses the Reprise License Manager. This manual explains how to install and configure the licensing system included with your software.

1.1.2 Introduction To RLM

You are most likely reading this manual because one or more of your software vendors (ISVs) have included RLM in their product(s) to enforce their license agreements. This manual describes the components of RLM that you need to understand in order to accomplish day-to-day license administration tasks.

RLM allows your organization to know that you are using purchased software within the license limits set by your ISV. In addition, RLM collects usage information (at your option) for later reporting and analysis. This usage information is provided in a fully documented report log format, described in appendix-a of this manual.

When one of your ISVs delivers software to you that incorporates RLM, in addition to the normal application files, you will receive some additional RLM components:

- The RLM (generic) license server provided by Reprise Software, called “rlm” on Unix systems, or “rlm.exe” on Windows. This is the same for every ISV who uses RLM.
- The RLM utilities (“rlmutil” on Unix, “rlmutil.exe” on Windows) provided by Reprise Software. This is the same for every ISV who uses RLM.
- A custom license server, built from components from Reprise Software by your ISV. This server will have a different name for each ISV.
- A license file to describe your rights to the product. This license file is unique to your site.

In addition to these components which your ISV supplies, you can create an ISV options file to control various aspects of operation of each licensed product. This options file is described later in this manual. In addition, an RLM options file allows you to restrict access to various administration commands.

RLM is a client-server system, with license requests transmitted over TCP/IP from the software application to a license server that controls license usage rights.

1.1.3 What sets RLM apart?

RLM was designed from the start to emphasize **openness**, **transparency**, and **simplicity**.

RLM is **open** because we publish the format of our report log file, so that you can always examine and generate usage reports on licensing activity from the RLM servers.

RLM is **transparent** in the sense that we do not allow "back doors" which lead to unique behaviors from one ISV to another. In addition, we have removed policy from the application code, and placed it into the license key itself, so that everyone will be able to understand the license terms without having to understand a particular implementation by an ISV.

RLM is **simple** because we include functionality like truly automatic selection of license servers from a set of multiple, independent servers. In older license management systems, the ISV ends up writing much code to manage multiple license servers. This is handled by RLM itself.

1.1.4 Software License Management Basics

RLM is similar in structure to most popular license managers. RLM consists of 3 major components:

1. A client library
2. A license server (RLM has 2 license servers - a generic server called *rlm* and an ISV-specific server.)
3. A text file which describes the licenses granted (the license file).

Your ISVs application is linked with the client library which provides access to the license management functions.

The license server is used for floating licenses and logging of usage data. You, as a License Administrator, have the ability to control certain aspects of the license server's operations by specifying options in a file called the ISV Options File.

The RLM client library (linked into your ISVs application) and the license server are both controlled by license authorizations stored in a text file called the *license file*.

Most license managers provide APIs with calls to control many of the aspects of licensing behavior, as well as options within the license servers to control licensing behavior. The design philosophy of RLM is to preserve the simplicity of the system for both ISVs and License Administrators by avoiding all unnecessary options in the client library and the license servers and moving all these options to the license file, where they are visible and understandable by everyone. In general, license policy should be kept out of the application and the license server, and placed into the license itself. This makes for a more understandable licensing system for both ISVs and License Administrators. The API is simpler, and the license server performs in a more standard way from ISV to ISV. This prevents license management confusion by customers. We learned this the hard way when we supported hundreds of customers in the past and applied these lessons to the design of RLM.

1.2 Installing an RLM-licensed Product

When you receive a product licensed with RLM, your Software Provider (or Independent Software Vendor, referred to in this manual as your "ISV") will provide an installation procedure that installs the license management components (in the case of floating licenses, this is typically separate from the installation procedure for the application, since the license server processes usually run on a different machine from the application). Your ISV will generally make the licensing installation as transparent as possible.

In some cases, your ISV will not provide you with an installation procedure for the license server (the license server is required for floating licenses only – it is not required for nodelocked

licenses). This might happen, for example, if you want to run the server on a platform which your ISV does not support. The remainder of this section describes a manual RLM server installation, in the event you need to install it yourself or troubleshoot your installation.

First, you need the three required licensing components for the license server:

- The generic license server, *rlm* on Unix, *rlm.exe* on Windows.
- The ISV's license server, either a settings file named isv.set, or a binary named isv on Unix, *isv.exe* on Windows.
- The license file which describes your rights to the product.

Optionally, you might want the RLM utilities - *rlmutil* on Unix, *rlmutil.exe* on Windows. These utilities are often installed as their separate command names, see Section 1.5.

For the easiest installation, place all three components in the same directory (put the utilities, if you want to install them, in that directory as well.) In this way, license servers, utilities, and application programs will all be able to locate the license without any additional environment settings for users. All that remains to get floating licensing working is to start the license servers.

Note If you did not locate the license file (or a link to it) in the binary directory, you need to provide a pointer to the license file (or license server) to the application, using the RLM_LICENSE environment variable.

To start the license server (again, only required for floating, or counted licenses):

1. Place the license file into the binary directory (or startup directory) and name it *something.lic* If you do not do this, then set the RLM_LICENSE environment variable to the path of the license file.
2. Execute the rlm command:

```
% rlm > output_file
```

To enable your users to find the license file or license server, either:

1. Put the license file (named something.lic) in the binary directory with the application program (RECOMMENDED), OR
2. setenv RLM_LICENSE license_file_path, OR
3. setenv RLM_LICENSE port@host, where port is the port # in the license file, and host is the hostname in the license file.

Note If you are using RLM v10.0 or later (both clients and servers), and you are running on a local-area network, the client will broadcast to locate the license server, and no other configuration is required beyond setting up the license server.

1.2.1 How a floating licensing works

To use floating licenses, an RLM client connects to an RLM server to check out a floating license. (The RLM client is usually part of the licensed application.)

RLM Server

- The RLM server is started by a start script on the server machine, or started manually.
- The RLM server scans for valid license files, for one or more ISVs. For each ISV, rlm will start the ISV server – either an ISV server binary, or another copy of rlm that takes on

the personality of the ISV server via the ISV server settings file.

- The RLM server opens and listens on one or more ports to receive license requests from the RLM clients.
- The RLM server port numbers are specified on the HOST line of the license files. The default (if none is given) is 5053.
- The ISV server's port number is specified on the ISV line of the license file. If there is more than one license file, there can be more than one port defined for the ISV server. If no port number is set, a random free port number is used.
- Once a day - at midnight local time - the RLM server will trigger a re-read of all ISV license files. All ISV servers will then reread the license file and continue processing requests.

Warning In RLM v14.2—15.2, if the default admin password is not changed, or no password is set, the RLM web server will automatically shutdown after 10 minutes. For information on access control, see Section 1.7.2.

RLM Client

- The *application* contains an embedded "RLM client". Eventually some part of the application will request a license "check out". To process this request, the RLM client connects to the RLM server.
- The RLM server gets the requests from the RLM client, looks up the ISV (in its internal list of ISV servers), and forwards the RLM client to the correct ISV server port number.
- The RLM client then connects to the ISV server and sends a "check out" request for the requested license.

Note The RLM client needs to 'know' only the RLM server host name and port number. The *application's installation procedure* must document how to set these values.

Troubleshooting in large networks

In order to check out a license, the RLM client (embedded in the application) needs to be able to connect to the RLM server and to the ISV server. Large companies with internal firewalls must make sure both ports are accessible for the RLM client to successfully check out a license.

1.3 The License Server

The license server consists of at least two processes:

- The generic server, called *RLM*
- At least one ISV server, named *isv* (where *isv* is the name of the software publisher)

The RLM server is provided by Reprise Software and is completely generic. The ISV server is configured to contain license key validation that is ISV-specific.

The RLM server handles requests from clients and directs them to the appropriate ISV server. In addition, the RLM server watches for failures in ISV servers and restarts them when appropriate. The RLM server also provides status to the various utilities, when appropriate.

The RLM server initiates a reread of the license files (for itself and any ISV servers) at midnight every night.

The RLM server is delivered with an embedded Web Server to perform normal administration tasks. For more information on the web server interface, see Section 1.7.

1.3.1 RLM startup options

The RLM command is:

```
% rlm [-c <license_file>] [-dat] [-dlog [+]<logfile>] [-info]
        [-l] [-noudp] [-nows | -ws port] [-x [rlmdown|rlmremove]]
        [-install_service] [-service_name <service_name>]
        [-v] [-user <username> -password <password>]
        [-isv_startup_delay <seconds>]
        [-verify] [-sslcert <certfile> -sslpriv <privkey>]
```

The **-c license_file** option specifies which license file to use. This option overrides the setting of the RLM_LICENSE environment variable. The license_file parameter can be a directory containing license files, all of which will be processed.

The **-dat** option specifies that license files should have the extension ".dat", rather than ".lic". If -dat is specified, the RLM server will search for all files ending in ".dat" instead of ".lic" as documented elsewhere.

The -dlog logfile specifies the pathname for the server debug log. If logfile is preceded by the

'+' character, the logfile will be appended, otherwise it is overwritten. All ISV servers will write their output to the same logfile specified in the -dlog option.

The **-info** option causes RLM to print information about all copies of RLM that are running on this computer, including copies which have run in the prior 24 hours, then exit.

The **-install_service** and **-service_name sname** options are used to run the RLM server as a service under Windows. Optionally a **username** and **password** of a user account under which you wish to run the service may be specified, via the **-user** and **-password** arguments. See the description of running the RLM server as a service below.

Notes on using the -user and -password options:

1. Windows expects the username argument to be <domain><user>. To use the local system domain, specify ".<username>", e.g., ".joe". Without the "." you will get a service creation failure.
2. In order to run a service, the account specified by the -user argument must have the "Log on as a Service" property set. Details on how to set that property on an account can be found here: https://learn.microsoft.com/en-us/windows/security/threat-protection/security-policy-settings/log-on-as-a-service

---

The **-isv_startup_delay seconds** option specifies that when running as a Windows service, RLM should delay *seconds* seconds before starting up the ISV servers. If not specified, there is no delay. This is useful if a license file specifies a hostid of type rlmid1 (a hardware key), the server is started at system boot time, and the key driver is not yet started at the time the ISV server needs to read it.

The **-l** switch causes rlm to only process command-line utilities from the local host (v12.1+).

The **-nows** and **-ws port** options control the operation of the embedded Web Server. The *-nows* option instructs the rlm server to not start the embedded web server. The *-ws port* option instructs the RLM server to use *port* as the port number for the web server.

The **-noudp** option tells RLM to not bind the UDP port (5053) used for replying to broadcast messages from clients in RLM v10.0 and later.

The **-v** option causes RLM to print its version and exit.

The **-verify** option will cause RLM to start all ISV servers to verify their licenses, then all servers

will exit.

The **-x [rlmdown | rlmremove]** option controls whether the *rlmdown* and/or *rlmremove* commands will be processed by the server. Specifying only *-x* will disable both commands. Specifying either command name after the *-x* will disable just that command.

The **-sslcert certfile** and **-sslpriv privkey** options together instruct the RLM web server to use HTTPS using the SSL certificate certfile and SSL private key privkey. These options first appeared in RLM v15.1

Note These options can appear in any order on the command line.

If you want to generate a report log file, specify this on an ISV-by-ISV basis in the individual ISV's options file. See the description of the Section 1.9.23 in The ISV Options File for more information.

Warning If the RLM server cannot bind the web server port (5054 by default), it will exit.

Also note that if there is not at least one license file with the current hostname (as returned by gethostname()), or "localhost", a warning is generated.

If the ISV server pathname is incorrect in a license file which RLM is processing, RLM will attempt to start that ISV server using the path information in other license files, if present.

Starting in RLM v15.1, processing of -iai and -z flags has been removed. RLM will silently ignore these flags without giving an error message.

1.3.2 The Server Debug Log

Both RLM and the ISV servers write debug logs, useful for diagnosing licensing inconsistencies or failures. By default, this output goes to stdout, which is usually the window where you started RLM. You can change the location of the debug log with the -dlog_logfile command line argument to RLM, described above. You can also change the location of the ISV server debug log with the DEBUGLOG line in the ISV options file. See Section 1.9 for more information.

When starting RLM as a service on windows, the debug logs will be written to the same directory which contains the rlm.exe binary, so long as this directory is writable. If it is not writable, the log files will be written to \\Windows\system32. (However, note that RLM will not allow itself to be installed as a service when an unwritable debug log is specified).

1.3.3 Running the RLM server as a service on Windows

On Microsoft Windows servers, you may want to install and run the RLM server as a Windows service process. A service process can start automatically at boot time and remain running as long as the system is up, regardless of user logins and logouts.

You can install RLM as a service in a command window. Once installed as a service, it remains installed until it is explicitly deleted as a service. Installing RLM as a service does not start RLM; services are started via the Windows Services control panel, and at boot time.

You can only install RLM as a service in a command window. To do this, use the RLM program itself (in a command window), with special arguments:

```
rlm -install_service -dlog [+]logfile [-service_name sname] [-user username]
[-password password] <rlm runtime args>
```

Where:

- **logfile** is the pathname for the server debug log. This parameter is required. If preceded

by the '+' character, the logfile will be appended, rather than created.

- **sname** is an optional name for the installed service. If not specified, sname defaults to "RLM". If sname contains embedded whitespace, it must be enclosed in double quotes.
- **<rlm runtime args>** are any other command line arguments to be passed to RLM when it is started.

Examples:

```
rlm -install_service -dlog c:\logs\server.log
```

The most basic installation. This will install the service under the default service name "RLM" with the logfile server.log under C:\logs\.

```
rlm -install_service -service_name rlm-xyz -dlog c:\logs\server.log -c
c:\licenses\xyz.lic
```

This installs RLM as a service under the name "rlm-xyz". When started via the Services control panel or at boot time, RLM will be passed the "-c c:\licenses\xyz.lic" args, and it will write it's debuglog information to the file c:\logs\server.log

Installed RLM services are also deleted with the RLM program. Services must be stopped via the service control panel before they can be deleted. Note that deleting a service deletes it from the Windows service database; it does not delete the RLM executable or associated license file(s):

```
rlm -delete_service [-service_name sname]
```

Where:

- **sname** is an optional name for the installed service. If not specified, service_name defaults to "RLM". If service_name contains embedded whitespace, it must be enclosed in double quotes.

Notes:

- It is desirable to use the **-c <license file>** command line argument with RLM when installed as a service. Use of environment variables with Windows services is undesirable, as the environment passed to started services is the one in effect at boot time.
- On systems which run RLM license servers, it is a good idea to install each ISV's instance of RLM with a service_name argument which reflects the ISV or ISVs whose licenses are being served by that instance of RLM. For example, if a system ran two instances of RLM as services, where the first instance served license for ISVs "Blue" and "Green", and the second instance served license for ISV "Yellow", they might be installed as "RLM Blue-Green" and "RLM Yellow", respectively.
- Because the Service Controller on Windows invokes services under a special user account in a special default directory, it is necessary to use full paths for:
    - the -c <license file> argument on the RLM command line
    - in ISV daemon paths in the license file
    - in options file paths in the license file
    - in debug log paths in the ISV options file
    - in report log paths in the ISV options file
    - for the -dlog debug_log argument on the command line

NOTE on the use of DEBUGLOG

When running the server as a Windows Service, if no DEBUGLOG is specified in the ISV options file, RLM will write the ISV debug log in: <location of rlm.exe>\<isv>.dlog

This file will be overwritten every time the ISV server starts, since there is no opportunity to specify that the file should be appended to in the default case. In fact, the ISV server logs a few lines to this file at startup time even if a DEBUGLOG is specified in the ISV options file. It is overwritten every time the ISV server starts, but its contents don't change from startup to startup, so nothing important is lost.

Reprise Software Inc. recommends that the debug log path be specified in the ISV options file, and that the append behavior be enabled with '+'<path>. However, it is important not to specify the debug log name as <isv>.dlog, as this specific file is overwritten at each startup.

- When running as service RLM changes its working directory to the directory where rlm.exe is installed, so log files will be written there if log file paths are not specified as absolute paths. rlm.exe checks to make sure that it can write to that directory before changing its working directory. If it can't be written, RLM leaves its working directory as c:windowssystem32.
- When you install RLM as a service, it starts and then stops the installed service. This is so that if there are any firewall issues - ports blocked that RLM needs to use - the warnings come at service installation time rather than when RLM is started for the first time.
- Starting in v10.0, RLM checks the path you specify for the debug log (-dlog <log> or in the "Server Debug Log box in the web interface). If this file cannot be written, an error is printed and the service install fails.

1.3.4 Starting the RLM server at boot using systemd on Linux

1. cd to /etc/systemd/system
2. Create file rlm.service:

```
[Unit]
After=network.target
Description=Reprise License Management Server

[Service]
# Optional environment variables
Environment=RLM_LICENSE=/path/to/license/files
Type=simple
User=<user>
WorkingDirectory=<Directory where RLM is located>
ExecStart=<Path to RLM [arg1 arg2 … argN]>
Restart=always

[Install]
WantedBy=multi-user.target
```

3. Start the service and enable at boot:

```
systemctl enable --now rlm.service
```

1.3.5 Starting the RLM server at system boot using initd on Linux

On most Unix systems, system services are started at boot time, usually via startup scripts located in /etc/rc.<something>. For example, on Solaris, the startup script might be placed in /etc/rc2.d/S98rlm. On Linux systems, the script could be located in /etc/init.d/rlm, with a link to /etc/rc5.d/S98rlm. You must install this startup script as root.

> **Warning** The startup script should su to a different user so that the RLM servers are not running as root.

The following is an example of a script which would start RLM at boot time on Unix systems. Modify the first 5 variables for the target system.

```
#! /bin/sh
#
# rlm   Start/Stop rlm
#
#-------------------------------------------------------------------
#-------------------------------------------------------------------
#-------------------------------------------------------------------
# NOTE:
# NOTE: Configure these 5 variables for your system
# NOTE:

# Set rlmuser to the user under which RLM will run
rlmuser=bobm

# Set rlmdir to the directory where the RLM binary is found
rlmdir=/home/bobm/rlm/dev/rlm

# Set rlmdir to the directory where the rlmdown binary is found
rlmdowndir=$rlmdir

# Set licfile to the path to the license file
licfile=$rlmdir/x.lic

# Set debuglog to the path to the debug log
debuglog=+$rlmdir/rlm.dl
#-------------------------------------------------------------------
#-------------------------------------------------------------------
#-------------------------------------------------------------------

start() {
echo $debuglog
        su - $rlmuser -c "$rlmdir/rlm -c $licfile -dlog $debuglog &"
}

stop() {
        su - $rlmuser -c "$rlmdowndir/rlmdown RLM -c $licfile -q"
}

case "$1" in
  start)
        start
        ;;
  stop)
        stop
        ;;
  restart)
        stop
        sleep 2
```

```
        start
        ;;
  *)
echo $"Usage: $0 {start|stop|restart}"
        exit 1
esac

exit 0
```

1.3.6 License Server Startup Processing

License servers use The License Environment to find their license file. In addition, any file whose name ends in **.lic** in the current directory when the RLM server is started (or when the rlmreread command is issued) is implicitly added to the end of the license file path. Finally, any file whose name ends in **.lic** in the directory where the RLM binary resides is added to the list of license files processed.

Note License files in the ISV server's binary directory are **not** processed, only the RLM binary directory is searched.)

License servers ignore *port@host* specifications in the License Environment. Once the list of license files is created, the license servers process all the resulting license files. The RLM server starts all ISV servers in all license files, and the ISV servers process and support all licenses in all license files with valid hostids.

When the RLM server starts, it uses the port # from the first file with the hostname on which it is running. The RLM server will attempt to use all the port #s in all the license files. It **must** be able to bind the port # in the first file. Once this is done, it attempts to use the port number from each additional file, logging either success or failure in the debug log. This means that when you receive a new product or license file, it can be installed in the application and RLM server directories without changing the port number in that file, which simplifies license administration.

ISV servers process all licenses in all files that have a valid *hostid* (meaning a *hostid* that corresponds to the computer on which the license server is running). The ISV servers attempt to combine licenses whenever possible - see the next section - and when combined the license counts add to create a single *pool* of licenses. ISV servers log (in the debug log) licenses with invalid signatures and licenses that will expire within 14 days. ISV servers do not process single-use (count==*single*) licenses.

ISV servers will detect that they are running on a virtual machine, and by default will refuse to run. The decision to run or not on a virtual machine is controlled by your ISV, and Reprise Software has no part in this decision.

1.3.7 ISV server open file limits

On Unix platforms, ISV servers will attempt to increase their open file limit. If a server is able to increase its open file limit, a line similar to the following will appear in the debug log when the server first starts up:

```
mm/dd hh:mm (isvname) File descriptor limit increased from 256 to 65536
```

If you do not wish the ISV server to unlimit it's open descriptor limit, set the RLM_NO_UNLIMIT environment variable in the process where you run the server:

```
% setenv RLM_NO_UNLIMIT anything
```

1.3.8 How Licenses are Pooled by the ISV Server

When the ISV server processes all its licenses in the license file, it combines as many as possible into single *pools* of licenses. In order for 2 licenses to be combined into a single license pool, the following license fields must match **exactly**:

- Product Name
- Product Version
- License Sharing specification
- License Timezone specification
- License Platform list
- Both licenses must be counted or uncounted
- License node-locked hostid
- Both licenses must be user-based or host-based (or neither)
- Neither license can be a named-user license
- License password
- License id (also, an id of 0 will pool with any prior license with non-zero id)

Once pooled, the following fields are processed as shown:

Field	Result
count	both counts added together
exp-date	earlier date is remembered
hold	maximum of the 2 values
max_roam	minimum of the 2 values
min_checkout	maximum of the 2 values
min_timeout	maximum of the 2 values
soft_limit	both soft_limit values added together
contract	if original is empty, use new
customer	if original is empty, use new
issuer	if original is empty, use new
type	if original is empty, use new

For all other fields, the field in the original license (i.e., the first to appear in the license file) is used. Note that different **named_user** licenses are never combined into one license pool.

The id of a license can affect license pooling as follows: A license that doesn't specify an id (or specifies 0), will pool with any other license that it would normally pool with. However, a nonzero id will only pool with the same ID# (assuming all the other attributes make it eligible to pool).

1.3.9 License Server Administration

There are various administration commands that can be used to cause the license servers to reread their license files, to remove licenses from certain users, etc. For a description of these administration commands, see Section 1.5. In addition, options can be specified for each ISV server in Section 1.9. You can restrict access to administration commands via Section 1.8.

1.3.10 Client timeouts connecting to the server

There are 2 different timeouts that clients can experience when talking to license servers: CONNECT timeouts and READ/WRITE communication timeouts.

By default, the CONNECT timeout is set to 10 seconds. To change this, set the environment variable RLM_CONNECT_TIMEOUT to a positive integer, e.g.:

```
% setenv RLM_CONNECT_TIMEOUT 5
```

to set the connect timeout to 5 seconds.

The read/write timeout is set to 5 seconds by default. To change this, set the environment variable RLM_COMM_TIMEOUT to 1000 times the timeout desired, e.g.:

```
% setenv RLM_COMM_TIMEOUT 10000
```

to set the comm timeout to 10 seconds.

1.3.11 A note on server lock files

On occasion, when you start the RLM servers, you will see messages similar to these in the debug log:

```
08/08 16:11 (reprise) Cannot set server lock - exiting
08/08 16:11 (reprise) This usually means another copy of "reprise" is running
08/08 16:11 (rlm)
08/08 16:11 (rlm) reprise initialization error: 8, not restarting
08/08 16:11 (rlm)
08/08 16:11 (rlm) reprise - lockfile problems
08/08 16:11 (rlm) This usually means another "reprise" server is running
```

This happens if another copy of the ISV server is running. Check to make sure that another copy is not running.

This can also happen if the lockfile directory is not writable by the user who started the server. Ensure that the lockfile directory is writable by the user who starts the server.

The "RLM directory" is:

OS	Location
/var/tmp	Unix
/var/tmp	Mac
C:\ProgramData\Reprise	Windows

The server's lock file name is rlmlock<isv> on Windows, and .rlmlock<isv> on Linux, where <isv> is the ISV name. If you have lockfile problems on Linux which can't be attributed to another copy of the server running, then check the protection on /var/tmp to make sure it allows world rwx. On Windows, what sometimes happens when the first version of the RLM server to run on a system is pre-v9.4, is that the lock file is created with restrictive permissions that don't allow a different user to lock it. 9.4 and later create the Reprise folder with wide-open access that is inherited by files created in it, so this problem doesn't occur. Deleting the lock file is OK to solve the immediate problem.

1.4 The License File

The license file contains information which configures the license servers and describes all the licenses granted from the ISV to your customer. There are 4 basic license types:

- **counted**
- **uncounted**
- **token**
- **metered**

Everything else in the license file is a modification of one of these 4 basic license types. For example, if a counted license has a hostid associated with it, it becomes a **nodelocked, counted license**. If the counted license has no hostid, it is a **floating license**. The various attributes modify the basic license types. For example, if a counted license has a **share=** attribute, then multiple application instances can use a single license. If the license file has counted or metered licenses, it requires a license server, so a HOST and ISV line must be present. If the license file has only uncounted licenses, the HOST and ISV lines are not required.

License Files have 6 types of lines:

Line	Description
HOST	Specify the license server host (SERVER is an alias for HOST – see Section 1.4.3)
ISV	Specify the ISV's license server information (DAEMON is an alias for ISV – see Section 1.4.4)
LICENSE	Describe license grants from the ISV to your customer (FEATURE is an alias for LICENSE – see Section 1.4.5)
UPGRADE	Upgrade the version number of some or all LICENSEs, (see Section 1.4.7)
CUSTOMER	**RLM Cloud only** - See: Section 1.4.8
Comment	(see next section)

Applications, License Servers, and License Administration Tools locate the license file using The License Environment.

Note RLM does not support byte order markers (BOMs) in license files. If a BOM is present in a license file, RLM will not recognize the content.

1.4.1 Comments in license files

Lines beginning with '#' are treated as comments and not interpreted by RLM. Comments may be added to a license file without invalidating the signatures of licenses, but should not be added between the lines of a multiple-line license. Here is an example:

```
#
# Licenses served by host gt2
#
HOST gt2 0000a74f88ce 5053
ISV reprise
#
# Original license for v3.0 (10 seats)
#
LICENSE reprise joe 3.0 permanent 10 _ck=f81efcf79a
sig="60PG451KTXVQ0WYBX785XAKTDKUCHB7T683Y2MG22M088S8UAFR0VKPMFGPKH
4XW4H5QQ8JSFFJG"
#
# v4.0 license (5 additional seats)
#
LICENSE reprise joe 4.0 permanent 5 _ck=3b1efcd48c
sig="60P045145JSKEJSR48V3GXCX29S8TM5TKE91TS022HW0XAEEWH82DRTCJB830AW
```

```
EV62MUE2N7C"
```

Note Prior to RLM v9.3, the comment character was not strictly required on comment lines. With improved error checking in 9.3 however, the comment character is required.

Special License Names

Any product name beginning with "rlm_" is reserved to Reprise Software.

The product name *rlm_roam* is treated specially by RLM. *rlm_roam* indicates that roaming has been enabled by an ISV. If an ISV issues an *rlm_roam* license, then roaming is enabled on any computer which is able to check out the *rlm_roam* license while in a disconnected state.

Legal characters in the license file

In general, all license file fields are white-space delimited, meaning that no data item can contain embedded spaces, tabs, newlines or carriage returns. In addition, the following six characters are illegal in data items in the license (and options) file except as noted below: "<", ">", "&", single quote (’), back-quote (`) and double-quote (").

As of RLM v12.2, "<", ">", "&", single quote (’) and back quote (`) are legal characters in the customer field.

ISV license names cannot begin with the characters "rlm_".

Note All lines in license files as well as option files (RLM or ISV) **must** be shorter than 1024 characters. Anything over 1024 characters will be truncated.

Everything in the license file is case-insensitive, with the following three exceptions:

- *isv-binary-pathname* on ISV lines [**Case-sensitive on Unix systems only**]
- *options-file-filename* on ISV lines [**Case-sensitive on Unix systems only**]
- Short (~62-character) license keys (keys with bits/character of 6 - see creating-licenses)

Note Any time RLM processes a *username*, it will replace any white space in the name with the underscore '_' character. This is true for *usernames* used as hostids, in server option files, or passed between client and server. Also note that *usernames* are case-insensitive – they are converted to all lowercase in the license server.

The following sections describe each of the three types of license file lines (Section 1.4.3, Section 1.4.4, Section 1.4.5).

1.4.2 Order of lines in the license file

In general, the order of lines in the license file does not matter, with the following exceptions:

- For an ISV/DAEMON line which specifies a password, all LICENSE/FEATURE lines following the ISV/DAEMON line which do not specify a password will use the password of the ISV/DAEMON line which precedes it. If the ISV/DAEMON line follows the LICENSE/FEATURE line, the password will not apply to that LICENSE/FEATURE.
- LICENSE/FEATURE lines are processed in the order they appear in the license file. This means that you can bias the selection of licenses by the order they appear in the license file. For example, if you have licenses for product ABC versions 1.0 and 2.0, and your software requests version 1.0, the license you receive will depend on the order: if the 2.0 license appears first in the license file, and it is available, your application will receive a v2.0 license. If the v1.0 license appears first and it is available, you will receive a v1.0

license.

1.4.3 HOST Line

```
HOST hostname hostid [tcp/ip port #]
```

The HOST line specifies which computer the license server is to run on. There is only one HOST line per license file. Note that if a license file contains only nodelocked-uncounted licenses, a HOST line is not required.

The *hostname* is the standard TCP/IP hostname for the system. This name is not an input to the license key signature algorithm, and thus can be changed at any time.

The *hostid* is RLM's idea of the computer's identification. The *hostid* is an input to the license key signature algorithm, so it cannot be changed. All licenses in the license file have this *hostid* as input to their license signatures, so it is important to avoid moving LICENSE lines from one license file to another, as this invalidates them.

The tcp/ip port number is the port which RLM attempts to use for communications. This number can be changed to any available port.

For a description of the various hostids that RLM supports, see rlm-host-ids.

Example:

```
HOST melody 80b918aa 2700
```

In this example, the license servers run on host "melody" at TCP/IP port # 2700.

Note The keyword "SERVER" can be used instead of "HOST".

1.4.4 ISV Line

Format:

```
ISV isvname [isv-binary-pathname [options-file-filename [port-number]]]
[binary=isv-binarypathname]
[options=options-file-filename] [port=port-number] [password=password-text]
```

The ISV line specifies an ISV's license server. There is one ISV line in the license file for every *isvname* which has licenses in that file. Note that if a license file contains only nodelocked-uncounted licenses for a particular ISV, an ISV line is not required for that ISV.

Parameter	Required	Description
`isvname`	Yes	The name assigned to the ISV - **does not change**.
`isv-binary-pathname`	Yes*	Location of the ISVs license server binary. Can be any accessible location. Not an input to license key signature algorithm so can be changed any time. * *Can be omitted if ISV server is in same location as the RLM binary.*

`options=options-file-filename`	No	Specifies whether an options file will be used (see: Section 1.9). Specify options file location here, or name the ISV options file *isvname*.opt and place in same directory as license file.
`port=port-number`	No	Specifies port ISV server will use. If omitted, port is allocated dynamically.
`password=password-text`	No	Specifies license password applied to all LICENSE or FEATURE lines which follow the ISV line in the license file. If individual LICENSE/FEATURE line has a password, password from LICENSE/FEATURE line is used.

Any of the optional parameters can be specified by themselves. Also note that any number of the positional parameters can be specified, and optional parameters can be added after the positional parameters.

Note If you specify the same parameter both as a positional parameter and as a keyword=value parameter, the value of the keyword=value parameter will be used.

Examples:

License server for ISV *reprise* is located at /home/reprise/reprise, options file is located at /home/reprise/reprise.opt:

```
ISV reprise options=/home/reprise/reprise.opt binary=/home/reprise/reprise
```

License server for ISV *reprise* located at /home/reprise/reprise and ISV server port # specified:

```
ISV reprise /home/reprise/reprise port=8765
```

License server for ISV *reprise* located at /home/reprise/reprise and ISV server binary name /a/b/reprise used instead of /home/reprise/reprise:

```
ISV reprise /home/reprise/reprise binary=/a/b/reprise
```

Note The keyword "VENDOR" can be used instead of "ISV".

1.4.5 LICENSE Line

Format:

```
LICENSE isv product version exp-date count [sig=]license-key [optional
parameters]
```

The LICENSE line defines the usage rights to a product. All fields in the license line are case-insensitive (with the exception of short, i.e., less than 62-character, license keys), and none may be modified by the license administrator, with the exception of the parameters whose names begin with the underscore ("_") character.

Note The license file parser will reject unknown keywords.

Fixed (positional) parameters

The first 6 parameters are required on every license and are present in the order shown above.
isv

The name of the ISV granting the rights.
product

The name of the product for which license rights are being granted.
version

The highest-numbered product version supported by this license, in the form "N.M". For example, 1.0, 2.37, or 2006.12. Each RLM license has a version number, of the form "major.minor". The version in the rlm_checkout() call must be less than or equal to the version in the license for the checkout to succeed. (Note: This comparison is done in the "normal" way, ie, 1.2 is greater than 1.10).

The version can be used in a number of ways:

- You could make all your software ask for version 1.0 with all your licenses issued for version 1.0, and the version would never be an issue, unless and until you wanted to obsolete all the old licenses on a new release.
- You could put your product's version number in the rlm_checkout() call, then licenses for an older version of your product will not work with a newer version of the product.
- You can use a date-based version. To do this, you might put the year and month of release into the rlm_checkout() call in your application, then when you issue licenses, issue them either for this year and month when your customer's maintenance period ends.
  This allows your customer to use products released on or before the date in the license. Bear in mind that you would need to use the leading 0 in the month, since 2006.2 is greater than 2006.11, which might not be what you intend.

exp-date

The date the license expires, in the form dd-mmm-yyyy, for example, 1-jul-2007. All licenses have an expiration date. If you prefer for your licenses to not expire, you can use the special expiration date of **permanent**, which never expires (any date with a year of 0 is also non-expiring, e.g. 1-jan-0). The license expires at midnight on the date specified, in other words, it is valid for the entire day of the expiration date.

You can also specify an expiration time, using the Expiration time keyword.

The license expiration date can be either **dd-mmm-yyyy** (e.g., 01-jan-2023), or it can be formatted entirely numerically as **yyyy-mm-dd** (e.g., 2023-01-01).

RLM uses a proprietary algorithm to check for clock windback. This algorithm does not access any other computers and has been used in RLM since version 1.0. It is fast but sometimes returns false positives.
count

The number of licenses granted. The count field defines the license type. See the license-models for a discussion of license types and modifiers. The license type is one of:

- A positive integer indicates a **counted license**.
- 0 or "uncounted" indicates an **uncounted license**.
- "single" means a node-locked, single-use license. single is a special case of a **counted license**, but it is different from "1". A license with a count of 1 is a regular counted license and requires a license server. A license with the keyword "single" is a single-use, node-locked license. This license does not require a license server, and in fact license servers will not process this license. single licenses are a convenient way to issue single-use licenses without the license administrator having to configure a license server.
- token, token_bound, and token_unlocked are used to specify a **Token-Based License**; this

license must also have the token=… optional parameter (see Section 2.1). The only optional parameter on a token-based license which is used by RLM is the start date. All other optional parameters are ignored.

- Meter indicates a **metered license**. See metered-licenses for a complete description of metered licenses.

license-key

This is a digital signature of all the license data, along with the hostid on the HOST line, if present. If a license has a non-zero count, it always requires a HOST line. An uncounted license does not require a HOST line, and even if there is a HOST line, the hostid of the license server is not used in computation of its *license-key*. The *license-key* will have "sig=" prepended after the license has been signed by the *rlmsign* utility.

Note If the license-key is preceded by sig=, it can be present after any or all of the optional parameters.

Licenses can also have the following optional license modifier attributes:

Locking: Node-locked, Username-locked, or Floating licenses.

RLM can lock a license in a variety of ways:
node-locked (counted or uncounted)

A node-locked license can only be used on a single node, as specified by the hostid of the license. For a description of the available hostids in RLM, see rlm-host-ids.

The hostid in a license can be a *hostid list*, which means that the license is usable **on any of the specified hostids**.

A node-locked license can be either counted, uncounted, or "single". If it is uncounted or single, then the software need only verify that it is executing on the correct computer, and no license server is required. If it is counted, however, a license server is required to maintain a count of licenses currently in use.

Note While uncounted nodelocked licenses can be served by a license server, single licenses cannot.)

To create a node-locked license, add the keyword **hostid=..** at the end of the license line.
username-locked (counted or uncounted)

A license can be locked to a user. This is a special case of a *node-locked* license, and is accomplished using the hostid **user=….**

Note Any white space in a username is converted to the underscore ('_') character. Also note that usernames are case-insensitive.

---

floating

A license can be floating. This license will work anywhere on the network that can communicate with the license server. To specify a floating license, do not put a hostid= keyword on the license.

hostid=hostid-string (used for license locking)

The optional hostid at the end of the line specifies that the licenses can only be used on the specified host. Uncounted licenses always require a hostid. Counted licenses generally do not have a hostid, but it could be present, in which case we would call this license a "node-locked, counted" license. (For a description of the various hostids that RLM supports, see rlm-host-ids.

The hostid on a LICENSE line can be a hostid list. The hostid list is a space-separated list of valid hostids, enclosed in double-quotes. The license can be used on any of the hostids in the list.

The list can contain at most 25 hostids, and can be no longer than 200 characters.

For example, this hostid list would allow the license to be used in any of the 4 specified environments:

```
hostid="ip=172.16.7.200 12345678 rlmid1=83561095 user=joe"
```

Other optional license parameters

Activation Key used to create this license

akey=activation-key

When requested in RLM Activation Pro, the license generator will include the akey= keyword with the activation key used to fulfill the license. The maximum length of the activation key is 40 characters, including the "akey=" part.

Caching Licenses on the Client Node

client_cache=seconds

When specified, the license will be held by the license server for "seconds" seconds, in exactly the same way as a "minimum_checkout" attribute would. However, on the client side, license data will be cached which can be used by subsequent checkout requests. The maximum cache time is one hour (3600 seconds). Note that the client-side caching will be ineffective if not enabled with rlm_isv_cfg_enable_client_cache() in rlm_isv_config.c, however the server will still hold the license checked-out in this case.

You can think of this as a very short-term, automatic, roam.

You should note that only licenses which have a sharing attribute will be usable as cached licenses. In other words, if the license has a "share=u" attribute, and the same user attempts a checkout, the cached license can be used. If the "share=" attribute is missing, no checkouts of the cached license will be possible. Also note that the host attribute is matched automatically, by definition, since the license is being used on the same computer.

Your customer can modify the value of client_cache between 0 and 2x the value you specify with the license administration CLIENT_CACHE option. See Section 1.9.3 in the License Administration manual for more details.

Disable Computing Environments

disable="computing-environment-list"

disable= specifies that clients running in the appropriate computing environment cannot use this license.

computing-environment-list is a list of different computing environment descriptions; if the application is running in any of these environments, the license will not be usable.

computing-environment-list is a space-separated list of the following environments. Put the list in quotes if more than one item is specified.

Environment	Description
TerminalServer	Disable use on Windows Terminal Server and Remote Desktop.
TerminalServerAllowRD	Disable use on Windows Terminal Server but allow use via Remote Desktop.
VM	Disable use on Virtual Machines.

Disabling TerminalServer is most useful for node-locked, uncounted licenses, if you do not want to allow multiple network users running remote sessions to make use of a single license.

Note that you can't disable both TerminalServer and TerminalServerAllowRD – they are mutually exclusive.

Disabling Virtual Machines is useful for node-locked, uncounted licenses in order to prevent these licenses from being used on multiple virtual machines with the same hostid.

Example:

```
disable=TerminalServer
```

Expiration time

exptime=hh:mm

Beginning in RLM v14.1, a license can be set to expire at a particular time on the expiration day. If the license includes an "exptime" keyword, the license will expire at this time on the expiration date rather than at midnight. If the exptime keyword isn't present, the license expires at midnight, as it has always done in previous versions of RLM. The expiration time, like the expiration date, is in local time either for the client (for nodelocked licenses) or on the server (for floating licenses).

Hold time and Minimum Checkout

hold=N and **min_checkout=n**

By specifying *hold=N* in the license, the license will be held for N seconds after the application exits or checks the license back in via rlm_checkin(). hold is typically used in the case of licenses that have a very short duty-cycle, in order to provide a "fairer" measure of concurrent usage.

min_checkout specifies that a license is to be "held" checked-out by the license server after the application performs a check-in call or exits, only if the license did not remain checked out for the minimum interval specified by *n*. n must be a positive integer, greater than 0. The license will remain checked-out such that the total checkout time is n seconds. *min_checkout* is typically used in the case of licenses that have a very short duty-cycle, in order to provide a "fairer" measure of concurrent usage.

hold and *min_checkout* both perform this function in slightly different ways. hold always keeps the license checked out for the specified amount of time, whereas *min_checkout* keeps the license checked out for an additional time only if the license was checked back in by the application before the specified minimum time.

Note Both *hold* and *min_checkout* time specifications are ignored for any license which is roaming. Also note that hold time is processed by the license server, so it has no effect on unserved nodelocked licenses.

License ID

_id=nnn

Any License Administrator can add *_id=nnn* to a license. "nnn" is a positive integer, less than 2^{31}, which is used to identify the license. If no _id= keyword is present, the id of the license is 0. The id of a license can affect license pooling as follows: A license that doesn't specify an id (or specifies 0), will pool with any other license that it would normally pool with. However, a non-zero id will only pool with the same ID# (assuming all the other attributes make it eligible to pool).

In addition, beginning in RLM v12.0 licenses are sorted (within a license file) in the order of the _id keyword. Licenses without _id keywords will remain unsorted (in their original order) at the end of all the licenses with _id keywords, which are sorted in increasing numerical order. This sort is done prior to REPLACE processing.

Other than license pooling and sorting, the id can be used to select which licenses to apply an option (such as RESERVE). The id is not used in the computation of the license signature, and as such can be added or changed by the License Administrator.

License Issue Date

issued=dd-mmm-yyyy

If *issued=dd-mmm-yyyy* is specified in the license, this license issue date will be used in the computation of license replacement. If no issue date is present, the license start date is used. If neither is present, then this license will be replaced by any license specifying a *replace=* keyword with this license's product name.

Maximum Roam Count

max_roam_count=num

A Roaming license is a license that is checked out from the license server and used on a disconnected system. During this time, the license server holds the license checked-out just as if the system were still connected to the license server. Roaming licenses are only allowed if you issue your customer an *rlm_roam* license that is valid on the disconnected system.

If you do not specify *max_roam_count* in an individual license, RLM allows the total number of licenses to be roamed. Setting *max_roam_count* to a number less than the total number of licenses will cause the server to only allow that number of licenses to roam. To disable roaming on a particular license, set *max_roam_count=0* for that license.

Note The ISV-supplied *max_roam_count* attribute is equivalent to the license administrator MAX_ROAM_COUNT option, except that *max_roam_count* in the license is always enforced, rather than being optional.

Maximum Roam Days

max_roam=days

If you do not specify *max_roam* in an individual license, RLM limits the maximum number of days that a license can roam to 30 days. To disable roaming on a particular license, set *max_roam=-1* for that license.

Specifying *max_roam* on the *rlm_roam* license itself will potentially lower the max_roam of all products. The effective *max_roam* is the minimum of the value specified in the license itself (or the default value) and the value in the rlm_roam license. So, for example, if you set *max_roam=20* on the rlm_roam license, then all licenses without a max_roam will be limited to 20 days. If you set *max_roam=40* on the rlm_roam license, then only individual licenses with max_roam set to **greater** than 40 days will be affected.

Metered licenses

Metered licenses are described in detail in metered-licenses. Metered licenses use the following 4 keywords:

- **meter_counter=counter#**
- **meter_dec=decrement_amount**
- **meter_period=period_in_minutes**
- **meter_period_dec=periodic_decrement_amount**

Named User licenses

named_user[=num] or **named_user="num min_hours"**

Named User Licenses allow the ISV to require that a list of users be created who can use the license(s). The number of users in the list can be less than, equal to, or greater than the number of licenses available. Once a user is added to the list, they can be deleted, but once deleted, they must remain off the list for a minimum number of hours (24 hours by default).

To create a **named user** license, add the **named_user** keyword to the LICENSE:

named_user	Require the same # of users as there are licenses.
named_user=n	Require *n* users to be named.
named_user="n min_hours"	Require *n* users, and specify the minimum number of hours.

With a **named_user** license, the license server can construct the list of users automatically as

license checkouts occur, or the list can be entered via the RLM web interface by the license administrator. If entered manually, either individual users or GROUP names (as defined in the ISV server options file) can be used.

named_user licenses utilize the INCLUDE functionality of the license server, and do not need the entire list of **num** users specified before the licenses can be used. In fact, no users need to be specified since the license server will add users who do not appear on the list if the current list size is less than the number of allowed named users.

Once a user is added to the list, they can be removed at any time. However, once removed, a user cannot be placed back on the list until they have been off the list for **min_hours** hours (default: 24 hours).

Note

- Different named_user licenses **will never be combined** into one license pool, even if all other license parameters match.
- named_user utilizes the INCLUDE list in the server, so **any INCLUDE specification for this license will be ignored**.
- named_user is processed by the license server, so it **has no effect on unserved nodelocked licenses**.
- **usernames are case-insensitive**, so user "Joe" is the same as user "joe".

License Options

options = options_list

The *options* specification is used to encode options for the product license. The options field is a string (up to 64 characters in length) which is completely defined by the ISV. The options are used to calculate the license signature, but otherwise are unused by RLM. You can retrieve the options from a license with either the *rlm_product_options()* or the *rlm_license_options()* call.

Note If the string contains embedded white space, it must be enclosed within double quotes.

You can specify a required substring in the options field with the *rlm_set_attr_req_opt()* call. See rlm-set-attr-req-opt for more information.

License Password

_password = password-string

A license password restricts access to this license to requests which have specified the same password-string. The password-string is specified either with the *rlm_set_attr_password()* call, or (for the command-line utilities) with the RLM_LICENSE_PASSWORD environment variable, or in the RLM web interface. The password string must be <= 32 characters.

Note The license password does not factor into the license signature, and hence can be changed at any time after the license is signed. Also note that license passwords only work with served licenses, not nodelocked, uncounted or single licenses in local license files.

If a request (checkout or status) is made for a license which contains a password, and the request does not specify a matching password, the license server will return RLM_EL_NO_SERV_SUPP (for a checkout request) or no data (for a status request). This prevents a user from knowing that a license exists and attempting to guess the password.

The license password can be specified on the ISV line with the password=password-text optional field. If specified on the ISV line, the password applies to all LICENSE or FEATURE lines which follow the ISV line in the license file. If any individual LICENSE/FEATURE line specifies a password, the password from the LICENSE/FEATURE line is used.

Platform Restrictions

platforms=platform_list

RLM allows you to specify one or more platforms on which the application must be running. If a *platforms=platform-list* specification is contained in the license, the computer on which the application is running must be one of the specified platforms.

To specify one or more platforms, create a list of platform names. The *platform-list* consists of a list of RLM-defined platform names, which consist of a machine architecture and an operating system version/revision. Specify *platforms=* as a space-separated list of platform names with the trailing OS revision removed, as shown in the following table.

Note If you specify more than one platform, enclose the entire string in double quotes (e.g., platforms="sun_s x86_w sun64_s").

Also note that while you can include the trailing revision number, it will not be used by RLM in any comparisons, so including it may lead to confusion.

Platform	RLM Platform name	string to use in *platforms=*
HP//UX on PA-Risc, 32-bit	hp_h1	hp_h
HP//UX Itanium, 64-bit	ia64_h1	ia64_h
IBM AIX, 32-bit	ibm_a1	ibm_a
IBM AIX, 64-bit	ibm64_a1	ibm64_a
IBM Power Linux, 64-bit	p64_l1	p64_l
Linux Intel, 64-bit	x64_l1	x64_l
Linux ARM, 64-bit	arm64_l1	arm64_l
Linux, 32-bit	x86_l2	x86_l
Linux, 64-bit	x64_l1	x64_l
Linux Itanium, 64-bit	ia64_l2	ia64_l
Linux PPC, 64-bit	ppc64_l1	ppc64_l
Mac Arm, 64-bit	arm64_m2	arm64_m
Mac Intel, 32-bit	x86_m1	x86_m
Mac Intel, 64-bit	x64_m2	x64_m
Mac PPC, 64-bit	ppc_m1	ppc_m
NetBSD, 32-bit	x86_n1	x86_n
Solaris Sparc , 32-bit	sun_s1	sun_s
Solaris Sparc, 64-bit	sun64_s1	sun64_s
Solaris, 64-bit	x64_s1	x64_s
Windows Intel, 32-bit	x86_w3, x86_w4	x86_w
Windows Intel, 64-bit	x64_w3, x64_w4	x64_w

Replacement Licenses

replace[=product_list]

In order to render ineffective one or more licenses which you have already issued, use the *replace[=product-list]* option in the new license. *replace=* causes RLM to ignore the "replaced" license(s). If *product_list* is the single character '*', **all licenses will be replaced**.

Note If you specify replace, you must also specify either start= or issued=.

---

replace operates as follows:

- Licenses from the *product_list* will be replaced (all licenses if product_list is '*'). If *product-list* is not specified, then the product name of the license containing the replace keyword will be the only product to be replaced.

- If the license with the *replace* keyword specifies an *issued= date*, then this is the "*replacement date*".
- If the license with the *replace* keyword does *not* have an issued date, then the "*replacement date*" is the *start* date of the license.
- If the license contains neither an *issued* date nor a *start* date, no licenses will be replaced.
- Any license in the list of products with an *issued* date prior to the *replacement date* will be replaced.
- Any license in the list of products which does not have an issued date, but which has a *start* date prior to the *replacement date* will be replaced.
- Finally, any license in the list of products with neither an *issued* nor a *start* date will be replaced.
- **EXAMPLE**: To replace products "a" and "b", use: **replace="a b"** in the license.

Warning If two replace licenses have the same issued/start date, **they will both be active** after the replace determination has been made.

Effective Start Date

start=dd-mmm-yyyy

If *start=dd-mmm-yyyy* is specified in the license, the license cannot be used before the specified date.

License Soft Limits.

soft_limit=n

A license can have a *soft_limit* that is lower than its count of available licenses. Once usage exceeds the *soft_limit*, checkout requests will return the RLM_EL_OVERSOFT status instead of a 0 status.

Note When the license server pools separate licenses into a single license pool, the *soft_limit* of each license is added to obtain the pool's soft_limit. Licenses which do not specify a soft_limit use the license count as their soft_limit.

Soft limits are processed by the license server, so they have no effect on unserved nodelocked licenses.

Sharing of licenses between different processes

share=UHI[:nnn]

Licenses can be shared between separate running processes by using a *share=* specification in the license. A license can be shared between processes with the same **username**, **hostname**, or **ISV-defined** data; or any combination of these. Specify share as *share=UHI* where the keywords 'U', 'H', and 'I' indicate that the Username, the Hostname, or the ISV-defined fields must match. **If more than one is specified, all specified must match in order to share the license**.

Note Usernames and hostnames in RLM are case-insensitive.

The share= keyword accepts an optional maximum process count which can share the license. To specify a maximum process count for a license that is shared by user, use **share=U:nnn**, where *nnn* is the number of processes which can share this license. The nnn+1st request will consume an additional license.

If the :nnn specification is omitted, any number of processes can share the license.

Note Once a shared license is in use, it will continue to be in use until all the processes sharing the license check it back in. In other words, if 2 processes are sharing one license, and a 3rd process consumes a 2nd license (in the case where n==2), 2 licenses will continue to be in use

until either (a) the 3rd process checks in its license, or (b) BOTH the first and second processes check in their licenses. In other words, there is no dynamic re-combination of shared licenses done at license check-in time.

Note Share is processed by the license server, so it has no effect on unserved nodelocked licenses.

Table 1.1. Examples

share=UH	Both username and hostname of a request must match.
share=U	Only the usernames must match on two processes to share license.
share=U:2	With matching usernames, one license is consumed for every two shares.

User, Host-based or Personal licenses.

user_based[=n] host_based[=n] personal (RLM Cloud only)

User-based licenses allow the ISV to require that the specified number of licenses (or all licenses) must be reserved to users (with RESERVE lines) in the options file. Note that the special user '*' does not count as being reserved. If fewer than the required number of licenses are reserved, the license server will log an error and discard the license. Also note that licenses reserved to a GROUP are not counted, otherwise the license administrator could simply bypass the intent of this license by creating a GROUP consisting of all their users. Thus, all reservations must be to individual users.

Similarly, host-based licenses allow the ISV to require that the specified number of licenses (or all licenses) must be reserved to hosts (with RESERVE lines) in the options file. Note that the special host '*' does not count as being reserved. If fewer than the required number of licenses are reserved, the license server will log an error and discard the license. Also note that licenses reserved to a HOST_GROUP are not counted, otherwise the license administrator could simply bypass the intent of this license by creating a HOST_GROUP consisting of all their hosts. Thus, all reservations must be to individual hosts.

Personal licenses (which are only available on RLM Cloud servers) allow the ISV to require that the authorized users be preassigned with the RLM Cloud portal GUI.

To create either user-based, host-based, or personal licenses, add the appropriate keyword to the LICENSE:

user_based[=nnn]	For user-based licenses.
host_based[=nnn]	For host-based licenses.
personal	For personal licenses.

If the optional "=nnn" is specified, only "nnn" of the total number of licenses need to be reserved, otherwise all licenses must be reserved.

For personal licenses, no count is specified, the personal user count is the license count.

Note user_based and host_based are processed by the license server, so they have no effect on unserved nodelocked licenses.

Timezone Restrictions

timezone=timezone-spec

RLM allows you to specify one or more timezones in which the applications must be running. If a *timezones=timezone-spec* specification is contained in the license, the computer on which the application is running must be set to one of the specified timezones.

To specify one or more timezones, create a bitmask of the desired timezones, expressed as hours west of GMT:

Bit 0 - GMT Bit 1 - 1 hour west of GMT Bit 2 - 2 hours west of GMT … Bit 23 - 23 hours west of

GMT (or 1 hour east of GMT)

This bitmask should be represented as a hex number. So, for example, to allow your application to run in the GMT timezone only:

timezone=1

To allow your application to run in timezone 8 (PST):

timezone=100

To allow your application to run in timezones 5-8 (continental USA):

timezone=1E0

Minimum rlmremove interval

min_remove=n

min_remove specifies the lowest value, in seconds, a License Administrator can set the MINREMOVE value for a license.

If not specified in the license, the RLM default of 120 seconds (2 minutes) is used.

If min_remove is set to -1, then rlmremove (and the Remove button in the RLM web interface) is disabled on this license.

Note min_remove is processed by the license server, so it has no effect on unserved nodelocked licenses.

Minimum Timeout specification

min_timeout=n

A license administrator can specify a TIMEOUT value for any idle license. If the license remains idle (i.e.. does not communicate with the license server) for this amount of time, the license server performs an automatic check-in of the license and informs the application (if it is still running).

min_timeout=n specifies the lowest value a license administrator can set for the timeout value for a license. If not specified in the license, the RLM default minimum of 3600 seconds (1 hour) is used.

Note Timeout is processed by the license server, so it has no effect on unserved nodelocked licenses.

Additional Fields

The following fields are not used by RLM, but are present to identify licenses, or can be used in your application to present to the user:

contract=contract-info

contract= is meant to hold the customer's purchase order or software agreement number. This can be used to display to the user to validate a support contract, etc. It is not used by RLM. Maximum of 64 characters.

customer=who

customer is to identify the customer of the software. This can be an added incentive to keep honest users honest, as it is unlikely that Mega South-East Airlines would want to use a license that was issued to Main St. Bank., for example. customer is not used by RLM. Maximum of 64 characters.

issuer=who

issuer= is used to identify the organization which issued the license. It is not used by RLM. Maximum of 64 characters.

_line_item="descriptive_text"

The *_line_item* field is used to map a particular product to the item purchased. This field will be logged into the report log at the start when all products supported are logged, so that a report writer can generate reports based on purchased products, as opposed to product names used for licensing purposes. If the descriptive text contains spaces, it should be enclosed in double-quote (") characters. The contents of the *_line_item* field can be modified (or the field can be added) without invalidating the license signature. Maximum of 64 characters.

type=type-spec

type= is used to identify the type of license. type-spec is a string containing one or more of the values:

- "beta"
- "demo"
- "eval"
- "subscription"

(For example, type="beta eval" or type="eval". The contents of the license type field are retrieved by the rlm_license_type() call (see rlm-license-xxxx). type is not used by RLM.)

Table 1.2. The maximum length and types of license fields are as follows:

Field	Type	max data length (excluding keyword=) or value range
isv	string	10 characters
product	string	40 characters
version	string, in the form nnn.mmm	10 characters
exp-date	string of the form dd-mmm-yyyy	11 characters
count	positive integer	2^{31} - 1
hold	positive integer - seconds	2^{31} - 1
host_based	int	2^{31} - 1
hostid (single)	string	75 characters
hostid (list)	space-separated, quoted string	200 characters, max of 25 hostids
hostname	string	64 characters
issued	string of the form dd-mmm-yyyy	11 characters
_line_item	string – license administrator defined	64 characters
max_roam	non-negative integer - days	2^{31} - 1
max_roam_count	non-negative integer - count	2^{31} - 1
min_checkout	positive integer - seconds	2^{31} - 1
min_remove	integer - seconds (-1 for no remove available)	2^{31}
min_timeout	positive integer - seconds	2^{31} - 1
options	string	64 characters
password	string	32 characters
personal	int	2^{31} - 1
platform	string	80 characters
share	enumerated	3 ("uhi") + :integer
soft_limit	int	2^{31} - 1
start	string of the form dd-mmm-yyyy	11 characters
timezone	int	bitmap with bits 0-23 set
user_based	int	2^{31} - 1
contract	string – unused by RLM	64 characters

customer	string – unused by RLM	64 characters
issuer	string – unused by RLM	64 characters
type	string - consisting of "demo" "eval" "beta" and/or "subscription"	14 characters

1.4.6 Valid LICENSE Option Configurations

Only certain combinations of license types and options are valid. The following table indicates which options are valid for various license types. An X in a cell indicates that the attribute on the left is valid with the main license type in the column; empty cells indicate unsupported option configurations. All license administrator editable options (beginning with "_") can be added to any license type.

License Type →	Floating	Nodelocked	Single	Uncounted	Token	Metered
Attributes ↓						
HOST/ISV line	required	required	ignored	X	X	required
UPGRADE-able	X	X	X	X		
contract	X	X	X	X	X	X
customer	X	X	X	X	X	X
disable	X	X	X	X	X	X
hold	X	X		when served		
host_based	X					
hostid		required	required	required	X	X
issued	X	X	X	X	X	X
issuer	X	X	X	X	X	X
max_roam	X					
max_roam_count	X					
min_checkout	X	X				
min_timeout	X	X			X	X
named_user	X	X				
options	X	X	X	X	X	X
platforms	X	X	X	X	X	X
replace	X	X	X	X	X	X
start	X	X	X	X	X	X
soft_limit	X	X				
share	X	X				
timezone	X	X	X	X	X	X
type	X	X	X	X	X	X
user_based	X	X				

Examples

```
LICENSE reprise write 1.0 permanent uncounted 98743597834597345734987539749587345987345
hostid=IP=172.16.7.3
```

```
LICENSE reprise calc 1.0 1-aug-2008 5
98743597834597345734987539749587345987345
```

In the first example, the write product is licensed to a host with an IP address of 172.16.7.3. This is a non-expiring, node-locked, uncounted license. The second example is for 5 licenses of product calc which expire on the first of August 2008. The first license would not require

a HOST line, whereas the second one would.

Note The keyword "FEATURE" can be used in place of "LICENSE".

Note Licenses are always additive, in other words, the counts on 2 license lines of the same product/isv/version/hostid would be added together by the license server and the total number of licenses would be available.

1.4.7 UPGRADE Line

Format:

```
UPGRADE isv product from-version to-version exp-date count [sig=]upgrade-key
[optional
parameters]
```

The UPGRADE line defines an upgrade from an older version (*from-version* or higher) to a newer version (*to-version*) of an existing *product*. All fields in the upgrade line are case-insensitive (with the exception of short, i.e., less than 62-character, license keys), and none may be modified by the license administrator with the exception of the parameters whose names begin with an "_" character.

We refer to the license specified by the UPGRADE line as the *upgrade license*, and the license it operates on as the *base license*.

An UPGRADE license will convert *count* licenses of *product* of version *from-version* or higher into *to-version*.

Note An UPGRADE license will never operate on a base license that is >= to-version.

In order for the upgrade to be performed, certain parameters of the *upgrade license* must match the parameters of a *base license* in order for that license to be eligible to be **upgraded**.

Certain licenses can never be upgraded. In particular, token-based licenses (the token definitions themselves) will never be upgraded. Also, named-user and metered licenses are not eligible for upgrades.

In order for a license to be upgraded, the following parameters must match on the *base license* and the *upgrade license*:

- License Disable specification
- License Options
- License Sharing specification (share and max_share)
- License Timezone specification
- License Platform list
- Both licenses must have the same counting type: *counted, uncounted, or single*
- License node-locked hostid
- Both licenses must be user-based or host-based (or neither)

Note This list is the same as the criteria for a license server combining multiple licenses into a license pool.

If the license is eligible for an upgrade, *count* licenses of the base license will be transformed into *to-version* licenses. An **upgrade** can be performed on multiple *base* licenses, until the *count*

on the *upgrade license* is exhausted.

Note License "replace" processing is done (both in client and server) before upgrade processing. This means that an upgrade license should not specify replacement of the base license which it is going to upgrade, because the base license will no longer exist when the upgrade is done.

There are 3 upgrade cases:

- Fewer *base licenses* than *upgrade licenses* - in this case, the extra *upgrade licenses* are "wasted", and the license server issues a warning. All base licenses are upgraded.
- Same number of *base licenses* and *upgrade licenses* - all base licenses are upgraded.
- Fewer *upgrade licenses* than *base licenses* - in this case, *count* licenses are upgraded, and the remaining *base licenses* remain at their old version.

The 3rd case above is the most useful - if you are upgrading all instances of an existing license, a *replace* option on a new license will do this just as well. The only advantage to the *upgrade license* in the first two cases is that the *base license* is required, i.e., the *upgrade license*, by itself, grants no rights.

When a license is **upgraded** by the license server, the new licenses will have their parameters modified as follows. All other license parameters will be the same as the *base license*:

Field	Result for Served *counted (or uncounted)* licenses	Result for Unserved *single or uncounted licenses*
exp-date	earlier date is used	earlier date is used
hold	maximum of the 2 values	• undefined -
max_roam	minimum of the 2 values	• undefined -
min_checkout	maximum of the 2 values	• undefined -
min_remove	maximum of the 2 values	• undefined -
min_timeout	maximum of the 2 values	• undefined -
soft_limit (upgrading all)	Larger of 2 deltas from license count is used.	• undefined -
soft_limit (partial upgrade)	Upgrade soft limit is preserved for the new version, base license soft limit delta is preserved (minimum value 0) for the old version.	• undefined -
user_based, host_based (upgrading all)	If licenses have a specification, the value closest to the license count is used .	• undefined -
user_based, host_based (partial upgrade)	If licenses have a specification, upgrade spec is preserved for the new version; base license spec delta (countuser_based) is preserved (minimum value 1) for the old version.	• undefined -
contract	If base license is empty, use upgrade license. On partial upgrade, if upgrade license is empty, use value from base license for the new version.	If base license is empty, use upgrade license.

customer	If base license is empty, use upgrade license. On partial upgrade, if upgrade license is empty, use value from base license for the new version.	If base license is empty, use upgrade license.
issuer	If base license is empty, use upgrade license. On partial upgrade, if upgrade license is empty, use value from base license for the new version.	If base license is empty, use upgrade license.
type	If base license is empty, use upgrade license. On partial upgrade, if upgrade license is empty, use value from base license for the new version.	If base license is empty, use upgrade license.

Fixed (positional) parameters

The first 7 parameters are required on every upgrade line and are present in the order shown above.

isv

The name of the ISV granting the rights.

product

The name of the product for which license rights are being upgraded.

from-version

The lowest-numbered product version which is eligible for an **upgrade**, in the form "N.M".

For example, 1.0, 2.37, or 2006.12

to-version

The highest-numbered product version supported by the license once it is **upgraded**, in the form "N.M".

For example, 1.0, 2.37, or 2006.12

exp-date

The date the upgrade expires, in the form dd-mmm-yyyy, for example, 1-jul-2007. A non-expiring upgrade can be specified with either a year of "0" (i.e., "1-jan-0"), or simply the word "permanent".

count

The number of licenses to be upgraded. **0** or **uncounted** means an uncounted, node-locked license is to be upgraded. **uncounted and 0 are 100% equivalent**.

> **single** means a node-locked, single-use license is to be upgraded. single is different from 1. See Section 1.4.5 above for more information.
>
> **token**, token_locked, token_bound and token_unlocked are not allowed in an UPGRADE license. All other optional parameters are ignored.

> **Warning** The keyword token_locked was deprecated in v12.4 and replaced with token_bound. Licenses with token_locked keywords will continue to operate, but they will not be locked to the server's hostid).

upgrade-key

A digital signature of all the upgrade data, along with the hostid on the HOST line, if present. If an upgrade license has a non-zero count, it always requires a HOST line. An upgrade to an uncounted license does not require a HOST line, and even if there is a HOST line, the hostid of the license server is not used in computation of its *upgrade-key*. The *upgrade-key* will have "sig=" prepended after the license has been signed by the *rlmsign* utility.

Note If the upgrade-key is preceded by sig=, it can be present after any or all of the optional

parameters.

Optional Parameters

Optional parameters are sometimes present in a license and can be present in any order. Optional parameters are allowed only once in an UPGRADE line. The syntax and meaning of optional parameters for the UPGRADE line are identical to the same parameters for the LICENSE line. Note that **token** and **named_user** are not allowed on an UPGRADE line. See Section 1.4.5 for information on the optional parameters.

Example:

```
LICENSE reprise write 1.0 permanent 5 sig=.....
UPGRADE reprise write 1.0 2.0 1-aug-2015 5 sig=.....
```

In this example, the 5 licenses of the write product have been upgraded from v1.0 to v2.0.

Note This license file would require a HOST line, since the licenses are counted.

1.4.8 CUSTOMER line

```
CUSTOMER customer-name isv=isvname server=server-name port=port#
password=yourpw
```

The CUSTOMER line, used for the client side of RLM Cloud-served licenses only, is used to identify the correct license server for a particular customer.

customer-name

Your customer's assigned Customer name.

isvname

Your ISV name.

server-name

The fully qualified hostname of the RLM Cloud license server machine.

port#

Either 5053 for normal RLM communications, or for HTTPS transport, 443).

yourpw

Your assigned password for this server.

Note HTTPS support must be enabled by your ISV to be effective. Please consult with your ISV to see if HTTPS support is available.

Example:

```
CUSTOMER yourco isv=reprise server=ls423.rlmcloud.com port=5053 password=xyz
```

isvname

Your ISV name.

1.4.9 The License Environment

All software that uses RLM attempts to read a license file or communicate with a license server. The specification of the license file or the license server is done with the license environment.

If the software is ISV-specific (e.g., an application program), the first place that is checked is any license file or port@host specified in the environment variable *ISV*_LICENSE (if it is defined), where ISV is name of the ISV (e.g., REPRISE_LICENSE).

If *ISV_LICENSE* is not defined (in the case of ISV software), or for generic software (RLM utilities, the RLM server) the path specified by the environment variable RLM_LICENSE is checked, if RLM_LICENSE is defined.

If neither environment variable is defined, the program's default location for the license is checked next. In the case of the RLM utilities and the RLM server, this is any file ending in *.lic* in the current directory. Note that the RLM utilities will substitute the path specification from a *-c* option in place of the current directory.

Finally, any *.lic* file in the directory containing the binary will be checked.

The format of the environment variable (RLM_LICENSE or *ISV*_LICENSE) is:

```
license_spec1
```

or

Listing 1.1. Unix

```
license_spec1:license_spec2:license_spec3: .... :license_specN
```

Listing 1.2. Windows

```
license_spec1;license_spec2;license_spec3; .... ;license_specN
```

where:

license_spec is a license specification of the form:

- port@host - or -
- license_file_pathname - or -
- directory pathname (containing license files, all of which are added to the list) - or -
- <actual text of license>

Note The separator character is ':' for Unix systems and ';' for Windows systems.

Note The license file cannot be placed in a path where any component of the pathname contains the '@' character.

Example

On Unix/Mac:

```
% setenv RLM_LICENSE 1700@myhost:/home/reprise/reprise.lic
```

On Windows:

```
C:\ set RLM_LICENSE=1700@myhost:/home/reprise/reprise.lic
```

In this example, the server at port 1700 on host "myhost" is checked first, then if the request is not satisfied, the license file at /home/reprise/reprise.lic is used. Note that this search order may be modified if the environment variable RLM_PATH_RANDOMIZE is set.

Example 2

```
% setenv RLM_LICENSE 1700@myhost:<LICENSE isv prod1 1.0 1-jan-09 uncounted
hostid=any
key="1234...">
```

This example specifies the license for "prod1" directly in the environment variable, in addition to using the server at port 1700 at host "myhost".

On Windows, you can also set the environment variable via the control panel using these steps:

1. Go to Control Panel->System (or Control Panel->System and Security)
2. Click on "Advanced System Settings"
3. Click on "Environment Variables"
4. Enter RLM_LICENSE as the environment variable name and the license file directory as the value

---

Note Beginning in RLM v14.2, white space is trimmed from the start and end of the license environment variable value.

Note Beginning in RLM v15.1, uppercase ISV_LICENSE can be used as well as isv_LICENSE on case-sensitive file systems. In the case that both are defined, the uppercase ISV_LICENSE will be used.

1.5 License Administration Tools

The RLM server is delivered with an embedded Web Server to perform normal administration tasks. For more information on the web server interface, see Section 1.7.

In addition, the RLM kit is delivered with several command-line administration tools to perform various administration tasks on the license servers as well as to retrieve information about licensing parameters. While the RLM web interface is the preferred method to administer RLM license servers, the command-line tools are provided as a convenience for use in administration scripts and programs. License Administrators can manage RLM by using the administration tools, as described in detail below: Section 1.5.1, Section 1.5.3, Section 1.5.4, Section 1.5.5, Section 1.5.6, Section 1.5.7, Section 1.5.8, Section 1.5.9, Section 1.5.10, Section 1.5.11, Section 1.5.2,.

All utilities (with the exception of **rlmremove**) can be run via **rlmutil**:

```
rlmutil [utility_name]
```

e.g.:

```
rlmutil rlmadduser
```

RLM had the ability to restrict usage of the **remove**, **reread**, **shutdown**, **status** and **option editing** requests via The RLM Options File previously, these are now controlled with user roles in the web interface. For a description of the new roles, see Section 1.7.2. In addition, starting in RLM v13.0, ISV servers can authenticate requests from clients. See Section 1.6 in the next section for more information.

All utilities take the following options:

Option	Meaning
-c license_spec	Use 'license_spec' instead of the current directory to find license files. 'license_spec' can be either a license file or a port@host specification. The -c option overrides RLM_LICENSE.
-dat	Use *.dat as the license file instead of *.lic.
-h	Print usage information and exit.
-q	Don't prompt/quiet (for rlmdown/rlmremove/rlmswitch/rlmhostid). -q also turns off the verification of license checksums and corresponding error messages for all commands.
-v	Print version number and exit.
-z	Password use password as license password for command (enclose in quotes if password contains white space).

1.5.1 rlmadduser

Add a user to the ISV server options file

Usage:

```
rlmutil rlmadduser [options-filename <username> <password>]
```

This command adds the specified username/password pair to the end of the options-file specified.

Note

- RLM Usernames and passwords must be <= 10 characters long
- RLM Usernames and passwords are CASE SENSITIVE
- Usernames and passwords cannot contain white space or any of the following characters: "<", ">", "&", ":", and single ('' or `') or double-quotes (").
- If there are no USER records in the ISV server options file, the server does no authentication of clients.

rlmadduser is new in RLM v13.0. All components must be RLM v13.0 or later for this functionality to work. This functionality is not supported for HTTPS communications in RLM Cloud. See Section 1.6 for more information.

1.5.2 rlmanon

Change user and host names in report log file

Usage:

```
rlmanon [logfile]
```

rlmanon reads the report log file *logfile*, and changes all the user and host names to the form **uNNN** and **hNNN**, where NNN is a sequence number. The result is written into the file *logfile.anon*.

There is a one-to-one mapping from a particular user or host name to its corresponding

sequence number, so that reports generated from the log file will accurately reflect the number of unique users and hosts as well as the sharing of licenses. However, actual user and host information is removed from the output *logfile.anon*.

rlmanon creates a new authenticated report log file if the input log file is an unmodified authenticated log file. If the input file has incorrect authentication records, an error message is generated, and no output file is written.

Note The **rlmanon** cannot be run via **rlmutil**.

1.5.3 rlmdebug

Display debugging information about products

Usage:

```
rlmutil rlmdebug [product]
```

rlmdebug prints information about the specified *product* (or all products if *product* is not specified). The debugging information is written to stdout (this requires running in a command window on Windows systems).

This capability is also built into every RLM v9.0 (or later) application. To use the debugging information directly from the application, set the *RLM_DEBUG* environment variable to the product name (or to an empty string if you wish to debug all products). (Note: you do not need to use the *RLM_DEBUG* environment variable with the **rlmdebug** utility.)

Sample rlmdebug output:

```
% setenv RLM_DEBUG
```

When the application is run the following (sample) output is displayed (in addition to any other output the application may produce):

```
RLM DEBUG for all products
In license file: ../rlm/z.lic (5555@paradise):
Product: test1, ISV: reprise, Floating
Product: test2, ISV: reprise, Floating
Product: test3, ISV: reprise, Floating
Product: rlm_roam, ISV: reprise, Uncounted
Product: testr1, ISV: reprise, Floating
Product: testr2, ISV: reprise, Floating
Checking server machine "paradise" ... server UP
Checking RLM server at port 5555 ... server UP

In license file: a.lic:
Product: test, ISV: reprise, Single

8 product instances found
```

1.5.4 rlmdown

Shuts down the license server

Usage:

```
rlmutil rlmdown [-q] [isv] [-c license_file]
```

rlmdown shuts down the first license server in the license path RLM_LICENSE. If the *-q* option is specified, the shutdown happens without a confirmation prompt. If the optional *isv* is specified, only that ISV's server is shut down.

In order to shut down the RLM server itself, specify the ISV name as **RLM**. (Note: RLM must be in capital letters).

Note for Unix systems

The servers can also be shut down by sending a SIGTERM signal to the RLM process. SIGTERM shuts down all the servers, including RLM.

1.5.5 rlmhostid

Print the hostid of this machine

Usage:

```
rlmutil rlmhostid
[[-]32|disksn|ether|gc|internet|ip|ipv6|uuid|host|user|rlmid1] [-q]
```

rlmhostid prints the hostid of the machine it is running on. If the *-q* flag is specified, the hostid is printed without any other output.

Hostid types described in RLM hostids.

1.5.6 rlmnewlog

Print the hostid of this machine

Usage:

```
rlmutil rlmnewlog <isv> [new-log-file-name]
```

rlmnewlog causes the ISV server isv to move the current reportlog output to new-log-file-name and continue logging to the original filename.

Note The *new-log-file-name* must be on the same filesystem as the original reportlog, otherwise the command will fail. The ISV server renames the old reportlog, it does not copy the data. *rlmnewlog* will fail if the server is not currently writing a report log.

1.5.7 rlmremove

Remove a checked-out license from a user

Usage:

```
rlmutil rlmremove [-q] <server-host> <port> <isv> <handle>
```

rlmremove removes a checked-out license. If the *-q* option is specified, the license is removed without a confirming prompt. *server-host*, *port*, and *handle* are indicated in the *rlmstat* output.

In the following example *rlmstat* output, the *server-host* is **melody**, the *port* is **1215**, and the *handle* is **809f418** and the *isv* is *reprise*:

```
reprise license usage status on melody (port 1215)

test3 v1.000: tom@sun1(v1.0) (809f418) 1/0 at 02/06 09:59
```

1.5.8 rlmreread

Cause the license server(s) to reread their license and option file(s)

Usage:

```
rlmutil rlmreread [isv]
```

rlmreread causes the specified ISV server *isv* to reread its license file, and options file if specified in the license file (or if in the default location *isv*.opt). If *isv* is omitted, the reread command is sent to RLM and all ISV servers. If *isv* is specified as *rlm*, then only the RLM server rereads its license file.

When *rlm* rereads its license file, it starts any new ISV servers that were not present before.

Note RLM performs an automatic reread of the license file(s) every night at midnight.

Unix only

The servers will do a reread if a SIGHUP signal is sent to the RLM process.

1.5.9 rlmstat

Obtains status from the license servers

Usage:

```
rlmutil rlmstat [-a] [-f] [-i [isv] ] [-I] [-l [ isv] ] [-n [host] ] [-p
[product] ] [-u [user] ]
[-z password]
```

rlmstat retrieves status from the license servers and prints it. Control over the status retrieved from *rlmstat* is specified as follows:

Option	Parameter (meaning if present)	Result
-a	(no parameters)	Print all status from RLM and all ISV servers.
-avail	[-i isv] [-p product] [-b]	Reports free license availability (see below).
-f	"full" license listing	Reports more information about the licenses in use.
-i	Display this ISB only	Display license checkout info from ISVs.
-I	Display isv-defined checkout data	"[I: isv-data]" appended to each user line.
-l	Display this *isv* only	Display license pooling info from ISVs.
-n	Display licenses from this *host* only.	Display license checkout info from ISVs.
-p	Display licenses for this *product* only.	Display license checkout info from ISVs.
-u	Display licenses from this *user* only.	Display license checkout info from ISVs.
-z	License password.	Supplies password to license server.

Example rlmstat output:

```
% rlmutil rlmstat -a
rlmstat v9.1
Copyright (C) 2006-2011, Reprise Software, Inc. All rights reserved.

        rlm status on bigserver (port 5053), up 00:03:51
        rlm software version v9.1 (build:3)
        rlm comm version: v1.1
        Startup time: Wed Jul 6 13:27:42 2011
        Todays Statistics (00:03:50), init time: Wed Jul 6 13:27:43 2011
        Recent Statistics (00:03:50), init time: Wed Jul 6 13:27:43 2011

                        Recent Stats    Todays Stats    Total Stats
                        00:03:50         00:03:50         00:03:51
        Messages:       9 (0/sec)        9 (0/sec)        9 (0/sec)
        Connections:    7 (0/sec)        7 (0/sec)        7 (0/sec)

        --------- ISV servers ----------
        Name Port Running Restarts
        reprise 62503 Yes 0
        ------------------------

        reprise ISV server status on bigserver (port 62503), up 00:03:49
        reprise software version v9.1 (build: 3)
        reprise comm version: v1.1
        reprise Debug log filename: <stdout>
        reprise Report log filename: <stdout>
        Startup time: Wed Jul 6 13:27:44 2011
        Todays Statistics (00:03:49), init time: Wed Jul 6 13:27:44 2011
        Recent Statistics (00:03:49), init time: Wed Jul 6 13:27:44 2011

                        Recent Stats    Todays Stats    Total Stats
                        00:03:49        00:03:49        00:03:49
        Messages:       17 (0/sec)      17 (0/sec)      17 (0/sec)
        Connections:    6 (0/sec)       6 (0/sec)       6 (0/sec)
        Checkouts:      2 (0/sec)       2 (0/sec)       2 (0/sec)
        Denials:        0 (0/sec)       0 (0/sec)       0 (0/sec)
        Removals:       0 (0/sec)       0 (0/sec)       0 (0/sec)

        ------------------------

        reprise license pool status on bigserver (port 62503)

        test v1.0
                count: 1, # reservations: 0, inuse: 1, exp: 1-jan-0
                obsolete: 0, min_remove: 20, total checkouts: 1
        test2 v1.0
                count: 1, # reservations: 0, inuse: 1, exp: 1-jan-0
                obsolete: 0, min_remove: 20, total checkouts: 1
        test3 v1.0
                count: 100, # reservations: 0, inuse: 0, exp: 1-jan-0
                obsolete: 0, min_remove: 20, total checkouts: 0

        ------------------------

        reprise license usage status on bigserver (port 62503)

        test v1.0: joe@library 1/0 at 07/06 13:27 (handle: 41)
        test2 v1.0: sam@kitchen 1/0 at 07/06 13:28 (handle: 81)
```

In this output, the first section (before the line of dashed lines ———) is the status of the RLM server, the next section is the status of the ISV server *reprise* (there would actually be one section of status for each ISV server if there were more than one running). Next comes the license pool

info for each ISV server (again, only one section for the *reprise* server), followed by the actual license usage information.

Also, please note that the expiration date shown in this output is the expiration date of the **first license to expire** out of all the licenses used to create the license pool in the license server. When more than one license is used to create a single license pool (licenses are combined when all relevant parameters of 2 different licenses match), then only the **earliest expiration date** is shown. The other license(s) may have any expiration date that has not yet expired. To determine the expiration date of all licenses used to make up a license pool the actual license file must be consulted. Also note that licenses from different license files could be combined to make a single license pool.

Beginning in RLM v14.2, the first line of the product information ("test v1.0" in the example above), will contain the server's license pool #, and an ID, if specified. So rather than

```
test v1.0
```

it will be something like this:

```
test v1.0, pool: 7, id: 60
```

The meaning of the license usage line:

```
test v1.0: joe@library 1/0 at 07/06 13:27 (handle: 41)
```

is as follows:

- **test** is the product name
- **v1.0** is the license version (from the license file) for test
- **joe** is the user who is using the license
- **library** is the host on which tom is using the license
- **(41)** is the license handle in the server. This handle is used by the rlmremove command.
- **1/0** indicates that 1 unreserved and 0 reserved licenses are in use
- **at 07/06 13:28** is the time the licenses were checked out

And if the optional "-I" switch is specified, the following will be appended to the line, if there is any isv-defined checkout data:

```
[I: isv-data]
```

where "isv_data" is the isv-defined checkout data.

If the -f option is used with rlmstat, additional information about the license usage will be displayed. -f is available in RLM v12.5, and in v12.5, the additional information displayed is the requested version, displayed as such:

```
test v1.0 (req: v0.8): joe@library 1/0 at 07/06 13:27 (handle: 41)
```

rlmstat -avail command

The *rlmstat -avail* command reports on license availability for a specified license, a specified ISV, or all licenses from all ISVs.

Usage:

```
rlmutil rlmstat -avail [-i isv] [-p product] [-b]
```

If *-i isv* is specified, only licenses from the selected ISV are displayed. If *-p product* is specified, only the selected product is displayed. If *-b* is specified, license availability is combined across license servers.

If you are looking for the availability of a license from a particular ISV, it is more efficient to

specify the ISV name in the command. If you do not specify the ISV name, rlmstat must contact the RLM server to request a list of ISV servers, then request information from each ISV server. If you specify the ISV, then only that ISV server is contacted.

Note also that there are situations when you may not be able to check out a license that is listed as available. This can happen if, for example, you are on the EXCLUDE list for a particular product, not on the INCLUDE list, already exceeded your MAX usage, etc.

Conversely, you might be able to check out one that is listed as not available. This could happen if that license is shared and you can share an existing checked-out license, or if one of the reservations for the license is for you (rlmstat -avail lists free available licenses; reservations are not generally available).

Example rlmstat -avail output

```
% rlmutil rlmstat -avail -i reprise
rlmstat v15.1
Copyright (C) 2006-2011, Reprise Software, Inc. All rights reserved.

License availability for all products from ISV "reprise"

        server host: telecard (port 5053)
                test1 v1.000 available: 15
                test1 v1.000 hostid: a8c00301 available: 10
                test5 v3.000 hostid: a8c00301 available: 2
                test5 v3.000 available: 10
                test v1.000 available: 10

        server host: spinout (port 5053)
                test1 v1.000 available: 15
                test1 v1.000 hostid: a8c00301 available: 10
                test5 v4.000 hostid: ip=172.16.7.28 available: unlimited
                test5 v2.300 available: 15
                test5 v3.000 hostid: a8c00301 available: unlimited
                test5 v3.000 available: 73
```

Example rlmstat -avail -b output (same situation as above)

```
% rlmutil rlmstat -avail -i reprise -b
rlmstat v15.1
Copyright (C) 2006-2011, Reprise Software, Inc. All rights reserved.

License availability for all products from ISV "reprise"

        ISV: reprise
                test1 v1.000 available: 30
                test1 v1.000 hostid: a8c00301 available: 20
                test5 v3.000 hostid: a8c00301 available: unlimited
                test5 v3.000 available: 83
                test5 v4.000 hostid: ip=172.16.7.28 available: unlimited
                test5 v2.300 available: 15
```

1.5.10 rlmswitch

Switches the debug log info to a new file

Usage:

```
rlmutil rlmswitch [isv] <new-log-file-name>
```

rlmswitch causes the server *isv* to close the current debug log file and begin output to *new-log-filename*. If *isv* is not specified, or if specified as *rlm*, the RLM server's debug log is

switched.

1.5.11 rlmswitchr

Switches the report log info to a new file

Usage:

```
rlmutil rlmswitchr [isv] <new-log-file-name>
```

rlmswitchr causes the ISV server *isv* to close the current reportlog file and begin output to *new-log-file-name*.

This command will fail if the server is not currently writing a report log.

1.6 Client Authentication

Beginning in RLM v13.0, ISV servers are able to authenticate clients before doing any processing on their requests. This ability is entirely under your control, based on information in the ISV server options file. If there are any USER records in the options file, the server will authenticate all requests (including checkouts, status, reread, shutdown, etc.).

Note This functionality is available only with RLM native comms, not with HTTPS communications.

1.6.1 Turning on Authentication

In order to turn on authentication, you must create one or more USER records in the Section 1.9. This is done with the new rlmadduser utility (or "rlmutil rlmadduser").

To add a user, run:

```
% rlmadduser options-file-name username password
```

This command adds the specified username/password pair to the end of the options-file specified.

Some notes:

- Usernames and passwords must be <= 10 characters long
- Usernames and passwords are CASE SENSITIVE
- Usernames and passwords cannot contain white space or any of the following characters: "<", ">", "&", ":", and single (' or `) or double quotes (")
- If there are no USER records in the ISV server options file, the server does no authentication of clients.

1.6.2 Passing authentication to the Server

In order to pass the authentication data to the license server, set the environment variable

RLMAUTH to <username>:<password>. The data will be passed on any client request to the server.

This capability requires an RLM v13.0-linked client as well as v13.0 license servers.

1.6.3 Server Processing

If the ISV server encounters any *valid* USER records in its options file, authentication is automatically turned on. If the same username is encountered twice in the options file, the **password from the last entry** will be used. All components must be RLM v13.0 or greater, both servers and applications.

If you use authentication on more than one ISV server, you will probably find it useful to have a single administration user to do shutdowns and rereads.

1.7 The RLM Web Server

The RLM server contains an embedded *Web Server* which can be used to perform most administration of the RLM server itself. The web server contains the functionality of all the rlmutil-based utilities except rlmhostid. The web server allows you to retrieve server and license status (similar to *rlmstat*), cause the servers to re-read the license files (*rlmreread*), switch debug (*rlmswitch*) or report log (*rlmswitchr*) files, move the current report log file to a new name (*rlmnewlog*), or shut down the license servers (*rlmdown*). Using this web-based interface, you can administer the license servers from any platform, and you do not need to install the RLM utilities - you only need a web browser.

Important Beginning in RLM v16.0, HTTPS is now used by default (HTTPS support was first added in v15.1). See Section 1.7.1 for more details.

In addition, the web server allows you to view ISV option files (with *Manage* permissions; from the action menu **?** of each ISV server), and the global RLM option file (as an *Admin*; from the Settings menu). Also, the web interface allows you to view the recent debug log information from any of the servers (via the action menu **?**) if you have *Manage* permissions. The **reread** and **shutdown** commands are possible with the *Manage* permission, while the *View* user can see the status and usage for running ISV servers. See: Section 1.7.2 for more on access permissions.

The web server is started automatically on port 5054 when RLM is started. To use the web server, simply point your browser to: https://ServerHostName:5054 and select the operation you would like to perform. You will be prompted for any required information.

Note For earlier version of RLM (15.2 and below) you may need to use http://ServerHostName:5054 to access the management interface.

If you would like to run the web server on a different port, specify the *-ws NNNNN* command-line argument when starting RLM, where *NNNNN* is the desired port.

The RLM web server is 100% self-contained in the RLM binary; no additional html files are required for operation.

> **Warning** In RLM v14.1, RLM v14.2 and RLM v15.0 the web server is disabled if the user running RLM is root or an admin user.

The remaining sections will describe some of the main functions of the web interface.

1.7.1 Enabling HTTPS in the RLM Web Server

Starting in version 16.0, HTTPS is enabled by default and will generate a self-signed certificate (rlm-cert.pem) and key (rlm-key.pem) on startup, if they do not exist. To use your own certificate and key, use the two startup options, *-sslcert* and *-sslpriv* to point to your SSL certificate and SSL private key.

e.g.:

```
% ./rlm -sslcert /path/to/cert.pem -sslpriv /path/to/privkey.pem
```

Tip If you rename your certificate "rlm-cert.pem" and your key "rlm-key.pem", and overwrite the self-signed copies in the working directory, RLM will pick them up automatically.

You can access your web server by typing https://ServerHostName:5054 (be sure to include the https:// if you are not automatically redirecting the URL via other means).

If you are running your server as root/administrator, it is possible to set your web server to run on port 443 (the default HTTPS port) by running the following:

```
% ./rlm -sslcert /path/to/cert.pem -sslpriv /path/to/privkey.pem -ws 443
```

While running on port 443 and pointing at a valid SSL certificate and private key, your browser should automatically redirect to https:// without needing to type it explicitly. (i.e., typing ServerHostName should take you to https://ServerHostName:443.

Warning Due to the self-signed nature of the automatically generated certificate, your browser will likely display a warning before you can get to the web server. **Traffic to the web server will still be encrypted**.

Self-signed certificates can not be generated with a password to use the web server.

1.7.2 Access Control to the RLM Web Interface

Note Earlier versions of RLM used the rlm.pw file to control access. For access control in version 15.2 and earlier, please see the legacy documentation

Starting in version 16.0, logins are now required for access to the web interface. The default login is:

Username:	admin
Password:	admin

On first login you will be prompted to set a new password.

Permission Levels

In version 16.0 there are three permission levels for web interface users:

Permission Level	Description
View	Can view the status of all ISV servers, products available on each ISV server, as well as usage information.

Manage	All *View* user permissions. *Manage* adds control over ISV servers (reread-/restart, shutdown, view options and logs), checkout removal, can add and remove named users, license activation, and the ability to view and download diagnostics.
Admin	All *View* and *Manage* permissions. Adds access to the Settings menu, which includes user creation, modification, and permission level control.

Managing Users

Add:

1. Login to the web interface as a user with *Admin* permissions.
2. Click the menu icon  in the top right corner and select **Settings**.
3. From the **User Management** tab, enter the name of the new user and select the permission level, then click **Create User**.
4. You will be presented with a dialogue with the user's temporary password. They will be prompted to set a new password on first login.

Remove:

1. Login to the web interface as a user with *Admin* permissions.
2. Click the menu icon  in the top right corner and select **Settings**.
3. From the **User Management** tab, locate the user you want to remove and click the trash  icon on the right.

Change permissions:

1. From **User Management** locate the user who needs a password reset.
2. Click the kebab menu **?** on the right and select **Edit Permissions**.
3. From the dropdown menu, select the new permission level, and click **Update Role**.

Password reset:

1. From **User Management** locate the user who needs a password reset.
2. Click the kebab menu **?** on the right and select **Reset Password**.
3. You will be given a temporary password to provide to the user. They will be prompted to set a new password on first login.

The RLM password file

Beginning in version 16.0, the password file (rlm.pw) no longer controls and web server functionality and may be removed.

1.7.3 License Server Status

After logging in you will be presented with the Home/License Server Status page. From here you can see all license servers currently running on RLM, the port each server is running on, and the server start time. You will also see the hostname of the server RLM is running on, as well as the RLM server port.

Click a server to expand it and view all products currently licensed on that server. Click

a product to expand it and **view usage** for that product.

Note The expiration date shown in this table is the expiration date of the first license to expire out of all the licenses used to create the license pool. When more than one license is used to create a single license pool (licenses are combined when all relevant parameters of 2 different licenses match), then only the earliest expiration date is shown. The other license(s) may have any expiration date that has not yet expired. To determine the expiration date of all licenses used to make up a license pool the actual license file must be consulted. Also note that licenses from different license files could be combined to make a single license pool.

If you are a *Manage* or *Admin* user, you will see the **Reread/Restart All Servers** and **Shutdown All Servers** server actions at the top of the page.

1.7.4 ISV Servers

In version 16.0 of RLM, ISV server information is all available directly from the home page. When a server is collapsed, you will see the port the ISV server is running on and the start time. The colored bar on the left side indicates at a glance whether the server is running normally (green), has a license issue (orange), or is not currently running (red).

ISV Server Actions

Manage and *Admin* users will also see the kebab **?** menu on the right side of each ISV server. This opens the individual ISV server actions. From here you can:

Command	Description
View / Edit License(s)	Allows you to view and edit any license files for that ISV.
Reread / Restart Server	Rereads and restarts the ISV server. Use if you have added a license or made other changes relevant to this specific ISV.
Shutdown Server	Shuts down the ISV server.
View ISV Options	This allows you to view the ISV options file. Any changes must be made directly to the options file on the server running RLM.
View Debug Log	Allows you to view the last 20 lines of the ISV debug log.
Switch Reportlog	Enter the filename for the new reportlog. After clicking **Yes**, RLM will begin writing the reportlog to this new file.
New Reportlog	Enter a new filename for the reportlog. After clicking **Yes**, RLM will move the contents of the reportlog to this new file, and continue logging to the original reportlog.
Switch Debuglog	Enter the new filename for the debuglog. After clicking **Yes**, RLM will begin writing the debuglog output to this new file.
License Transfer	Coming in v16.1
Alternate Server Hostid	Coming in v16.1

Note On Unix systems, all servers (RLM plus all ISV servers) initially write their debug log to the same file (stdout of the RLM process). Once you switch any server to a different file, it is not possible to combine the debug log output again.

1.7.5 Maintaining Named User Licenses

If a product uses a named user license, you will see an **Edit** button on the right-hand side of the product row. Click this to open the named user dialogue.

The **Add Group** button allows you to add any groups specified in the ISV options file (see Section 1.9.11). This will add users from the specified group until or unless the named user list is full.

Clicking **Add User** will allow you to enter the name of a specific user to be added to the list.

To remove a user, simply click the **Remove** button next to the user's name. The **Recently Removed** tab shows all users that have been recently removed, and the minimum amount of time before that user can be added back to the list.

Note The user cannot be removed from the list if he/she currently has any licenses checked out at the server (including roaming licenses).

1.7.6 Activate

When you click the **Activate** tab at the top of the page you will be presented with fields to enter the information necessary to activate a license from an ISV server's internet site. Once you have entered all of the required information, click **Activate License** to generate the license file. This may also be used to generate license files for other machines if, for example, they do not have internet access. Doing so will require the hostid information for the other machine(s).

You can control the appearance and defaults of the "Activate License" command with the RLM "-activate" options in the RLM options file. See: Section 1.8.3.

1.7.7 Diagnostics

Click the **Diagnostics** tab to view the diagnostic information for the RLM server. This page is very useful for troubleshooting any issues you may be experiencing with the license manager. In addition to all diagnostics information being available in the browser, you have the option to **Download** the diagnostics to your local machine if you need to send this in to your software support team for troubleshooting.

1.7.8 RLM System Info

If you select System Info from the profile menu  on the top right, you will see the RLM system information screen. This information contains the platform type and hostid information for the system where the RLM process is running (NOTE: not where your browser is running). This screen will also display a list of all RLM processes running on this computer (including processes which are not currently running but which have run in the prior 24 hours).

1.8 The RLM Options File

The RLM options file allows control over access to the *status*, *reread*, *shutdown* administration commands as well as control over the editing of options files. Options are provided to either allow (INCLUDE or INCLUDEALL) or disallow (EXCLUDE or EXCLUDEALL) administration command usage. Additionally, options are provided to create groups of users (GROUP) or hosts (HOST_GROUP) or IP addresses (INTERNET_GROUP).

In addition, the RLM options file allows you to turn off logging of status requests (to the debug log) via the NOLOG option.

Finally, the RLM options file allows you to set the defaults for the "Activate License" command in the menu.

The RLM options file is called **rlm.opt**, and should be placed in the directory from which you run the RLM (or rlm.exe) binary.

If you would like to add comments to the options file, start the line with the '#' character.

There are 8 privileges which can be controlled in the RLM options file. Each privilege is specified with the appropriate privilege name in the RLM options file. Note that these privilege names are the same names that are used in the *RLM password file* if you are controlling access to the RLM web interface via user login. If you use the RLM password file, you should not use these lines in the RLM options file - in other words, you should use one mechanism or the other, but not both.

1.8.1 RLM privileges controlled by the RLM options file

Privilege Name	Meaning
edit_meter	Allows modifying count for meter counters.
edit_xfer	Allows editing server-server license transfer settings for ISV servers.
logfiles	Enables the functions which change log files - switch, switchr, newlog.
edit_options	*Removed in v16.0. Options now read-only.*
edit_rlm_options	*Removed in v16.0. Options now read-only.*
remove	*Removed in v16.0. Now restricted to Manage and Admin roles.*
reread	*Removed in v16.0. Now restricted to Manage and Admin roles.*
shutdown	*Removed in v16.0. Now restricted to Manage and Admin roles.*
status	*Removed in v16.0. Now restricted to Manage and Admin roles.*

The RLM options file syntax is a subset of Section 1.9 syntax. The **privilege** names listed in the table above are used, where as in the ISV options file a **product** name would be used. By default, all privileges are granted to all users unless otherwise restricted in the RLM options file.

1.8.2 Legal characters in the RLM options file

In general, all options file fields are white-space delimited, meaning that no data item can contain embedded spaces, tabs, newlines or carriage returns. In addition, the following four characters are illegal in data items in the ISV or RLM options (and license) file: "<", ">", "&", and double quote (").

The characters "<", ">", and "&" are legal in comments (lines beginning with "#") but nowhere else.

> **Warning** All lines in option files (RLM or ISV) as well as license files must be shorter than 1024 characters. Anything over 1024 characters will be truncated.

Note Everything in the RLM options file is case-insensitive.

In the following example RLM options file, status commands are only allowed from hosts on subnet 172.16.7.*, no one on host "excluded_host" can do a reread command, and only users on IP address 172.16.7.93 can do a shutdown. Each command (INCLUDE, EXCLUDE, etc) must be on a separate line. Also, RLM will not process reread or shutdown requests from pre-v9 command-line utilities.

```
NO_OLD_RLMUTIL
INCLUDE status internet 172.16.7.*
EXCLUDE reread host excluded_host
INCLUDE shutdown internet 172.16.7.93
```

For a detailed description of each option, see the section below.

Note *privilege* should be **one** of the privilege names from the table above.

1.8.3 ACTIVATE

```
ACTIVATE [ url URL | isv ISVNAME]
```

The ACTIVATE line allows you to set the defaults for the URL and ISV name for activation.

The 2 forms of the ACTIVATE line are:

- ACTIVATE url URL
- ACTIVATE isv ISVNAME

In the first form "URL" is the default URL used for activation. For example:

```
ACTIVATE url www.reprisesoftware.com
```

In the second form "ISVNAME" is the default ISV name used for activation. For example:

```
ACTIVATE isv reprise
```

Note In v15.2 and earlier a third form was available (ACTIVATE off). Activation is now restricted to user with Manage or Admin roles.

1.8.4 ACTIVATE_TEST

Note This option is not currently available in v16.0.

```
ACTIVATE_TEST <url> <isv> [activation_key [hostid [count]]]
```

The ACTIVATE_TEST line allows you to specify the parameters for the "Test License Activation" button in the GUI.

You must specify the URL and the ISV on this line. The defaults for the other parameters are:

activation_key	"test"
hostid	"12345678"
count	1

Example:

```
ACTIVATE_TEST activate.yourwebsite.com yourISVName 1234-1234-1234-1234
ab12cd34 5
```

No special privileges are required to use the "Test License Activation" button.

ACTIVATE_TEST is new in RLM v15.0

1.8.5 EXCLUDE

```
EXCLUDE <privilege> [user|host|group|host_group|internet|project] <who>
[id=nnn]
```

The EXCLUDE line removes the specified *privilege* from a particular user, host, group, host_-group, IP address, or project. If you specify group or host_group, it must be defined by a GROUP or HOST_GROUP line in the RLM options file.

Portions of the INTERNET address can be specified with a '*' (wildcard) which matches any address, e.g., 172.16.7.*

If specified, the *id* applies this option to the license with an *id* of "*nnn*".

Example excluding the user group "engineers" (see GROUP below) from editing meter:

```
EXCLUDE edit_meter group engineers
```

Note To use PROJECT the user needs to specify the project name in the environment variable RLM_PROJECT prior to starting RLM. To use INTERNET specify the internal (vs. external) IP address of the client machine.

For a list of the privileges available, see Section 1.8.1 above.

1.8.6 EXCLUDEALL

```
EXCLUDEALL [user|host|group|host_group|internet|project] <who>
```

The EXCLUDEALL line prevents usage of all capabilities defined by all privileges by a particular user, host, group, host_group, IP address, or project. If you specify group or host_group, it must be defined by a Section 1.8.7 or Section 1.8.8 line in the RLM options file.

Portions of the INTERNET address can be specified with a '*' (wildcard) which matches any address, e.g., 172.16.7.*

Example exluding the user "mike" from all privileges:

```
EXCLUDEALL user mike
```

Note To use PROJECT the user needs to specify the project name in the environment variable RLM_PROJECT prior to starting RLM. To use INTERNET specify the internal (vs. external) IP address of the client machine.

For a list of the privileges available, see Section 1.8.1 above.

1.8.7 GROUP

```
GROUP <name> <list-of-usernames>
```

The GROUP line defines a group of users to be used in an EXCLUDE, EXCLUDEALL, INCLUDE, or INCLUDEALL line. Separate the usernames in the list by spaces. Multiple lines that specify the same GROUP name will have their lists of usernames concatenated.

Example for a group named "engineers" with 3 users:

```
GROUP engineers tom dick harry
```

This example results in a group with 6 users:

```
GROUP engineers tom dick harry
GROUP engineers larry curly moe
```

1.8.8 HOST_GROUP

```
HOST_GROUP <name> <list-of-hostnames>
```

The HOST_GROUP line defines a group of hosts to be used in an EXCLUDE, EXCLUDEALL, INCLUDE or INCLUDEALL line. Separate the hostnames in the list by spaces. Multiple lines that specify the same HOST_GROUP name will have their lists of hostnames concatenated.

Example for a group named "corporate" with 3 hosts:

```
HOST_GROUP corporate node_a node_b node_c
```

This example results in a group of 6 hosts:

```
HOST_GROUP corporate node_a node_b node_c
HOST_GROUP corporate node_d node_e node_f
```

1.8.9 INTERNET_GROUP

```
INTERNET_GROUP <name> <list-of-ip-addresses>
```

The INTERNET_GROUP line defines a group of IP addresses to be used in an EXCLUDE, EXCLUDEALL, INCLUDE, INCLUDEALL, MAX or RESERVE line. Separate the IP addresses in the list by spaces.

Multiple lines that specify the same INTERNET_GROUP name will have their lists of IP addresses concatenated.

IP addresses can contain the wildcard ('*') character.

Example:

```
INTERNET_GROUP corporate 192.168.1.4 2.*.*.7 172.16.7.*
```

This example results in a group of 6 IP addresses:

```
INTERNET_GROUP corporate 192.168.17.2 192.168.17.2 192.168.17.2
INTERNET_GROUP corporate 172.16.7.4 172.16.7.5 172.16.7.6
```

1.8.10 INCLUDE

```
INCLUDE <privilege> [user|host|group|host_group|internet|project] <who>
[id=nnn]
```

The INCLUDE line grants the specified privilege to a particular user, host, group, host_group, IP address, or project. If you specify group or host_group, it must be defined by a GROUP or HOST_GROUP line in the RLM options file. Anyone not specified by the INCLUDE line is not

allowed access to the capabilities defined by privilege.

If specified, the *id* applies this option to the license with an *id* of "nnn".

Portions of the INTERNET address can be specified with a '*' which matches any address, e.g., 172.16.7.*

Example granting the *status* privilege to the host group "engineers":

```
INCLUDE status host_group corporate
```

Note To use PROJECT the user needs to specify the project name in the environment variable RLM_PROJECT prior to starting RLM. To use INTERNET specify the internal (vs. external) IP address of the client machine.

For a list of the privileges available, see Section 1.8.1 above.

1.8.11 INCLUDEALL

```
INCLUDEALL [user|host|group|host_group|internet|project] <who>
```

The INCLUDEALL line grants all privileges to a particular user, host, group, host_group, IP address, or project. If you specify group or host_group, it must be defined by a GROUP or HOST_GROUP line in the RLM options file. Anyone not on the INCLUDEALL list is not allowed to use a capability controlled by any privilege.

Portions of the INTERNET address can be specified with a '*' which matches any address, e.g., 172.16.7.*

Example granting all privileges to the user group "admins":

```
INCLUDEALL group admins
```

Note To use PROJECT the user needs to specify the project name in the environment variable RLM_PROJECT prior to starting RLM. To use INTERNET specify the internal (vs. external) IP address of the client machine.

For a list of the privileges available, see Section 1.8.1 above.

1.8.12 NO_OLD_RLMUTIL

The NO_OLD_RLMUTIL line prevents pre-RLM-v9 command-line utilities from performing a reread, remove, or shutdown operation. The pre-v4.0 RLM utilities do not respect the RLM permissions for the reread or shutdown commands, and the pre-v9.0 utilities do not respect the permissions for the remove command.

Adding NO_OLD_RLMUTIL to your ISV options file will prevent these older utilities from performing these commands, and only a v9 (or newer) RLM command-line utility can be used for this purpose.

By default, all operations can be performed by all versions of the RLM command-line utilities. In order for NO_OLD_RLMUTIL to be effective, **it must be specified in both the RLM and the ISV server options files**.

Example:

```
NO_OLD_RLMUTIL
```

1.8.13 NOLOG

NOLOG status

The NOLOG option instructs the RLM server to omit logging of status requests to the debug log.

Example:

```
NOLOG status
```

This example causes the RLM server to omit the logging of status requests in the debug log.

1.9 The ISV Options File

The ISV options file allows control of the use of licenses by various users and groups of users within your organization. Options are provided to reserve licenses (Section 1.9.24), and to either allow (Section 1.9.14 or Section 1.9.15) or disallow (Section 1.9.5 or Section 1.9.6) license use. Additionally, options are provided to create groups of users (Section 1.9.11) or hosts (Section 1.9.12), or IP addresses (Section 1.9.13), and to control the maximum number of licenses a user or group can check out (Section 1.9.17). Certain users/hosts/groups, etc. can be exempted from the MAX settings with the Section 1.9.17 and Section 1.9.8 lines.

Logging of the license servers is controlled by the (Section 1.9.4) and (Section 1.9.23) options. In addition, you can suppress debug logging of check-in/checkout/denied entries with the Section 1.9.20 option. Automatic report log file rotation is supported in ISV servers via the Section 1.9.27 option and automatic purging of old report logs is supported with the :ref:purge-reportlog` option (new in v12.1).

Control over license roaming is provided with the Section 1.9.16, Section 1.9.7, Section 1.9.26, and Section 1.9.25 options.

License timeout is controlled via the Section 1.9.28 and Section 1.9.29 options.

License timezone restrictions are controlled with the Section 1.9.30 option.

Client license caching is controlled with the Section 1.9.3 option.

License queuing behavior can be modified with the Section 1.9.10 and Section 1.9.21 options.

There is one additional option available in the ISV options file: Section 1.9.19. This option goes on a line by itself, with no parameters. If specified, the RLM command-line utilities prior to RLM v9.0 will not be able to perform an rlmdown, rlmreread, or rlmremove on this server. By default, all versions of the RLM utilities are enabled unless NO_OLD_RLMUTIL is specified in both the RLM and the ISV options files.

Note that, in RLM v15.0, the server will add the client's external IP address to the list of IP addresses supplied by the client. This IP address is not added if it is the same as one of the client supplied IP addresses. This means that if your license server is on the internet, you can use the client's external IP address to INCLUDE/EXCLUDE license checkouts.

1.9.1 How The ISV Options File is located

The ISV options file can be located in 3 ways:

- You can specify the ISV options file location on the ISV line in The License File.
- If no specification is on the ISV line, RLM will look for <ISV>.opt (where <ISV> is the name of the ISV) in the location with the first license file.
- If there is no options file in either of the first 2 locations, RLM will look for <ISV>.opt in

the working directory where you started the RLM server.

1.9.2 Legal characters in the ISV options file

In general, all options file fields are white-space delimited, meaning that no data item can contain embedded spaces, tabs, newlines or carriage returns. In addition, the following four characters are illegal in data items in the ISV or RLM options (and license) file: "<", ">", "&", and double quote (").

The characters "<", ">", and "&" are legal in comments (lines beginning with "#") but nowhere else.

Warning All lines in option files (RLM or ISV) as well as license files must be shorter than 1024 characters. Anything over 1024 characters will be truncated.

To add comments to the options file, start the line with the '#' character.

Note Everything in the ISV options file is case-insensitive with the exception of the file pathname in the DEBUGLOG and REPORTLOG lines (case-sensitive on Unix systems only).

For a detailed description of each option, see below.

1.9.3 CLIENT_CACHE

```
CLIENT_CACHE <secs> <product>
```

The CLIENT_CACHE line sets the cache time for a license to *secs* seconds for *product*. If the license for *product* does not have a "client_cache" specification, the CLIENT_CACHE option as no effect.

The cache value *secs* can be set to between 0 and 2 times the value in the license's *client_cache* parameter. Attempting to set it to more than 2 times the license parameter will result in a log line similar to the following:

```
07/17 10:40 (reprise) foo: CLIENT_CACHE value (200) > 2x license value (120)
, 120 used.
```

If *product* is not specified, the cache value applies to all products.

Example 1:

To disable client caching for all licenses:

```
CLIENT_CACHE 0
```

This is logged:

```
07/17 10:40 (reprise) Setting CLIENT_CACHE for all products to 0 secs.
```

Example 2:

Here we try to set the client cache to 200 seconds for *product* "foo". In this example, "foo" had a client_cache value of 60 in the license, so it was limited to 120:

```
CLIENT_CACHE 200 foo
```

This is logged:

```
07/17 10:40 (reprise) Setting CLIENT_CACHE for foo to 200 secs.
07/17 10:40 (reprise) foo: CLIENT_CACHE value (200) > 2x license value (120),
120 used.
```

1.9.4 DEBUGLOG

```
DEBUGLOG [+]/ <file_path>
```

The DEBUGLOG option instructs the ISV server to write a debug log to the filename file_path. If file_path is preceded with a '+' sign, the new data is appended to the file, otherwise the file is overwritten. This may be useful if you have many ISV servers and want to isolate the debug output from one server in a separate file.

When running the RLM server as a Windows Service:

If no DEBUGLOG is specified in the ISV options file, RLM will write the ISV debug log in:

<location of rlm.exe><isv>.dlog

This file will be overwritten every time the ISV server starts, since there is no opportunity to specify that the file should be appended to in the default case. In fact, the ISV server logs a few lines to this file at startup time even if a DEBUGLOG is specified in the ISV options file. It is overwritten every time the ISV server starts, but its contents don't change from startup to startup, so nothing important is lost.

Reprise Software Inc. recommends that the debug log path be specified in the ISV options file, and that the append behavior be enabled with '+'<path>. However, it is important not to specify the debug log name as <isv>.dlog, as this specific file is overwritten at each startup.

Example:

```
DEBUGLOG +C:\rlm\logName.dlog
```

1.9.5 EXCLUDE

```
EXCLUDE <product>
[user|host|group|host_group|internet|internet_group|project] <who>
```

or

```
EXCLUDE <product> noproject [id=nnn]
```

The EXCLUDE line prevents usage of a product by a particular user, host, group, host_group, IP address, or project. If you specify group, host_group, or internet_group, it must be defined by a GROUP or HOST_GROUP or INTERNET_GROUP line in the options file.

Alternatively, with **noproject** specified, checkout requests for product from users who do not have the RLM_PROJECT environment variable set will be rejected.

Portions of the INTERNET address can be specified with a '*' which matches any address, e.g., 172.16.7.* Note that INTERNET and INTERNET_GROUP are ineffective on RLM Cloud servers where the application is using HTTPS transport.

In all cases, the status returned to the user will be RLM_EL_ON_EXC - "User/Host on exclude list".

If specified, the id applies this option to the license with an id of "nnn".

Note To use PROJECT the user needs to specify the project name in the environment variable RLM_PROJECT prior to starting the application. To use INTERNET, specify the internal (vs. external) IP address of the client machine.

1.9.6 EXCLUDEALL

```
EXCLUDEALL [user|host|group|host_group|internet|internet_group|project] <who>
```

or

```
EXCLUDEALL noproject
```

The EXCLUDEALL line prevents usage of all products by a particular user, host, group, host_group, IP address, or project. If you specify group, host_group, or internet_group, it must be defined by a GROUP or HOST_GROUP or INTERNET_GROUP line in the options file.

Alternately, with **noproject** specified, all checkout requests from users who do not have the RLM_PROJECT environment variable set will be rejected.

Portions of the INTERNET address can be specified with a '*' which matches any address, e.g., 172.16.7.* Note that INTERNET and INTERNET_GROUP are ineffective on RLM Cloud servers where the application is using HTTPS transport.

In all cases, the status returned to the user will be RLM_EL_ON_EXC_ALL - "User/Host on excludeall list".

Note To use PROJECT the user needs to specify the project name in the environment variable RLM_PROJECT prior to starting the application. To use INTERNET, specify the internal (vs. external) IP address of the client machine.

1.9.7 EXCLUDEALL_ROAM

```
EXCLUDEALL_ROAM [user|host|group|host_group|internet|internet_group|project]
<who>
```

The EXCLUDEALL_ROAM line prevents usage of roaming by a particular user, host, group, host_group, IP address, or project. If you specify group, host_group, or internet_group, it must be defined by a GROUP or HOST_GROUP or INTERNET_GROUP line in the options file. Anyone on the EXCLUDEALL_ROAM list is not allowed to set up roaming licenses.

Portions of the INTERNET address can be specified with a '*' which matches any address, e.g. 172.16.7.* Note that INTERNET and INTERNET_GROUP are ineffective on RLM Cloud servers where the application is using HTTPS transport.

Note To use PROJECT the user needs to specify the project name in the environment variable RLM_PROJECT prior to starting the application. To use INTERNET, specify the internal (vs. external) IP address of the client machine.

1.9.8 EXEMPTALL_MAX

```
EXEMPTALL_MAX [user|host|group|host_group|internet|internet_group|project]
```

```
<who>
```

The EXEMPTALL_MAX line "exempts" the specified user/host, etc. from the MAX settings for all products.

User/host/group/host_group/internet_group/project are all as specified in INCLUDE/EXCLUDE, etc.

The EXEMPTALL_MAX line was added in RLM v15.0.

1.9.9 EXEMPT_MAX

```
EXEMPT_MAX <product>
[user|host|group|host_group|internet|internet_group|project] <who>
```

The EXEMPT_MAX line "exempts" the specified user/host, etc. from the MAX settings for product product.

User/host/group/host_group/internet_group/project are all as specified in INCLUDE/EXCLUDE, etc.

The EXEMPT_MAX line was added in RLM v15.0.

1.9.10 EXPRESS

```
EXPRESS [ON|OFF] <product>
```

The EXPRESS line controls queuing behavior for one or all products. If EXPRESS is turned on for a product, then queued requests which can be satisfied immediately are granted independent of whether other requests are ahead in the queue. If EXPRESS is turned off, then a queued request will always be placed at the end of the queue if it exists.

If the EXPRESS line does not specify a product, it applies to all products.

By default, EXPRESS is turned ON for all products in RLM.

1.9.11 GROUP

```
GROUP <name> <list-of-usernames>
```

The GROUP line defines a group of users to be used in an EXCLUDE, EXCLUDEALL, INCLUDE, INCLUDEALL, MAX or RESERVE line. Separate the usernames in the list by spaces.

Multiple lines that specify the same GROUP name will have their lists of usernames concatenated.

Example:

```
GROUP engineers tom dick harry
```

This example results in a group of 6 users:

```
GROUP engineers tom dick harry
GROUP engineers larry curly moe
```

1.9.12 HOST_GROUP

```
HOST_GROUP <name> <list-of-hostnames>
```

The HOST_GROUP line defines a group of hostnames to be used in an EXCLUDE, EXCLUDEALL, INCLUDE, INCLUDEALL, MAX or RESERVE line. Separate the hostnames in the list by spaces.

Multiple lines that specify the same HOST_GROUP name will have their lists of hostnames concatenated.

Example:

```
HOST_GROUP corporate node_a node_b node_c
```

This example results in a group of 6 hosts:

```
HOST_GROUP corporate node_a node_b node_c
HOST_GROUP corporate node_d node_e node_f
```

1.9.13 INTERNET_GROUP

```
INTERNET_GROUP <name> <list-of-ip-addresses>
```

The INTERNET_GROUP line defines a group of IP addresses to be used in an EXCLUDE, EXCLUDEALL, INCLUDE, INCLUDEALL, MAX or RESERVE line. Separate the ip addresses in the list by spaces.

Multiple lines that specify the same INTERNET_GROUP name will have their lists of ip addresses concatenated.

IP addresses can contain the wildcard ('*') character.

Example:

```
INTERNET_GROUP corporate 1.2.3.4 2.*.*.7 172.16.7.*
```

This example results in a group of 6 IP addresses:

```
INTERNET_GROUP corporate 1.1.1.1 2.2.2.2 3.3.3.3
INTERNET_GROUP corporate 4.4.4.4 5.5.5.5 6.6.6.6
```

Note that INTERNET_GROUP is ineffective on RLM Cloud servers where the application is using HTTPS transport.

1.9.14 INCLUDE

```
INCLUDE <product>
[user|host|group|host_group|internet|internet_group|project] <who> [id=nnn]
```

The INCLUDE line allows usage of a product by a particular user, host, group, host_group, IP address, or project. If you specify group, host_group, or internet_group, it must be defined by a GROUP or HOST_GROUP or INTERNET_GROUP line in the options file. Anyone not specified by the INCLUDE line is not allowed to use product.

INCLUDE has no effect on Named User licenses (the INCLUDE line will be ignored).

Portions of the INTERNET address can be specified with a '*' which matches any address, e.g., 172.16.7.* Note that INTERNET and INTERNET_GROUP are ineffective on RLM Cloud servers

where the application is using HTTPS transport.

If specified, the id applies this option to the license with an id of "nnn".

Note To use PROJECT the user needs to specify the project name in the environment variable RLM_PROJECT prior to starting the application. To use INTERNET, specify the internal (vs. external) IP address of the client machine.

1.9.15 INCLUDEALL

```
INCLUDEALL [user|host|group|host_group|internet|internet_group|project] <who>
```

The INCLUDEALL line allows usage of all products by a particular user, host, group, host_group, IP address, or project. If you specify group, host_group, or internet_group, it must be defined by a GROUP or HOST_GROUP or INTERNET_GROUP line in the options file. Anyone not on the INCLUDEALL list is not allowed to use any product.

Portions of the INTERNET address can be specified with a '*' which matches any address, e.g., 172.16.7.*

INTERNET and INTERNET_GROUP are ineffective on RLM Cloud servers where the application is using HTTPS transport.

Note To use PROJECT the user needs to specify the project name in the environment variable RLM_PROJECT prior to starting the application. To use INTERNET, specify the internal (vs. external) IP address of the client machine.

1.9.16 INCLUDEALL_ROAM

```
INCLUDEALL_ROAM [user|host|group|host_group|internet|internet_group|project]
<who>
```

The INCLUDEALL_ROAM line allows usage of roaming by a particular user, host, group, host_group, IP address, or project. If you specify group, host_group, or internet_group, it must be defined by a GROUP or HOST_GROUP or INTERNET_GROUP line in the options file. Anyone not on the INCLUDEALL_ROAM list is not allowed to set up roaming for any license.

Portions of the INTERNET address can be specified with a '*' which matches any address, e.g., 172.16.7.*

INTERNET and INTERNET_GROUP are ineffective on RLM Cloud servers where the application is using HTTPS transport.

Note To use PROJECT the user needs to specify the project name in the environment variable RLM_PROJECT prior to starting the application. To use INTERNET, specify the internal (vs. external) IP address of the client machine.

1.9.17 MAX

```
MAX <num> <product>
[user|host|group|host_group|internet|internet_group|project] <who> [id=nnn]
```

The MAX line limits the specified user or group to num licenses of product. If you specify group, host_group, or internet_group, it must be defined by a GROUP or HOST_GROUP or INTERNET_GROUP line in the options file. If you specify user, the special name * indicates that all users are subject to the maximum limit.

Portions of the INTERNET address can be specified with a '*' which matches any address, e.g., 172.16.7.*

INTERNET and INTERNET_GROUP are ineffective on RLM Cloud servers where the application is using HTTPS transport.

If specified, the id applies this option to the license with an id of "nnn".

Note To use PROJECT the user needs to specify the project name in the environment variable RLM_PROJECT prior to starting the application. To use INTERNET, specify the internal (vs. external) IP address of the client machine.

1.9.18 MINREMOVE

```
MINREMOVE <secs> <product>
```

The MINREMOVE line sets the minimum time before an rlmremove or RLM web interface "Remove" operation can be used after a license checkout. If no "product" is specified, the MINREMOVE applies to all products.

The default minimum removal time in RLM is 120 seconds, but it can be specified in an individual license, and an ISV can also disallow removal.

If the MINREMOVE line specifies -1 for "secs", then the license(s) cannot be removed.

MINREMOVE cannot be used to lower the minimum removal time set by the ISV in the license.

1.9.19 NO_OLD_RLMUTIL

The NO_OLD_RLMUTIL line prevents pre-RLM-v9 command-line utilities from performing a reread, remove, or shutdown operation. The pre-v4.0 RLM utilities do not respect the RLM permissions for the reread or shutdown commands, and the pre-v9.0 utilities do not respect the permissions for the remove command. Adding NO_OLD_RLMUTIL to your ISV options file will prevent these older utilities from performing these commands, and only a v9 (or newer) RLM command-line utility can be used for this purpose.

By default, all operations can be performed by all versions of the RLM command-line utilities.

In order for NO_OLD_RLMUTIL to be effective, it must be specified in both the RLM and the ISV server options files.

1.9.20 NOLOG

```
NOLOG [in|out|denied]
```

The NOLOG option instructs the ISV server to omit logging to the debug log of either CHECKIN, CHECKOUT, or DENIED messages, as specified.

You must specify one NOLOG line for each item which you do not want to be logged.

Example:

```
NOLOG denied
```

This example causes the ISV server to omit the logging of DENIED events in the debug log.

1.9.21 PRIORITY

```
PRIORITY <num> <product>
[user|host|group|host_group|internet|internet_group|project] <who> [id=nnn]
```

The PRIORITY line causes the order of queued requests to be modified. Any user/host/etc. with an assigned priority will be placed ahead of others in the queue. A new request will go at the end of all equal-priority requests in the queue, but ahead of all requests with a higher-numbered priority. All requests with no specified priority will be last in the queue. PRIORITY can specify a particular user, host, group, host_group, IP address, or project. If you specify group, host_group, or internet_group, it must be defined by a GROUP or HOST_GROUP or INTERNET_GROUP line in the options file.

Portions of the INTERNET address can be specified with a '*' which matches any address, e.g., 172.16.7.* Note that INTERNET and INTERNET_GROUP are ineffective on RLM Cloud servers where the application is using HTTPS transport.

Note To use PROJECT the user needs to specify the project name in the environment variable RLM_PROJECT prior to starting the application. To use INTERNET, specify the internal (vs. external) IP address of the client machine.

Note that PRIORITY specifications are searched in order until a match is found, and the remainder are disregarded. Thus, if a request comes in for user plum on host conservatory, the following 2 PRIORITY lines would assign this request a priority of 7, not 3:

```
PRIORITY 7 candlestick user plum
PRIORITY 3 candlestick host conservatory
```

In this case, user plum would have a lower priority for the product candlestick than other users from host conservatory, since a lower number represents a higher priority.

If specified, the id applies this option to the license with an id of "nnn".

1.9.22 PURGE_REPORTLOG

```
PURGE_REPORTLOG <#days>
```

The PURGE_REPORTLOG option instructs the ISV server to automatically remove report logs older than the specified number of days. PURGE_REPORTLOG is supported on Unix servers only (i.e., it is not supported on Windows servers).

Note that PURGE_REPORTLOG only affects the current report log name, and PURGE_REPORTLOG will have no effect if the server is not writing a report log, or if the report logs are not named according to the standard RLM rotation naming convention.

```
report_log_file_name.yyyy.mm.dd
```

The purge operation is done at midnight, immediately following the renaming of the log, if applicable.

Example 1:

```
ROTATE daily
```

```
PURGE_REPORTLOG 3
```

In this case, after the reportlog rename on Feb 23, 2016, the following reports would remain:

```
reportlog
reportlog.2016.02.23
reportlog.2016.02.22
reportlog.2016.02.21
```

Example 2:

```
ROTATE weekly
PURGE_REPORTLOG 3
```

In this case, after the rename on Feb 23, the renamed reportlog (reportlog.2016.02.23) would remain until Feb 27, at which point it would be deleted, and only the current reportlog would remain until the next rotation.

The PURGE_REPORTLOG option first appeared in RLM v12.1.

1.9.23 REPORTLOG

```
REPORTLOG [+]file_path [std | small | detailed]
```

The REPORTLOG option instructs the ISV server to write a file suitable for usage reporting to the filename *file_path*. If *file_path* is preceded with a '+' sign, the new data is appended to the file, otherwise the file is overwritten.

The third (optional) argument specifies the format of the reportlog file. Valid values are:

- **std** - write the standard report log file (the default if this field is not present)
- **small** - a smaller report log file
- **detailed** - write a reportlog that logs check-in/checkout events down to the tenth of a millisecond

All reportlogs are authenticated.

If your server is writing a reportlog, it is important to shut the server down gracefully (i.e., don't kill the server, shut it down with the RLM web interface or with an rlmdown command, or via the service controller if running as a Windows service). If you don't do this, the server won't write the final authentication record to the report log, and you will not be able to verify the last section of the report.

For details of the various output formats, see Section 3.3.

Detailed reportlogs on Windows

If a detailed reportlog is specified on Windows, RLM will produce a detailed reportlog, however, the fractional seconds field will always be 0.

1.9.24 RESERVE

```
RESERVE <num> <product>
[user|host|group|host_group|internet|internet_group|project] <who> [id=nnn]
```

The RESERVE line reserves *num* licenses of *product* for use by a particular user, host, group, host_group, IP address, or project. If you specify group, host_group, or internet_group, it must

be defined by a GROUP or HOST_GROUP or INTERNET_GROUP line in the options file. Note that reservations are subtracted from the number of floating licenses available and can only be used by the specified user(s).

Portions of the INTERNET address can be specified with a '*' which matches any address, e.g., 172.16.7.*

INTERNET and INTERNET_GROUP are ineffective on RLM Cloud servers where the application is using HTTPS transport.

If specified, the id applies this option to the license with an id of "nnn".

Note To use PROJECT the user needs to specify the project name in the environment variable RLM_PROJECT prior to starting the application. To use INTERNET, specify the internal (vs. external) IP address of the client machine.

1.9.25 ROAM_MAX_COUNT

```
ROAM_MAX_COUNT <num> <product> [id=nnn]
```

The ROAM_MAX_COUNT line limits the number of roaming licenses to num for product. Once num licenses of product are roaming, new roam requests will be denied.

If specified, the id applies this option to the license with an id of "nnn".

Note: The id option was added in RLM v11.1.

1.9.26 ROAM_MAX_DAYS

```
ROAM_MAX_DAYS <num> <product> [id=nnn]
```

The ROAM_MAX_DAYS line limits the number of days which a license can roam to num days for product.

If specified, the id applies this option to the license with an id of "nnn".

Note: The id option was added in RLM v11.1.

1.9.27 ROTATE

```
ROTATE [daily | weekly | monthly | #days ]
```

The ROTATE option instructs the ISV server to automatically close and rename the old reportlog file, and begin writing a new reportlog file, according to the time specification given.

The ROTATE command will have no effect if the server is not writing a report log.

The (single) argument specifies the frequency of rotation of the reportlog file. Valid values are:

- **daily** - rotate the report log file every night at midnight.
- **weekly** - rotate the report log file every 7 days (at midnight) after it is opened.
- **monthly** - rotate the reportlog file at midnight on the first day of the month.
- **#days** - this is an integer specifying the number of days between rotations. The report log file will be rotated just after midnight after this # of days. Specifying #days as 7 is equivalent to "weekly".

When the ISV server rotates the report log, the old report log file will be named:

```
report_log_file_name.yyyy.mm.dd
```

And the new report log file will be named:

```
report_log_file_name
```

Where:

- report_log_file_name is the reportlog filename specified in the REPORTLOG option.
- The yyyy.mm.dd is a decimal date, e.g., for September 13, 2007: 2007.09.13

Note If the file report_log_file_name.yyyy.mm.dd already exists, the server will append a sequence number to the renamed report log, e.g., report_log_file_name.yyyy.mm.dd.N where N is an integer that starts at 1 and increases until a unique filename is created. The server will make 1000 attempts to create a unique filename, then log an error.

1.9.28 TIMEOUT

```
TIMEOUT <secs> [product [id=nnn]]
```

The TIMEOUT line sets the inactivity timeout for a license to *secs* seconds for *product*. If the application using *product* does not contact the license server for *secs* seconds, the license server will reclaim the license and notify the application that the license was timed out.

Note A license could have a min_timeout specification, in which case if you specify a TIMEOUT that is lower, the TIMEOUT will be set to the minimum. The default minimum TIMEOUT in RLM is 3600 seconds (1 hour).

If no TIMEOUT or TIMEOUTALL specification is present, the license will never time out.

If multiple TIMEOUT options are specified for the same product, the last one will be used. If TIMEOUTALL is specified after a TIMEOUT option, the TIMEOUTALL value will be used.

If *product* is not specified, the timeout applies to all products. This is equivalent to TIMEOUTALL.

If specified, the id applies this option to the license with an id of "nnn".

Note: The id option was added in RLM v11.1.

1.9.29 TIMEOUTALL

```
TIMEOUTALL <secs>
```

The TIMEOUTALL line sets the inactivity timeout for all licenses to *secs* seconds. If the application using a product does not contact the license server for *secs* seconds, the license server will reclaim the license and notify the application that the license was timed out.

Note A license could have a min_timeout specification, in which case if you specify a TIMEOUT that is lower, the TIMEOUT will be set to the minimum. The default minimum TIMEOUT in RLM is 3600 seconds (1 hour).

If multiple TIMEOUTALL options are specified, the last one will be used. Any TIMEOUT

option after the TIMEOUTALL option will be used for that specific product.

If no TIMEOUT or TIMEOUTALL specification is present, the license will never time out.

1.9.30 TIMEZONE

```
TIMEZONE <timezone-spec> [product [id=nnn]]
```

The TIMEZONE option allows the specification of a set of valid timezones for the client machine that performs the license checkout. The timezone-spec is a 24-bit HEX number, with one bit set for each timezone you wish to be valid. Bit 0 represents GMT and each bit to the "left" of bit 0 represents one timezone (one hour) west of GMT. Thus bit 5 would be EST, bit 8 would be PST, bit 23 would be one hour east of GMT, etc. Note that RLM uses the current local time, so the timezones will move by one hour when Daylight Savings Time is in effect (ie, PST varies from 7 to 8 hours west of GMT), so for example, to enable use of a license in PST during both normal and Daylight Savings Time, specify bits 7 and 8 in the *timezone-spec* as follows:

```
TIMEZONE 180 <product-name>
```

If *product-name* is specified, the timezone restriction applies to all licenses for product *product-name*. If *product-name* is not specified, the timezone restriction applies to all licenses for this server.

The timezone restrictions specified in the TIMEZONE option are used to further restrict the license. If the combination of the timezone specified in the license and the timezone in the option would result in a license that has no valid timezones, then the TIMEZONE option is ignored. So, for example, if the license specified timezone=fff000 and the option specified TIMEZONE ff, the result would be no valid timezones, and the license timezone (fff000) would be used. Note that this can also happen if you have more than one TIMEZONE option in your options file as well, eg:

```
TIMEZONE ff0000
TIMEZONE 180 <product>
```

would cause the second TIMEZONE option (for product *product*) to be ignored.

If specified, the id applies this option to the license with an id of "nnn". Note: The id option was added in RLM v11.1.

1.9.31 USER

This line is used for client authentication, and must be added using the *rlmadduser* command.

For more information, see Section 1.6

Advanced Topics

2.1 Token-Based Licenses

Token-Based licenses are licenses that are defined in terms of another license. For example, an application could request a license for product **write**. If this were a *normal* license, the product **write** would appear in the license file (if the request succeeds) with a license count (or *uncounted*). On the other hand, if this were a *token-based* license, the product **write** would appear in the license file without a count, but with a specification of one or more other products which are used to satisfy the request. When the license server encounters a request for a **token-based** license, it uses the other products specified in the license to satisfy the request, rather than the originally requested product. These other licenses are called the *primary* licenses.

There are two main uses for token-based licenses. The first use of token-based licenses allows a license administrator to mix-and-match different products as their needs change. If several products are all defined in terms of a single pool of *primary* licenses, the license administrator can control license usage as needs demand. This can be a benefit to both ISVs and their customers, since as new products are introduced, if they are licensed based on the same *primary* license, all customers instantly have access to the new products without having to go through a purchase-order cycle.

Another use of token-based licenses is to allow alternate licenses to satisfy a request for a product. To use the familiar example, if product **write** checks out a *write* license, the addition of a *token-based* license for *write* mapping it to *office* would allow an *office* license to be used in the case where no *write* licenses are available. Even though the *office* license is a more expensive license, the customer is allowed to continue working by consuming the more expensive *office* license. Several *token-based* licenses can be used in this way, and the order of the licenses in the license file will determine the order that alternate checkouts are attempted.

A token-based license differs from a normal license in a few significant ways:

- The count field contains one of the 3 token keywords (**token**, **token_bound**, or **token_unlocked**) rather than an **integer**, **uncounted**, or **single**.
- The license has a token spec: **token="<prod ver count> … <prodN verN countN>"**
- The only optional parameter on a token-based license which is used by RLM is the start date. All other optional parameters are ignored.
- License Administration option processing is different for *token-based* licenses. See below.
- There are a few restrictions on *token-based* licenses, especially for queuing. See below.

Warning The keyword **token_locked** was deprecated in v12.4 and replaced with token_bound. Licenses with token_locked keywords will continue to operate, but they will

not be locked to the server's hostid).

2.1.1 Types of token-based Licenses

When a product is specified as a *token-based* license, requests for that product are turned into requests for the *primary* license(s) specified in the **token=** part of the license. For example, consider this license for product **test** (*primary* license **dollars**):

```
LICENSE reprise test 1.0 permanent token token="<dollars 2.0 5>" sig=xxxx

LICENSE reprise dollars 2.0 permanent 10 sig=xxxx
```

This license is called a *simple token-based* license. Any *token-based* license that maps a checkout of one product into a (single) primary license is a *simple token-based* license.

A *token-based* license can map one request into multiple checkouts, however. In this case, it is called a *compound token-based* license. Using our product **test** as an example again:

```
LICENSE reprise test 1.0 permanent token token="<dollars 2.0 5> <cents 3.4
53>" sig=xxxx
LICENSE reprise dollars 2.0 permanent 100 sig=xxxx
LICENSE reprise cents 3.4 permanent 1000 sig=xxxx
```

Now, a request for 1 license of **test** v1.0 would result in the license server checking out 5 v2.0 licenses of the product **dollars**, and 53 v3.4 licenses of the product **cents**. If **both** of these primary licenses are available, the checkout request for test succeeds, otherwise it fails. Note that when a compound tokenbased license is checked out, the rlm_license_xxx functions return information about the first license in the list only. In this example, rlm_license_xxx functions would return information about the dollars license.

2.1.2 Issues with roaming

When a license is roamed, only the name of the license that was requested in the checkout can be used on the roamed system. What this means, in practice, is that as long as you use the same name to attempt the checkout, the checkout will succeed. However, sometimes token-based licenses are used to migrate a product license name.

For example, if v2.0 of your product checks out an "old_name" license, and v3.0 checks out a "new_name" license, and you have a token license definition to map the "new_name" checkout to "old_name", then a roamed license created by checking out "new_name" will not work for the product that checks out "old_name", even though the "new_name" checkout may have been satisfied by an "old_name" license. In other words, while this mapping works on the server side, it does not work for a roamed license.

2.1.3 The License Count Keywords

In a *token-based* license, the count keyword is one of:

- token
- token_bound
- token_unlocked

token and *token_unlocked* imply that the *token-based* license itself does not include the license

server hostid in its license signature. This makes the license usable in **any** license file.

Note token and token_unlocked are 100% equivalent.

---

token_bound means that the *token-based* license includes the license server hostid in its signature, and is valid only in *this* license file.

2.1.4 The token= keyword

In a *token-based* license, the *token*= keyword specifies the *primary* licenses which are checked out in response to a request for the *token-based* license itself. Specify one or more licenses to be checked out. These licenses can also be *token-based* licenses themselves (in which case the *primary* license(s) will be the ultimate expansion of all *token-based* licenses). The format is:

```
token="<product1 ver1 count1>[ <product2 ver2 count2> .... <productN verN
countN>]"
```

The request for the one of the original licenses turns into checkouts of:

- **count1** of product1, ver1
- **count2** of product2, ver2
- …
- **countN** or productN, verN

2.1.5 Nesting token-based licenses

The definition of a token-based license can include other licenses which are token-based licenses themselves. For example:

```
LICENSE reprise test 1.0 permanent token sig=xxxx token="<t2 2.0 5>"
LICENSE reprise t2 2.0 permanent token sig=xxxx token="<dollars 2.0 5>"
```

In this example, a request for one test v1.0 license results in 25 dollar v2.0 licenses checked out.

Note The license server uses nesting of greater than 20 levels to detect token "loops", so any licenses nested this deeply will be rejected. Also note that nesting has no effect on whether a *token-based* license is *simple* or *compound* - this is determined solely by whether a single request maps into a single checkout or not.

2.1.6 Restrictions on token-based licenses

There are a few restrictions on token-based licenses:

- All *token-based* licenses are processed by the license server, so there can be no uncounted, node-locked *token-based* or primary licenses that operate without a license server. (However, a license server can serve a node-locked, uncounted *primary* license.)
- All individual checkouts for a *compound token-based* license are satisfied by a single license server. This means that if a license turns into checkouts of *primary* licenses a, b, and c, where only a and b are available on one server and only c is available on a second server, the request will fail.

- The mapping from a token-based license to its *primary* license(s) maps to a single *primary* license. This means that a single token-based license generates requests of a **single primary license pool** (for each of it's primary licenses). So, for example, if your token-based license turned into a request for *primary* license *prim*, the first pool of *prim* licenses would be used, and other pools of *prim* licenses could not be used to satisfy this token request.
- Queuing is not allowed for *compound token-based* licenses. This would lead to license server insanity.

Note Token licenses, once fully resolved, should NOT contain multiple instances of the same primary license name. If this is the case, the server can inadvertently overdraft that license while processing the token-based checkout.

The following licenses are examples of licenses which contain multiple instances of the same primary license:

```
LICENSE reprise t1 1.0 permanent token token="<x 1.0 1><x 2.0 1>" sig=...
```

or:

```
LICENSE reprise t1 1.0 permanent token token="<t2 1.0 1><x 2.0 1>" sig=...
LICENSE reprise t2 1.0 permanent token token="<x 2.0 1>" sig=...
```

The server will generate a warning about this condition similar to the following:

```
01/01 10:09 (reprise) *WARNING: token definition for t1 has multiple
01/01 10:09 (reprise) instances of primary license x
```

For the 2 examples above, if there is only a single license for "x", a checkout of "t1" will cause the server to checkout 2 copies of "x" and return an RLM_EL_OVERSOFT error to the application.

2.1.7 rlmremove and token-based licenses

The *token-based* license itself cannot be removed. If any of the *primary* licenses are removed, the server will remove all *primary* licenses, and the application will notice a loss of license on the next check with the server resulting from an rlm_get_attr_health() call or from the heartbeats generated by rlm_auto_hb(). In this sense, the *token-based* license works just like a regular, non *token-based* license.

2.1.8 Report Log

When a *token-based* license is checked out, the name of the license requested (and its version) is logged in the checkout record for each resulting *primary* license checkout. If the check-out results from the dequeuing of a previously queued request, the requested product name/version will not appear (they appear in the queue record, but not the checkout record).

2.1.9 License Administration Options

The only options that apply to the token-based license itself are the include and exclude keywords:

- INCLUDE

- INCLUDEALL
- INCLUDEALL_ROAM
- EXCLUDE
- EXCLUDEALL
- EXCLUDEALL_ROAM

All other license administration options have no effect on the token-based license.

All options affect the primary licenses, however.

2.2 How to Queue for Licenses

In RLM, queuing for licenses is under the software user's control for well-behaved applications.

Unlike older license managers, RLM will queue for a license at every available license pool on every server, meaning you will not be stuck in a queue when there are licenses available on a server elsewhere.

In order to enable your application to queue for a license, set the environment variable **RLM_QUEUE** to any value. If **RLM_QUEUE** is set, the application will queue for its license if it is not able to check the license out on any license server. Once the application recognizes that the license has been granted by a server, it automatically de-queues the requests at all the other servers.

Note This capability depends on your Software Provider having coded their application to handle the QUEUED status return from the license server.

Note If you have set RLM_ROAM (or <ISVNAME>_ROAM), the setting of RLM_QUEUE is ignored.

2.2.1 EXPRESS License Pools

RLM has the concept of an **EXPRESS** License Pool. If a license pool is marked as **EXPRESS** (the default), then queued requests which can be satisfied immediately are granted independent of whether other requests are ahead in the queue. If EXPRESS is turned off, then a queued request will always be placed at the end of the queue if there is a waiting request.

For more information on how to set up license pools as EXPRESS, see Section 1.9.10.

Since RLM applications queue at all license servers, a request for, say, 5 licenses, might take a long time to grant. In the meantime, up to 4 licenses could be available which could be granted to other applications which need only one license each. By setting some license queues to EXPRESS and turning off EXPRESS on others, you can bias the operation of different license servers to handle single or multiple license requests more favorably.

2.3 How to use Roaming Licenses

RLM has the ability to allow a floating license to roam to a system which will subsequently be disconnected from the network. The resulting license can be used for the number of days specified when the license was set to roam and is checked back in automatically at the end of this time. In addition, you can return the roamed license back to the license pool early if this is desired.

2.3.1 How to know if License Roaming is Available

Your ISV makes the decision to enable roaming licenses. If you have been issued an rlm_roam license, then roaming is available to you with the restrictions specified in the *rlm_roam* license.

Note that you must be able to check out an rlm_roam license on any system that is disconnected. Practically speaking, this means that disconnected systems need a local license file with a nodelocked *rlm_roam* license in it.

2.3.2 How to make a License Roam

If you have an *rlm_roam* license, set the environment variable **RLM_ROAM** to the number of days which you would like to use the license (this license will be available until midnight on the last day of the roam, so for example if you specify one day, the license is available until midnight tomorrow). Once **RLM_ROAM** is set, run the product and let it check out its license(s). If the checkout succeeds, then the license is set up to roam. You can repeat this procedure for any other products that have roaming capability enabled. If you set **RLM_ROAM**, the setting of **RLM_QUEUE** is ignored.

Note Beginning in RLM v12.2, if **ISVNAME_ROAM** is set (where *ISVNAME* is the ISV's name in uppercase), it will be used in place of **RLM_ROAM**. In the remainder of this document, we will use the name **RLM_ROAM** to signify either **RLM_ROAM** *or* **ISVNAME_ROAM**.

Be sure your *rlm_roam* license is contained in a license file that is local to this system, otherwise you will not be able to use these licenses.

Your Software Provider might have supplied a GUI to handle the setting of the RLM_ROAM environment transparently to you. If this is the case, they will have documented this capability in their application's documentation.

Note Beginning in RLM v11.1, **RLM_ROAM** can be set to the special value "today". If set to "today", the license will roam until the end of the day today. Please note that if you use a v11.1 client with an older (pre-11.1) server, **the roam time will be one day longer than what you specified**. Pre-11.1 clients will roam as expected with 11.1 and later servers, however they will not be able to take advantage of the "today" value of **RLM_ROAM**.

2.3.3 While your system is connected to the network

After the initial checkout of the roaming license, but during the time your system is connected to the network, the value of **RLM_ROAM** will affect the behavior of the roaming license. If **RLM_ROAM** remains set to the original value, the license will be "refreshed" each day to the total roam time. On the other hand, if **RLM_ROAM** is set to 0, the original roam end date will remain, and no subsequent checkouts of the license will alter the final roam day.

What this means, for example, is that if you set **RLM_ROAM** to 14 days, the license will always be available to you 14 days after the last time you checked it out on the network. If, however, you first set **RLM_ROAM** to 14 days, check out the license, then set **RLM_ROAM** to 0, the license will be available on the disconnected system for 14 days from the date of the first checkout, no matter how many times you check it out while connected.

Again, your Software Provider may have taken care of this for you in their GUI.

2.3.4 While your system is disconnected

During the time your system is disconnected, **RLM_ROAM** must remain set to a non-negative value. This license will be available on this system for the number of days you requested, and you no longer need to be able to access the license server from which the license was granted.

On the network, the license server will show the license checked out to you.

Note **RLM_ROAM** does not need to be set *while* the system is disconnected.

Note If you are connected to the network and do not set **RLM_ROAM**, you may check out a license from the license server rather than using the roamed license. If this is not desired, you can set **RLM_ROAM** to a positive number which will cause the roamed license to be used. Also note that if **RLM_ROAM** is not set, the checkout of the "rlm_roam" license must come from a local license file, not from the license server.

2.3.5 If you want to return the Roamed License early

If your plans change and you would like to return the license before the roaming time has expired, reconnect the system to the network (so that it can contact the original license server), and set the environment variable **RLM_ROAM** to -1. Now run the program and let it check out the license. Once the program exits (or does a check-in), the roamed license will be returned to the license pool on the server.

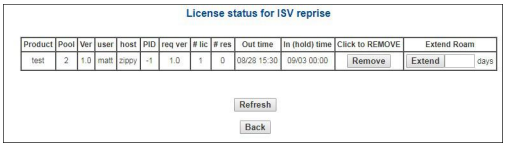
> **Warning** If you change the server node name or port number after you roam the license, you will not be able to return the license early.

2.3.6 If you want to extend the period of the Roamed License

If your plans change and you would like to keep the license until after the roaming time has expired, but cannot reconnect the system to the network, you can extend it by having someone on the network extend the roam in the RLM web interface. Note that this only works before your original roam period has expired, and only on a v12.3 or later server (and client).

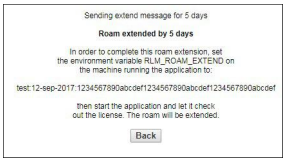
Note This capability must be explicitly enabled by your ISV; it does not appear automatically. If the "Extend Roam" column in the License Status screen (shown below) does not appear, this means your ISV does not support the ability to extend a roam remotely.

If your ISV does support this, to extend the roam, someone on the network views the license status and finds the license you have roaming. This user must have the "extend_roam" privilege to see the last field. By selecting Status → License Usage for this license, they will see a screen similar to the following:



By entering the # of days to extend the roam in the text box on the right and pressing "Extend", the process of extending the roam will start, and the following screen will appear:

If "Yes – Extend user's license" is selected, the roam will be extended on the server side, and the screen will display the roam extension code:



Now, on the disconnected client, set the environment variable RLM_ROAM_EXTEND to the roam extension code:

```
setenv RLM_ROAM_EXTEND
test:12-sep-2017:1234567890abcdef1234567890abcdef1234567890abcdef
```

and run the application which checks out the "test" product. Once the checkout happens, the roam period is extended.

If you do not record the *roam extension code* from the web interface, it is also recorded in the server's debug log as follows:

```
08/28 15:45 (reprise) ROAM EXTENDED: 5 days test v1.0 by matt@zippy
08/28 15:45 (reprise) Roam extended for product test by 5 days
08/28 15:45 (reprise) Roam extension code, place in RLM_ROAM_EXTEND
08/28 15:45 (reprise) and run client to check out license:
08/28 15:45 (reprise)
test:12-sep-2017:1234567890abcdef1234567890abcdef1234567890abcdef
```

Note that if you extend the roam past a daylight-savings-time transition day, you may get one hour more or less than you might expect (when viewing the RLM web interface), but the roam ends at midnight.

2.3.7 Special note on roaming licenses from a broadcast-discovered server

Reprise Software strongly recommends that you do not attempt to use the broadcast method to locate the server if a license is to be roamed. While this will often work, there are circumstances where RLM cannot re-locate the original server which supplied the licenses, and the roamed license cannot be returned early.

If the broadcast server is the only license server on the network, RLM will be able to return these licenses starting with v11.2, however, earlier versions of RLM could not return the roamed license. If there are multiple license servers on the network, there is no guarantee that the application will find the correct server to return the licenses, even if you manually set the port and host of the server before attempting to return the license. If you know the port and host of the ISV SERVER which roamed the licenses, you can then:

- Make sure the ISV server is running on the same port# (set the port # on the ISV line, if necessary)
- Set RLM_LICENSE to this port and host (the ISV server, not RLM)
- Set **RLM_ROAM** to -1 and check out the license again.

(This technique will work for earlier versions of RLM as well).

2.4 Failover License Servers

RLM provides the capability for a license server to take over the license complement of another server which has gone down. The server whose licenses are being taken over is called the *primary server*. The server which takes over the license load of the *primary server* is called the *failover license server*. During the time that the failover server is serving the licenses, no roaming operations are permitted on the licenses.

Note When using failover servers, there cannot be a firewall between the two servers.

Note Failover servers are not supported on HP/UX or IBM AIX systems.

The ability for a server to take over the load of another server is selected by an ISV on a customer-bycustomer basis, and enabled by an *rlm_failover*, *rlm_failover_server* or *rlm_-failover_server_activate* license in the license file of the server that is to take over.

The difference between an *rlm_failover* license and an *rlm_failover_server/rlm_failover_server_activate* license is whether RLM checks for the machine being down (*rlm_failover*) or the RLM process on the machine being down (*rlm_failover_server/rlm_failover_server_activate*) before serving the licenses from the primary server. In addition, the *rlm_failover_server_activate* license requires a license activation from an Activation Pro server before proceeding to serve the primary server's licenses. With this license, you can see in your ActPro database how often a particular customer's failover servers have been activated.

While the *failover license server* can be serving its own compliment of licenses, Reprise Software recommends that it should have no licenses of its own, but simply be standing by, waiting for one (or more) other server(s) to fail so that it can take over. In the remainder of this chapter, we will use the term **failover license** to mean either the *rlm_failover*, *rlm_failover_server*, or *rlm_-failover_server_activate* license.

2.4.1 Enabling a Failover Server

In order to enable a failover license server, the ISV issues a **failover license** with the following characteristics:

- **count=***some non-zero value*
- **hostid=**<*hostid of the primary server*>
- **_primary_server=**<*hostname of the primary server*> (rlm_failover licenses)
  <**OR**>
  _primary_server=<*host or port@host of the primary server RLM process*> (rlm_failover_server or rlm_failover_server_activate licenses)
- **akey=***activation-key* – this is the activation key to be used to activate in the case of an rlm_-failover_server_activate license.
- **issuer=***url-of-activation-server* - this is the URL of the activation server to be used to activate in the case of an rlm_failover_server_activate license.

Note A failover_server/failover_server_activate license can specify just the hostname of the server, in that case, the default port (5053) is used.

2.4.2 Configuring Failover License Servers

Reprise Software recommends configuring *failover license servers* as stand-alone servers that do not serve their own complement of licenses. In other words, Reprise recommends configuring a license server that has **only failover licenses** for the other license servers on the network. In general, one license server configured in this way should be sufficient to support failover of all other license servers on the network.

The exception would be a case where each individual license server is serving thousands of clients, in which case we recommend that you configure a *failover license server* for each one or two of the normal license servers.

Note Failover license servers do not support license roaming operations. Actually, this is determined on a license pool by license pool basis - any license pool that contains any licenses from a primary server will not support any of the roaming operations.

Example Failover License Server license file

The following license file would specify a license server that is acting as a failover server (on node **failover_host**, hostid **11111111**) for the license server on hostid **12345678** (node **primary_server**):

```
HOST failover_host 11111111 5053
ISV reprise
LICENSE reprise rlm_failover 1.0 permanent 1 hostid=12345678
        _primary_server=primary_server sig="x"

## Alternate LICENSE line:

LICENSE reprise rlm_failover_server 1.0 permanent 1
        hostid=12345678 _primary_server=5053@primary_server sig="x"

## Alternate for an activated failover server:

LICENSE reprise rlm_failover_server_activate 1.0 permanent 1
        hostid=12345678 _primary_server=primary_server
        issuer=hostedactivation.com akey=1234-5678-9012-3456
        sig="x"
```

Note The hostid in the LICENSE line for the failover license is the hostid of the server on the node primary_server).

Note The "_primary_server" keyword was called "_failover_host" prior to RLM v9.2. "_primary_server" is an alias for "_failover_host", which will continue to work. However, RLM versions prior to v9.2 will only recognize the "_failover_host" keyword to specify the primary server name/port #.

The license file on "primary_server" should be a normal RLM license file, without the rlm_-failover licenses. This license file MUST be present and readable by RLM on the failover machine as well, otherwise the failover server will not serve the licenses from the primary when it goes down. So, for this example, a license file on the primary server would look something like this:

```
HOST primary_server 12345678 5053
ISV reprise
LICENSE reprise product1 1.0 permanent 3 sig=....
LICENSE reprise product2 1.0 permanent 7 sig=....
 ...
```

```
LICENSE reprise productN 1.0 permanent 2 sig=....
```

2.4.3 Installing Failover License Servers

When you receive the failover license file, do the following to install your *failover license server*:

1. Install *RLM*, your *ISV server* and the *primary server license files* on the *primary server*. Start RLM.
2. Install *RLM* and your *ISV server* on the *failover license server* node.
3. Install all license files from the *primary server on the failover license server node*.
4. Edit the license file with the **failover license** to put the hostname of the *primary server* in the *_primary_server*= field, and the hostname of the *failover license server* on the HOST line.
5. Install any ISV options in an options file and make it accessible to the *ISV server* either through its license file or by giving it the default options filename of *isv.opt*.
6. Ensure that *RLM* will process all the license files above and start *RLM*.
7. Ensure that your license administrator's client software sees the failover license server's license file, i.e., put a license file with a HOST line for the failover server *where your application will find it*. If you omit this step 6, the failover license server will take over, but your application will not be able to check out a license from the failover server.

2.5 Transferring Licenses to Another Server

RLM has the ability to transfer a set of licenses from one license server to another. This is useful, for example, in situations where a temporary project involves a number of users who need access to the software and the primary license server is across a WAN with a high-latency connection. In this case, a number of licenses can be moved to a server on the part of the WAN with the users. Once the project is complete, the licenses can be returned to the primary server.

When licenses are transferred from the primary (**source**) license server, they are no longer available at the **source** server, and are then available at the **destination** server. The licenses are available at the destination server as long as the source server is up; in other words, if the source server goes down or the network loses connectivity, the destination server will lose its complement of transferred licenses.

License transfers are always initiated by the destination server - the server which is going to receive the licenses. The license transfer is specified in the RLM administration GUI on the destination server.

2.5.1 Setting up a license transfer

Any platform which your ISV supports can be used as the destination for a license transfer. The particular machine does not need to have any RLM-licensed applications present - all you need is the RLM (or rlm.exe) binary as well as an ISV server binary or settings file.

License transfers are accomplished by creating a set of transfer definitions. Each transfer definition describes the source license server, the licenses you wish to transfer, and some parameters of those licenses.

To set up the license transfer, do the following:

- Place the RLM and ISV servers into a directory, and create a license file that specifies the HOST and ISV lines:

```
HOST hostname any port
ISV isvname
```

Note We used "any" as the hostid. The hostid is irrelevant, because this license server will act as a client of the source server and check out licenses from it - these transferred licenses are not ever going to be locked to the destination server.)

- Next, start RLM - RLM will start up, and start the ISV server.
- Next, point your browser to the RLM admin page (hostname:5054), and select Status from the menu on the left-hand side.
- Next, select Transfer from the right-hand side of the list of buttons for the ISV for which you wish to edit a transfer definition. This will bring up the Edit License Transfer Definitions page, which will indicate that no license transfers have been defined. Select Add New Transfer Definition… at the bottom of the page, and you will be able to specify the license transfer parameters:

2.5.2 Add License Transfer Definition for ISV isvname

```
Transfer server type: Choicelist: {RLM | ISV-defined}
Source Server Hostname:
Source Server Port:
Backup Server Hostname:
Backup Server Port:
Product:
Minimum Version:
Count:
Days to hold license;
```

Fill out this form as follows:

- In the first section of this form, you select the type of license server on which the transferred licenses are served. "RLM" indicates an RLM license server. "ISV-defined" indicates another licensing system for which your ISV has added support, if applicable.
- Next, you specify the source license server with a hostname and a port number. (Some other licensing systems will use a backup server name and port # - RLM does not use this).
- Next, you specify the product name, minimum allowable version, and the number of licenses you would like to transfer to this (the **destination**) server.
- Finally, if you have RLM v10.0 or later, you have the option to specify "Days to hold license". If you specify this as a non-zero value, the license transferred is a "disconnected (roamed)" license. This license can be used even when the destination license server is not in communication with the source license server. For this to work, your ISV must have provided you with an "rlm_roam" license, or enabled this capability in their license server.
- Finally, click **Add Transfer** at the bottom of the form. The transfer specification will be saved.

If you would like these licenses to be transferred immediately, select **Reread/Restart Servers** on the left-hand side of the main form. This will cause the ISV server to reread its license files and transfer definitions, and process the license transfer. If you have already transferred licenses of this product to this destination server and you wish to change the number of licenses, you must

delete the first transfer definition and re-create it with the new number of licenses, then perform a reread on the server. When you delete the old transfer definition, all clients using that license will receive an error on their next heartbeat (RLM_EL_SERVER_LOST_XFER, -51). In most cases, you will be able to delete the old transfer, create the new one, and perform a reread on the server within the time the client does its next heartbeat to the server.

Once transfer definitions are present, pressing the **Transfer** button on the status page will display a list of transfer definitions. Each of these can be independently enabled/disabled or deleted.

2.5.3 Restrictions on License Transfer

Not all licenses can be transferred to another server. In particular, the following licenses cannot be transferred:

- User-based licenses
- Host-based licenses
- Named-user licenses
- Token licenses

Note that any license to be transferred must be able to be checked-out on the destination license server. In particular, this means that most node-locked licenses cannot be transferred (unless you set up a separate destination server on each node-locked host to accept the licenses). Other license restrictions, such as platforms= will restrict licenses which can be transferred as well.

Transferred licenses cannot roam. If you attempt to roam a transferred license, you will receive a -31 (RLM_EL_TOOMANY_ROAMING - "too many licenses roaming") error (pre RLM v10.0), or a -57 (RLM_EL_NONE_CANROAM - "This license not allowed to roam") on RLM v10.0 or later.

When you disable a transfer, the licenses are returned to the source server immediately. All clients will be notified that the licenses are no longer available on their next heartbeat check. In this sense, disabling a transfer is identical to deleting the transfer, with the exception that the transfer can be reinstated by re-enabling it and doing a reread on the server.

2.5.4 Additional notes on disconnected (roamed) transfers

If your Software Provider allows it, license transfers can happen onto a disconnected server node. For disconnected (roamed) transfers, the GUI also presents a "Refresh" option, if the transfer is enabled. If "Refresh" is pressed, the transferred license will be checked back in and then checked back out again, in order to refresh the roam time to the specified number of days. Note that the server performs this refresh action each time it is started (but not on a reread). The "Refresh" button allows you to reset the # of days of disconnected operation at any point that the server is running, and the source license server is up.

Both "disable" and "delete" operations on disconnected (roamed) transfers cause the transferred license to be returned to the source server. You should always delete or disable a disconnected (roamed) transfer when the source server is up, so that the destination server can return the licenses cleanly. RLM makes every attempt to catch errors in the return of roamed license, but any error here will cause the transferred license to remain checked-out on the source server. Note that when RLM detects any error in returning the license, it keeps the transfer definition in place, so that you can return it when the source server is up. So the source server must be up to return the license, anyway.

2.5.5 Additional notes on ISV-defined transfers

For the ISV-defined transfer type, each individual ISV may impose other restrictions on what types of licenses can be transferred. This, of course, can vary for other license managers as well.

Reference Material

3.1 RLM Environment Variables

RLM uses a number of environment variables to control licensing behavior. These variables are discussed in this section.

Note: To set an environment variable on Windows systems, bring up the Control Panel, select System, click on the Advanced Tab, select Environment Variables, then select New or Edit in the User Section.

3.1.1 RLMAUTH

RLMAUTH is used to set the username and password for the client application which will be checking out licenses from the ISV Server. Set RLMAUTH=<username>:<password>. This username and password must match a username and password contained in Section 1.9 in order to checkout licenses from the server. Note that this capability requires an RLM v13.0, or higher, client as well as an RLM v13.0, or higher, license server. See Section 1.6 for more details.

3.1.2 RLM_COMM_TIMEOUT

RLM_COMM_TIMEOUT sets the application timeout for receipt of messages from the license server. If RLM_COMM_TIMEOUT is set, the read timeout will be set to the value of this environment variable. The default is 5 seconds, which should be sufficient in most, if not all, situations. (The default was 3 seconds prior to RLM v9.3)

Note RLM_COMM_TIMEOUT is specified in milliseconds, so for a 7 second timeout, set RLM_COMM_TIMEOUT to 7000.

RLM_COMM_TIMEOUT first appeared in RLM v9.3.

3.1.3 RLM_CONNECT_TIMEOUT

RLM_CONNECT_TIMEOUT sets the application timeout for a connection to an individual license server. If RLM_CONNECT_TIMEOUT is set, the connection timeout will be set to the value of this environment variable. The minimum connection timeout is 5 seconds, and the default is 10 seconds. If RLM_CONNECT_TIMEOUT is set to a negative value, the connect

timeout will be the absolute value of RLM_CONNECT_TIMEOUT, and if any particular server connection times out, no further attempts will be made to that server again. If RLM_CONNECT_TIMEOUT is set to a positive value, a connection will be attempted to the server even if it timed out on the last attempt. This is the default behavior in RLM.

RLM_CONNECT_TIMEOUT is specified in seconds. Note also that the value you specify is divided by 5 (and truncated), then used 5 times, so the minimum resolution is 5 seconds. For example, if you set RLM_CONNECT_TIMEOUT to 8, the timeout will be 5 seconds. Setting it to 11 will yield a 10-second timeout, etc.

3.1.4 RLM_DEBUG

RLM_DEBUG, if set to a product name, will cause any RLM-licensed application to output product debugging information about the specified product. If RLM_DEBUG is set without a value, the debugging information will be output for all products which can be found.

RLM_DEBUG was introduced in RLM v9.0.

3.1.5 RLM_DIAGNOSTICS

RLM_DIAGNOSTICS, if set to a file name, will cause any RLM-licensed application to output diagnostic information to the specified file. If RLM_DIAGNOSTICS is set without a value, the diagnostic information will be written to the standard output, which may or may not be desirable, depending on the application.

3.1.6 RLM_EXTENDED_ERROR_MESSAGES

If RLM_EXTENDED_ERROR_MESSAGES is set, the internal RLM functions which generate error messages will output more verbose messages (in certain cases) with suggestions for solving the problem.

3.1.7 RLM_LICENSE

The RLM_LICENSE variable specifies the path to one or more license files and/or license servers.

See Section 1.4.9 for a complete description of how to use RLM_LICENSE.

3.1.8 RLM_LICENSE_PASSWORD

The RLM_LICENSE_PASSWORD variable specifies a password for access to particular licenses which have the _password attribute.

See Section 1.4 for a complete description of how to use RLM_LICENSE_PASSWORD.

3.1.9 ISV_ACT_URL

ISV_ACT_URL (where "ISV" is replaced with the ISV name) is used to override the url that is built into the application, used in rlm_activate() and rlm_act_request() API calls.

This is primarily used for debugging purposes.

3.1.10 ISV_LICENSE

ISV_LICENSE (where "ISV" is replaced with the ISV name) is used exactly like RLM_LICENSE, however if ISV_LICENSE is present, it will be used instead of RLM_LICENSE.

3.1.11 RLM_PATH_RANDOMIZE

Setting RLM_PATH_RANDOMIZE (to any value) causes RLM to randomize the license path before any subsequent processing. This is useful in large installations to provide a rudimentary form of load balancing by causing different users with the same setting of RLM_LICENSE or ISV_LICENSE to use different license servers.

Note that RLM_PATH_RANDOMIZE has no effect on the RLM or ISV servers. RLM_-PATH_RANDOMIZE operates by selecting a new starting point in the license list and wrapping around from the last license spec to the first. Otherwise, the order or the license list is preserved.

3.1.12 RLM_PROJECT

RLM_PROJECT communicates project information to the license server. The value of a client's setting of RLM_PROJECT (up to 32 characters) is logged in the report log file for later use.

3.1.13 RLM_QUEUE

RLM_QUEUE informs an application that it should queue for a license if none are available. If RLM_QUEUE is set, any subsequent application started will queue for its license if none are available.

If RLM_ROAM is set, RLM_QUEUE is ignored (starting in RLM v12.0).

3.1.14 RLM_ROAM

RLM_ROAM controls license roaming operations. If RLM_ROAM is set to a positive integer N, subsequent checkout requests will attempt to create roaming licenses for N days. If RLM_ROAM is set to any negative integer, any subsequent license checkouts will cause a roaming license to be returned to the floating license pool. This last operation must be done when connected to the network such that the original license server can be contacted.

Setting RLM_ROAM disables RLM_QUEUE (starting in RLM v12.0).

3.1.15 RLMSTAT

RLMSTAT, if set, will cause the rlm checkout routine to print status as it attempts the checkout. This is sometimes useful to diagnose license checkout failures. The format of this output is:

```
RLMSTAT(location): (product_name) license_path: error
```

where: * **location** is the letter 'N', 'C', or 'U', meaning:

- N - attempt to check out a local nodelocked, uncounted (or single) license
- C - attempt to checkout a license from a license server which is already connected
- U - attempt to checkout a license from a license server with no prior connection

- **product_name** is the product name being checked out
- **license_path** is the path of license files or port@host specs
- **error** is the error status from the checkout

3.1.16 Obsolete Environment Variables

RLM_DISCONNECTED

RLM_DISCONNECTED informs an application that the system is not connected to the internet. If RLM_DISCONNECTED is set, any subsequent application started will not attempt any internet operations outside the local firewall (unless the license server is specified to be outside the firewall).

RLM_DISCONNECTED was removed in v3.0, as it was no longer used.

3.2 RLM Performance Testing

The RLM License Administration bundle includes several useful binaries as well as a copy of this manual. The License Administration bundle includes a performance test program called **rlmtests**.

RLM License Administration bundles can be found here: https://reprisesoftware.com/support/admin/

Agree to the license terms and click **GET THE BUNDLE**, then select the kit(s) you want to download. Save these on your system, then uncompress and (on Unix) extract the binaries with the tar xvf command. On Windows, the kit is in a WinZip archive.

Each kit has a descriptive name on the website. The file names of the kits follow Reprise Software's platform naming conventions, with .tar.gz appended for Unix, or .zip for Windows.

To unpack the License Administration bundle, follow these steps:

At the shell prompt on Unix:

```
% gunzip platform.tar.gz
% tar xvf platform.tar
```

That is all there is to it - the kit contains pre-built binaries and this manual.

On Windows, the kit is in WinZip format. Extract the binary directory (x86_w3 for 32-bit or x64_w3 for 64-bit) to a convenient location.

The performance test can be run by typing the command **rlmtests** at the shell on Unix or in a command window on Windows. When run, **rlmtests** creates the licenses required and starts a license server, then it runs the tests, reporting the results on the screen.

3.2.1 The Tests

rlmtests performs 2 categories of tests:

- checkout performance tests
- server capacity tests

The checkout performance tests consist of 6 separate tests, named A through F. Each of these tests does performance measurements in different scenarios, as shown in the table below.

Test	Server Environment	Test Details	Notes
A	No clients connected	One rlm_init() followed by a loop of rlm_checkout() calls.	The server must maintain the total number of checkout contexts over the course of this test.
B	No clients connected	One rlm_init() followed by a loop of rlm_checkout()rlm_checkin() calls.	This is the fastest test, because the server is only handling one checkout context at a time.
C	No clients connected	Loop of rlm_init()rlm_checkout()rlm_checkin()rlm_close() calls.	Similar to test B, with the overhead of initialization on each call.
D	300 (default) clients connected	Same as "A"	Similar to A except that the server now has a number of clients that it is managing.
E	300 (default) clients connected	Same as "B"	Similar to B except that the server now has a number of clients that it is managing.
F	300 (default) clients connected	Same as "C"	Similar to C except that the server now has a number of clients that it is managing.

The server capacity test attempts to determine the total number of clients that the server can handle. This test starts up a number of sub-processes which each simulate 1000 clients connected to the server. Sub-processes are started until a checkout fails (or when 50 sub-pro-

cessesall succeed.) During the course of this test, the checkout performance time is reported as each subprocess completes.

3.2.2 The Test Environment

rlmtests runs the client and server on the same machine by default. However, you can run the server on a separate machine from the test client by specifying the -s option on the machine that is to run the server, and the -a hostname option on the client machine to specify the server machine name.

Note that **rlmtests** depends on having the RLM, ISV server, rlmutil, and rlmsign binaries in the same directory as **rlmtests**. This will be the case when you install the License Administration bundle. If you wish to run **rlmtests** from another directory, be sure to copy the other binaries as well.

rlmtests options

rlmtests usage is as follows:

```
rlmtests [-a host] [-c loopcount] [-d] [-h] [-l static#] [-p port#] [-s]
[subprocess_client_count]
```

Where:

- **-a host** - Run the client side only against the server on machine host. Note that the server must be already running on host.
- **-c loopcount** - Controls the number of iterations for the loops for tests A-F. Default is 12000. (Note that not all the loops use this count, but all are scaled in proportion to this number).
- **-d** - Turn on debugging output.
- **-h** - Print help text and exit.
- **-l static#** - Use static# simulated clients for tests D-F (300 default).
- **-p port#** - Use port# for the server port (should be specified on both server and client side if running on different machines). The default port # is 30000.
- **-s** - Run as the server side only.
- **subprocess_client_count** - The number of simulated clients which each subprocess will run in the server capacity tests. The maximum value is 2000.

3.3 Reportlog File Format

The RLM servers create a reportlog in three formats, selectable by the License Administrator. The 3 formats are “std”, “small”, and “detailed”.

Note that prior to RLM v9.3 there was no “detailed” format on Windows; specifying “detailed” would produce a standard report output. Starting in RLM v9.3, the detailed report will be produced, but all fractional seconds fields will be 0.

The formats differ only in the contents of the checkout and check-in records, as described below. Common to all formats are the classes of data logged, and the data for all except check-in and checkout records.

The reportlog format has a version number in the start record. All data described applies to all versions of the reportlog, except where indicated.

The data is:

- authentication data
- check-in
- checkout
- dequeue
- dynamic reservation created/removed (added in RLM v12.2)
- isv-specific data
- license denial
- license in use (added in RLM/reportlog v10.0)
- log file start
- log file end (can correspond to a switch to a new logfile)
- meter transaction data (added in RLM v9.3)
- periodic timestamps
- queue
- roam extend
- server shutdown
- server time jumped
- server reread of license/option file
- support for a product (feature) - one for each license pool at log file start time
- temporary license creation/removal (added in v14.1 – **RLM Cloud servers only**)

Note In every reportlog entry which contains a username or a hostname, the field is presented without quotes. However, if the corresponding value is empty, a pair of double-quotes (“”) will be placed in the reportlog where the username or hostname would be.

A reportlog consists of the following data:

- log file start records
- one PRODUCT support line for each supported product
- one license in use record for each currently-checked-out license
- all license activity data, including checkouts, check-ins, AUTH records, etc.
- one license in use record for each currently-checked-out license
- optional SWITCH line
- END line
- final AUTH line

The format of the data logged is described in the following sections.

3.3.1 Authentication data

All formats:

AUTH section *signature*

This line specifies the authentication signature (*signature*) for the preceding data in the file. If any of the data since the last AUTH line is modified, the authentication value will no longer be

correct. This would typically be used by an ISV if they are using the report log data for post-use billing. The ISV can run a utility in order to verify the authentication records in a report log. Report writers can (and should) ignore this line.

BADAUTH (x errors, y sessions, z authentication sections)

When rlmanon processes a reportlog with AUTH errors, it rewrites the file with all bad AUTH records, and places this line in each section. Prior to RLM v12.3, rlmanon simply stopped processing when a bad AUTH section was encountered. In v12.3, the entire report log will be processed, but all sections will have bad AUTH sections, allowing you to process the anonymized report log but not verify the integrity of the data in it.

3.3.2 check-in

"standard" format:

IN *why product version user host "isv_def" count cur_use cur_resuse server_handle* mm/dd hh:mm:ss

"detailed" format:

IN *why product version user host "isv_def" count cur_use cur_resuse server_handle* mm/dd hh:mm:ss.tenths_of_msec

"small" format:

IN *why count server_handle* hh:mm

The *server_handle* parameter is a hex number. All other numbers are decimal.

The *isv_def* field contains the value of the optional ISV-defined field.

The *why* parameter is one of:

***why* value**	**Reason**
1	"Normal" check-in by application
2	Application exited, automatic check-in
3	License removed by (rlmremove) utility
4	License removed by server after timeout
5	License hold/minimum checkout period expired
6	Client requested license dequeue
7	Portable hostid removed
8	Failed host back up
9	Server lost its transferred licenses
10	Meter ran out of count during a periodic decrement
11	Client failed to send heartbeat within "promise" interval.
12	Temporary License expired (**RLM Cloud only**)
13	Temporary license returned (**RLM Cloud only**)

Note *why* value 12 is new in RLM v14.1, on RLM Cloud only.

The *cur_use* and *cur_resuse* fields indicate the current number of free licenses in use and reservations in use after this check-in.

The *ver* parameter will be the version requested, not the version of the license which satisfied the request. To find the version of the license that satisfied the request, check the version field in the matching checkout request referenced by "server_handle".

Note On Windows, *tenths_of_msec* will always be 0.

3.3.3 checkout

"standard" format:

OUT product version pool# user host "isv_def" count cur_use cur_resuse server_handle share_handle process_id "project" "requested product" "requested version" mm/dd hh:mm:ss

"detailed" format:

OUT *product version pool# user host "isv_def" count cur_use cur_resuse server_handle share_handle process_id "project" "requested product" "requested version"* mm/dd hh:mm:ss.tenths_of_msec *"client_machine_os_info" "application argv0"* roam_days roam_handle client-ip-address

"small" format:

OUT *product version user host "isv_def" count server_handle share_handle* hh:mm

---

Note The *project* field will contain the contents, if any, of the RLM_PROJECT environment variable of the application that checked out the license. This project name has a maximum of 32 characters.

The *isv_def* field contains the value of the optional ISV-defined field.

The *cur_use* and *cur_resuse* fields indicate the current number of free licenses in use and reservations in use after this checkout.

The *server_handle*, *share_handle*, and *process_id* parameters are hex numbers. All other numbers are decimal.

The *share_handle* field contains the value of the handle of a shared license.

Note The *share_handle* is the handle of the first license that was checked out in this group of shared licenses. It is possible (and perhaps even likely) that the handle associated with the group will change if the first license is checked in before other shared licenses are checked in. In this case, new checkouts will specify a *share_handle* of a different license in the group of shared licenses.

The *requested product* and *requested version* fields represent the requested product and version in the application's checkout call. In the case of a token-based license, the product (or products) actually checked out will differ, and *requested product* and *requested version* provide what the application actually requested.

The *process_id* field is the PID of the process requesting the license. The PID will be -1 (ffffffff), when the server logs a roaming "checkout" at startup time.

The *client_machine_os_info* field is a combination of the platform type and OS version running on the client machine. This string is a maximum of 41 bytes long, and is in the format:

"rlm_platform_name os_version"

For example:

> "x64_w42 5.1" - a windows system running Windows XP.

The *application argv0* field is the argv[0] of the product requesting the license.

The *roam_days* field is the (hex) number of days for which the license will roam PLUS 1. This will appear on the initial checkout of a roaming license as well as on the subsequent checkout when the server checks out an already-roaming license. Beginning in rlm v11.1, roam_days is one greater than it had been in prior versions, so a value of 2 means a license roaming for 1 day (i.e., RLM_ROAM set to 1), meaning until the end of tomorrow.

On Windows, *tenths_of_msec* will always be 0.

The *roam_handle* field is the server handle (in hex) of a roaming license re-checkout. This field will be non-zero when the server checks out an already-roaming license (as it does when starting up).

The client-ip-address is new in reportlog v14.1. It is of the form a.b.c.d If the client's IP address is unknown, it will appear as 0.0.0.0.

3.3.4 dequeue

"standard" format:

DEQUE *why product version user host "isv_def" count server_handle* mm/dd hh:mm:ss

"detailed" format:

DEQUE *why product version user host "isv_def" count server_handle* mm/dd hh:mm:ss.tenths_of_msec

"small" format:

DEQUE *why count server_handle* hh:mm

The *server_handle* parameter is a hex number. All other numbers are decimal.

On Windows, *tenths_of_msec* will always be 0.

Note The dequeue record is identical to the check-in record except the keyword is "DEQUE" rather than "IN". A dequeue record is generated on all other servers when a client is granted a license on one server.

3.3.5 dynamic reservation

All formats:

DYNRES [create | remove] *user host license-pool count "string"* mm/dd hh:mm:ss

All numbers are decimal.

string is the identifier used to create the reservation.

The DYNRES record is new in v12.2.

3.3.6 isv-specific data

All formats:

log mm/dd hh:mm:ss isv-specific-data-here

Individual ISVs can log unformatted data to the report log. This data appears in a "log" record.

3.3.7 license denial

"standard" and "small" formats:

DENY *product version user host "isv_def" count why last_attempt pid* mm/dd hh:mm

"detailed" format:

DENY *product version user host "isv_def" count why last_attempt pid* mm/dd hh:mm:ss.tenths_of_msec

The *why* parameter is an RLM_LICENSE error status return (RLM_EL_xxxx) as documented in Section 3.6.

The *last_attempt* parameter is 0 if the application will attempt another checkout, or non-zero if this is the last attempt it will make to check the license out. Thus, denials with *last_attempt* set to 0 are not "true" denials of the license to the application, they are simply denials of the license at this license server. A report writer should only report application license denials when *last_attempt* is set to a non-zero value.

The *pid* field is the process's PID, in hex. The *pid* field is new in reportlog v11.3.

The *isv_def* field contains the value of the optional ISV-defined field.

On Windows, *tenths_of_msec* will always be 0.

3.3.8 license in use

All formats:

INUSE *product version pool# user host "isv_def" count server_handle share_handle process_id* mm/dd hh:mm:ss

The *isv_def* field contains the value of the optional ISV-defined field.

The *server_handle* and *process_id* parameters are hex numbers. All other numbers are decimal.

The *process_id* field is the PID of the process requesting the license.

License in use records are new in RLM v10.0 and appear both at the beginning and the end of the report log if any licenses are in use at that time.

The *share_handle* field contains the value of the handle of a shared license. This field is new in reportlog v12.2.

3.3.9 log file start

All formats:

- SWITCH from old-reportlog-name (new in v14.0, not authenticated)
- RLM Report Log Format *d*, version *x.y authentication flag*
- REPROCESSED with rlmanon vx.y
- ISV: <isvname>, RLM version a.b BLc
- <several lines of header text>
- START *hostname* mm/dd/yyyy hh:mm
- TIMEZONE minutes-west-of-UTC daylight rules # readable version of data
- LICENSE FILE *filename*

The *d* in the first line is the format: 0 for "std", 1 for "small", and 2 for "detailed" *x.y* is the reportlog version. Reportlog v1.0 corresponds to RLM version 1.0. reportlog v1.1 corresponds to RLM v1.1. The *authentication flag* is either blank (no authentication) or the string ", **authenticated**". All report logs are authenticated.

The second line will be present if the *rlmanon* utility was used to anonymize the reportlog data. The version of rlmanon corresponds to the RLM version. *rlmanon* first appeared in RLM v4.0, however while this was added in RLM v4.0, this line can appear in any reportlog version, since

rlmanon can process any version of reportlog. Note that this line can be repeated if multiple runs of rlmanon were made on the same log file.

The third line displays the ISV name, and the RLM software version of the ISV server (a.b BLc, BL means "Build").

In general, numeric data is in decimal format. The 3 exceptions to this are *server_handle*, *share_handle*, and *process_id* parameters which are always hex numbers.

The TIMEZONE line is new in RLM v12.3 This represents the server's timezone data: the first number is the minutes west of UTC time; the second number indicates whether daylight savings rules ever apply in this timezone. The data after the # is a human-readable version of the timezone information.

There will be one LICENSE FILE line(s) for each license file which the server is processing.

Note that the "SWITCH from" line is not included in the report authentication.

3.3.10 log file end

All formats:

SWITCH to *filename* (if an rlmswitch was done)

END mm/dd/yyyy hh:mm

3.3.11 meter decrement

All formats:

METER_DEC *license_handle meter_counter amount_decremented* mm/dd hh:mm:ss[.tenths_of_msec]

A *meter decrement* record will immediately follow a checkout record for a metered license. In addition, an additional *meter decrement* record will appear periodically when the meter is decremented for this product. The *license handle* is a hex value; both *meter counter* and *amount decremented* are decimal.

The format is the same for all reportlog types, with the exception of the time - in the detailed reportlog format, tenths of milliseconds are added.

3.3.12 periodic timestamp

All formats:

mm/dd/yyyy hh:mm

Timestamps are performed by the ISV servers every night just after midnight.

Timestamps are added to the log file every 30 minutes.

3.3.13 queue

"standard" format:

QUE *product version user host "isv_def" count server_handle "project" "requested product" "requested version" mm/dd hh:mm:ss*

"detailed" format:

QUE *product version user host "isv_def" count server_handle "project" "requested product" "requested version"* mm/dd hh:mm:ss.tenths_of_msec

"small" format:

QUE *product version user host "isv_def" count server_handle* hh:mm

The *server_handle* parameter is a hex number. All other numbers are decimal.

The *project* field will contain the contents, if any, of the *RLM_PROJECT* environment variable of the application that checked out the license. This project name has a maximum of 32 characters.

The *isv_def* field contains the value of the optional ISV-defined field.

On Windows, *tenths_of_msec* will always be 0.

If the queued license becomes available, a checkout record will be logged with the same handle. If the client abandons the checkout, however, no other records will be logged.

3.3.14 roam extend

The roam extend record has only a single format, for all 3 reportlog formats:

ROAM_EXTEND *product version pool# user host "isv_def" #days_extended server_handle process_id* mm/dd hh:mm:ss

The *isv_def* field contains the value of the optional ISV-defined field.

The *#days_extended* parameter is the # of days which the roam was extended. So, if a license was originally roaming for 6 days, then extended to 10 days, this parameter will be 4, independent of which day the extension was done.

The *server_handle* and *process_id* parameters are hex numbers. All other numbers are decimal.

The *process_id* field is the PID of the process requesting the license. Note that if the pid is -1 (ffffffff), this means that the roam was extended by the RLM web interface when the client was no longer connected to the server.

Roam extend records are new in RLM v10.0.

3.3.15 server shutdown

All formats:

SHUTDOWN *user host* mm/dd hh:mm:ss

3.3.16 server time jump

All formats:

TIMEJUMP [+ | -]minutes mm/dd hh:mm:ss

The ISV server logs this when the time changes by more than 15 minutes on subsequent passes thru its main loop. The main loop executes whenever there is a message to process, but not less frequently than once/minute.

3.3.17 server reread of license/option file

All formats:

REREAD *user host* mm/dd hh:mm:ss

3.3.18 support for a product

There will be one record per license pool for each product served. These lines come immediately after a START or REREAD record. Note that there is not a one-to-one correspondence between the **support** records and LICENSE lines in the license file.

All formats:

PRODUCT *name version pool# count #reservations soft_limit "hostid" "contract" "customer" "issuer" "line_item" "options" share max_share type named_user_count meter_type meter_counter meter_initial_decrement meter_period meter_period_decrement*

- *count* is the number of free licenses (i.e., non-reserved licenses)
- *#reservations* is the number of reserved licenses (not generally available).
- *pool#* is an internal server pool identifier. This number appears in checkout records in some formats.
- *line_item* is the contents of the product's _line_item field, used for mapping license product names to actual purchased products.
- *meter_type* is always 0 in RLM v9.3. Other values may be defined in later versions. A non-zero value in meter_counter indicates a metered license.
- *meter_counter* is the counter which is used for this product.
- *meter_initial_decrement* is the amount to be decremented from the meter when a license is checked out.
- *meter_period* is the number of minutes before an additional decrement is performed (0 means no periodic decrements)
- *meter_period_dec* is the amount to be decremented from the meter each meter_period minutes.

3.3.19 Temporary license creation/removal

All formats:

TEMP [create | remove | restart | expired] *product version license-pool user host "isv_def" expdate exptime server_handle* mm/dd hh:mm:ss

The *server_handle* is a hex number. All other parameters are strings.

- *server_handle* is the handle of the associated checkout. Temporary licenses are created after a successful checkout.
- *expdate* is the expiration date (local time on the server) of the temporary license.
- *exptime* is the expiration time (local time on the license server) of the temporary license.

On a server restart, there will be a *TEMP* restart record with (possibly) a different license-pool value if the server's licenses have been modified before the restart. This is not a new temporary license, it is essentially a checkout of the same temporary license which was created earlier.

When a temporary license is refreshed, a new *TEMP create* record will be logged, with the same time as the original *TEMP create* record (or INUSE record, if the server preformed a reread). This new record does not represent an additional checkout, however the *expdate* and *exptime* fields will be updated. For example, a server had a temporary license and performed a reread:

Before the reread, a temp license was created that expired at 15:55:

```
TEMP create test2 2.0 2 matt zippy "" 2020-05-19 15:55 41 05/19 15:45:43
```

After the reread, the in-use temp license is logged, then later refreshed to 15:58:

```
INUSE test2 2.0 2 matt zippy "" 1 41 41 4a07 05/19 15:45:43
TEMP create test2 2.0 2 matt zippy "" 2020-05-19 15:58 41 05/19 15:45:43
```

The TEMP record is new in v14.1 and appears on RLM Cloud servers only.

3.3.20 Reportlog version change history

V14.1 reportlog changes
The OUT record in detailed format now includes the client's IP address
The TEMP_CREATED record is new (RLM Cloud servers only)
The check-in why values 12 and 13 are new (RLM Cloud servers only)

V14.0 reportlog changes
SWITCH from old-reportlog added to header

V12.3 reportlog changes
The TIMEZONE record is added to the header information.
The BADAUTH record is added by rlmanon when processing logs with AUTH errors.

V12.2 reportlog changes
The share handle is added to INUSE records
The DYNRES record is added.

V12.1 reportlog changes
The Server Time Jump (TIMEJUMP) record is added.

V11.3 reportlog changes
The DENY records now include the process's PID.

V11.1 reportlog changes
roam_days is now one greater than prior versions (and one greater than the value of
RLM_ROAM). roam_days == 2 now means roaming until midnight tomorrow.

v10.0 reportlog changes
All logfiles will now log all licenses currently in use both at the start and the end of the reportlog, with the new "INUSE" record.
The ROAM_EXTEND record is added.

v9.3 reportlog changes
Windows systems will now produce a detailed reportlog, however, all fractional seconds fields will be 0.
meter decrement records added.
metering parameters added to the support record

The logfile is timestamped every 30 minutes

v9.0 reportlog changes
roam days and roam handle added to the OUT record for detailed format.
All fields which have a username or a hostname will now contain only an empty pair of double-quotes (“”) if the corresponding value is empty. This only happens when rlm cannot determine the username or hostname on the system using the standard system calls.

v8.0 reportlog changes
client machine OS version added to the OUT record for detailed format.
application argv[0] added to the OUT record for detailed format.

v5.0 reportlog changes
requested product and requested version added to QUE record for std and detailed formats.
requested product and requested version will always be empty strings in OUT records that result from a de-queueing of a previously queued request.
options, share, max_share, type, and named_user_count added to PRODUCT support record
Portable hostid removed - status 7 - added to checkin reasons.

v4.0 reportlog changes
REPROCESSED line added for rlmanon. Note that this line can appear in any version reportlog, since rlmanon can process all reportlog versions. Report writers can ignore this line, it is in the file for informational purposes only.
“detailed” format added to DENIAL records to add seconds and fractional seconds.
REREAD END records were not generated prior to v4.0. This record is removed.
New denial status -45 (RLM_EL_NOT_NAMED_USER) added
All report logs are authenticated.

v3.0 reportlog changes
NOTE: the v3.0 reportlog is incorrectly logged as v2.0 in the “RLM Report Log” line. The version in the ISV line will correctly indicate that it is a v3.0 logfile.
line_item added to PRODUCT records
“, authenticated” added to the end of the ‘RLM Report Log’ (first) record if this reportlog is authenticated
AUTH records added (for authenticated reportlogs)

v2.0 reportlog changes
LICENSE FILE line added in header section
isv-specific data (“log” records) added
cur_use and cur_resuse fields added to IN records (standard and detailed formats)
process_id, requested_product, and requested_version added to OUT records (standard and detailed formats)
hostid and pool# added to PRODUCT records

3.4 RLM Host IDs

RLM supports several different kinds of identification for various computing environments, as well as some generic identification which are platform-independent.

Table 3.1. RLM's host identification (hostid) types are:

Type	Meaning	Example	Notes
ANY	runs anywhere	hostid=ANY	
DEMO	runs anywhere for a demo license	hostid=DEMO	
serial number	runs anywhere	hostid=sn=123-456-789	Used to identify a license, any string up to 64 characters long.
disksn	Hard Disk hardware serial number	hostid=disksn=WDWX60AC94686O	Windows only.
32	32-bit hostid, native on Unix, non X86 based platforms	hostid=10ac0307	This is the volume serial number on windows, and is not recommended.
ip (or internet)	TCP/IP address	hostid=ip=192.156.1.3	Always printed as "ip=". Wildcards allowed.
ether	Ethernet MAC address	hostid=ether=00801935f2b5	Always printed without leading "ether=".
rlmid1	External key or dongle	hostid=rlmid1=9a763f21	External key or dongle.
uuid	BIOS uuid	hostid=uuid=699A4D56-5BBE-1C83-DE3C-27A8BEB8011A	Windows only.
user	User name	hostid=USER=joe	Case-insensitive
host	Host name	hostid=host=melody	
gc	*Google Compute Engine*	*hostid=gc=3797742226458986650-.k6qt9v5h38w2adwqgc9fdhdf3w0m761p*	*Linux only, DEPRECATED.*

Warning The Google Compute host ID is deprecated as of v14.2 and should not be used.

To determine the hostid of a machine, use the hostid type from the table above as input to the rlmhostid command:

```
rlmutil rlmhostid hostid type
```

For example:

```
rlmutil rlmhostid 32
```

or

```
rlmutil rlmhostid internet
```

When an application requests a license from a license server, it will transmit the hostid information from the local machine to the license server, so that the server can process node-locked licenses without additional queries to the application. The application will transmit a maximum of 25 different hostids: * one 32-bit hostid, if present on this platform * 1 or 2 Disk serial numbers (windows only) * up to 3 IP address * up to 5 ethernet MAC addresses (ether=) * up to 6 RLMID portable hostids * a UUID hostid, if present * a minimum of 3 ISV-defined hostids (usually more, but guaranteed to be at least 3)

Note In RLM v15.0, the server will add the client's external IP address to the list of IP addresses supplied by the client. This IP address is not added if it is the same as one of the client-supplied IP addresses.

3.5 IPv6 Considerations

Beginning in RLM v11.0, RLM supports IPv6 on Windows and Linux.

Linux support operates correctly on all RLM builds.

Windows support, however, is limited to the x86_w3//x64_w3 and x86_w4//x64_w4, since earlier versions of the compiler do not support IPv6. This can lead to some inconsistent behavior if different RLM versions are used on IPv6 networks.

Let's assume that you have an IPv6 network, and both client and server are running on IPv6-capable machines. If your ISV's application is built with one of the _w3 kits, it will attempt to communicate to an IPv6 address. However, if either RLM or the ISV server is built with the _w1 or _w2 kit, it will not be able to bind an IPv6 address, and the connection to the server will fail.

If this is the case, you can do one or 2 things:

1. Use an IPv4 address in the SERVER line in place of the hostname, or
2. Ensure that you are running _w3//_w4 versions of RLM and the ISV server.

If you have received a _w1 or _w2 version of an ISV server, you must use technique #1 above, until your ISV can supply you with an ISV server built with a _w3//_w4 kit. If your ISV supplies a settings file, however, you only need to make sure that the version of the RLM binary you are running is a _w3//_w4 version.

To determine the version of the RLM kit a server was built with, you can look at the "Server Architecture:" line in the first few lines of the debug log file, as in this example:

```
05/21 14:19 (rlm) RLM License Server Version 11.3BL1
Copyright (C) 2006-2015, Reprise Software, Inc. All rights reserved.

05/21 14:19 (rlm) License server started on aztec
05/21 14:19 (rlm) Server architecture: x86_w3
```

This example is a copy of RLM built with the x86_w3 kit, so it will operate correctly on IPv6 networks.

You can view the debug log from the ISV server to determine which RLM kit was used to build it as well.

3.6 Status Values

3.6.1 rlm_stat() returns general RLM_HANDLE errors. These are:

0	0	Success.
RLM_EH_NOHANDLE	-101	No handle supplied to call.
RLM_EH_READ_NOLICENSE	-102	Can't read license data.
RLM_EH_NET_INIT	-103	Network (msg_init()) error.
RLM_EH_NET_WERR	-104	Error writing to network.
RLM_EH_NET_RERR	-105	Error reading from network.
RLM_EH_NET_BADRESP	-106	Unexpected response.
RLM_EH_BADHELLO	-107	HELLO message for wrong server.
RLM_EH_BADPRIVKEY	-108	Error in private key.
RLM_EH_SIGERROR	-109	Error signing authorization.
RLM_EH_INTERNAL	-110	Internal error.
RLM_EH_CONN_REFUSED	-111	Connection refused at server (this can also happen if you have a bad TCP/IP address in your local database).
RLM_EH_NOSERVER	-112	No server to connect to.
RLM_EH_BADHANDSHAKE	-113	Bad communications handshake.

RLM_EH_CANTGETETHER	-114	Can't get ethernet address.
RLM_EH_MALLOC	-115	malloc() error.
RLM_EH_BIND	-116	bind() error.
RLM_EH_SOCKET	-117	socket() error.
RLM_EH_BADPUBKEY	-118	Error in public key.
RLM_EH_AUTHFAIL	-119	Authentication failed.
RLM_EH_WRITE_LF	-120	Can't write new license file.
RLM_EH_DUP_ISV_HID	-122	ISV-defined hostid already registered.
RLM_EH_BADPARAM	-123	Bad parameter passed to RLM function.
RLM_EH_ROAMWRITEERR	-124	Roam File write error.
RLM_EH_ROAMREADERR	-125	Roam File read error.
RLM_EH_HANDLER_INSTALLED	-126	Heartbeat handler already installed.
RLM_EH_CANTCREATELOCK	-127	Can't create 'single' lockfile.
RLM_EH_CANTOPENLOCK	-128	Can't open 'single' lockfile.
RLM_EH_CANTSETLOCK	-129	Can't set lock for 'single'.
RLM_EH_BADRLMLIC	-130	Bad/missing/expired RLM license.
RLM_EH_BADHOST	-131	Bad hostname in license file or port@host.
RLM_EH_CANTCONNECTURL	-132	Can't connect to specified URL (activation).
RLM_EH_OP_NOT_ALLOWED	-133	Operation not allowed on server. The status, reread, shutdown, or remove command has been disabled for this user.
RLM_EH_ACT_BADSTAT	-134	Bad status return from Activation server.
RLM_EH_ACT_BADLICKEY	-135	Activation server built with incorrect license key.
RLM_EH_ACT_BAD_HTTP	-136	Error in HTTP transaction with Activation server.
RLM_EH_DEMOEXISTS	-137	Demo already created on this system.
RLM_EH_DEMOWRITEERR	-138	Demo install file write error.
RLM_EH_NO_DEMO_LIC	-139	No "rlm_demo" license available.
RLM_EH_NO_RLM_PLATFORM	-140	RLM is unlicensed on this platform.
RLM_EH_EVAL_EXPIRED	-141	The RLM evaluation license compiled into this binary has expired.
RLM_EH_SERVER_REJECT	-142	Server rejected (too old).
RLM_EH_UNLICENSED	-143	Unlicensed RLM option.
RLM_EH_SEMAPHORE_FAILURE	-144	Semaphore initialization failure.
RLM_EH_ACT_OLDSERVER	-145	Activation server too old (doesn't support encryption).
RLM_EH_BAD_LIC_LINE	-146	Invalid license line in LF.
RLM_EH_BAD_SERVER_HOSTID	-147	Invalid hostid on SERVER line.
RLM_EH_NO_REHOST_TOP_DIR	-148	No rehostable hostid top-level dir.
RLM_EH_CANT_GET_REHOST	-149	Cannot get rehostable hostid.
RLM_EH_CANT_DEL_REHOST	-150	Cannot delete rehostable hostid.
RLM_EH_CANT_CREATE_REHOST	-151	Cannot create rehostable hostid.
RLM_EH_REHOST_TOP_DIR_EXISTS	-152	Rehostable top directory exists.
RLM_EH_REHOST_EXISTS	-153	Rehostable hostid exists.
RLM_EH_NO_FULFILLMENTS	-154	No fulfillments to revoke.
RLM_EH_METER_READERR	-155	Meter read error.
RLM_EH_METER_WRITEERR	-156	Meter write error.
RLM_EH_METER_BADINCREMENT	-157	Bad meter increment command.

RLM_EH_METER_NO_COUNTER	-158	Can't find counter in meter.
RLM_EH_ACT_UNLICENSED	-159	Activation Unlicensed.
RLM_EH_ACTPRO_UNLICENSED	-160	Activation Pro Unlicensed.
RLM_EH_SERVER_REQUIRED	-161	Counted license requires server.
RLM_EH_DATE_REQUIRED	-162	REPLACE license requires date.
RLM_EH_NO_METER_UPGRADE	-163	METERED licenses can't be UPGRADED.
RLM_EH_NO_CLIENT	-164	Disconnected client data can't be found.
RLM_EH_NO_DISCONN	-165	Operation not allowed on disconnected handle.
RLM_EH_NO_FILES	-166	Too many open files.
RLM_EH_NO_BROADCAST_RESP	-167	No response to broadcast message.
RLM_EH_NO_BROADCAST_HOST	-168	Broadcast response didn't include hostname.
RLM_EH_SERVER_TOO_OLD	-169	Server too old for disconnected operations.
RLM_EH_BADLIC_FROM_SERVER	-170	License from server doesn't authenticate.
RLM_EH_NO_LIC_FROM_SERVER	-171	No License returned from server.
RLM_EH_CACHEWRITEERR	-172	Client Cache File write error.
RLM_EH_CACHEREADERR	-173	Client Cache File read error.
RLM_EH_LIC_WITH_NEW_KEYWORDS	-174	License returned from server has keywords I don't understand.
RLM_EH_NO_ISV	-175	Unknown ISV name.
RLM_EH_NO_CUSTOMER	-176	Unknown Customer name.
RLM_EH_NO_SQL	-177	Cannot open MySQL database (RLM Cloud only).
RLM_EH_ONLY_LOCAL_CMDS	-178	Only local command-line commands allowed.
RLM_EH_SERVER_TIMEOUT	-179	Server timeout on read.
RLM_EH_NONE_SIGNED	-180	rlmsign did not sign any licenses (warning).
RLM_EH_DUP_XFER	-181	Duplicate disconnected transfer.
RLM_EH_BADLOGIN	-182	Bad/No login credentials to server.
RLM_EH_WS_NOSUPP	-183	Function not supported with web services.
RLM_EH_NOFUNC	-184	Function not available.
RLM_EH_TOOMUCHJSON	-185	JSON reply too big.
RLM_EH_NOLICFROMSERV	-186	Server returned no temp license.
RLM_EH_TEMPFROMCLOUD	-187	Temporary licenses come from RLM Cloud servers only.
RLM_EH_NOTTEMP	-188	License is not a temporary license.
RLM_EH_NOLICENSE	-189	No license supplied to call.
RLM_EH_NOTEMPFROMLOCAL	-190	Local license can't be converted to temp license.
RLM_EH_NOHTTPSSUPPORT	-191	ActPro HTTPS support not available.
RLM_EH_NOHTTPSDATA	-192	No returned data from HTTPS.
RLM_EH_NOTTHISHOST	-193	Wrong Host.
RLM_EH_NOTRANSBIN	-194	Translated binaries not supported (mac).

3.6.2 In addition, rlm_activate()/rlm_act_request() will return the following

errors:

RLM_ACT_BADPARAM	-1001	Unused – RLM_EH_BADPARAM returned instead.
RLM_ACT_NO_KEY	-1002	No activation key supplied.
RLM_ACT_NO_PROD	-1003	No product definition exists.
RLM_ACT_CANT_WRITE_KEYS	-1004	Can't write keyf table.
RLM_ACT_KEY_USED	-1005	Activation key already used.
RLM_ACT_BAD_HOSTID	-1006	Missing hostid.
RLM_ACT_BAD_HOSTID_TYPE	-1007	Invalid hostid type.
RLM_ACT_BAD_HTTP	-1008	Bad HTTP transaction. Note: unused after v3.0BL4.
RLM_ACT_CANTLOCK	-1009	Can't lock activation database.
RLM_ACT_CANTREAD_DB	-1010	Can't read activation database.
RLM_ACT_CANT_WRITE_FULFILL	-1011	Can't write licf table.
RLM_ACT_CLIENT_TIME_BAD	-1012	Clock bad on client system (not within 7 days of server).
RLM_ACT_BAD_REDIRECT	-1013	Can't write licf table.
RLM_ACT_TOOMANY_HOSTID_CHANGES	-1014	Too many hostid changes for refresh-type activation.
RLM_ACT_BLACKLISTED	-1015	Domain on blacklist for activation.
RLM_ACT_NOT_WHITELISTED	-1016	Domain not on activation key whitelist.
RLM_ACT_KEY_EXPIRED	-1017	Activation key expired.
RLM_ACT_NO_PERMISSION	-1018	HTTP request denied (this is a setup problem).
RLM_ACT_SERVER_ERROR	-1019	HTTP internal server error (usually a setup problem).
RLM_ACT_BAD_GENERATOR	-1020	Bad/missing generator file (ActPro).
RLM_ACT_NO_KEY_MATCH	-1021	No matching activation key in database.
RLM_ACT_NO_AUTH_SUPPLIED	-1022	No proxy authorization supplied.
RLM_ACT_PROXY_AUTH_FAILED	-1023	Proxy authentication failed.
RLM_ACT_NO_BASIC_AUTH	-1024	No basic authentication supported by proxy.
RLM_ACT_GEN_UNLICENSED	-1025	Activation generator unlicensed (ISV_mklic).
RLM_ACT_DB_READERR	-1026	Activation database read error (ActPro).
RLM_ACT_GEN_PARAM_ERR	-1027	Generating license - bad parameter.
RLM_ACT_UNSUPPORTED_CMD	-1028	Unsupported command to license generator.
RLM_ACT_REVOKE_TOOLATE	-1029	Revoke command after expiration.
RLM_ACT_KEY_DISABLED	-1030	Activation key disabled.
RLM_ACT_KEY_NO_HOSTID	-1031	Key not fulfilled on this hostid.
RLM_ACT_KEY_HOSTID_REVOKED	-1032	Key revoked on this hostid.
RLM_ACT_NO_FULFILLMENTS	-1033	No fulfillments to remove.
RLM_ACT_LICENSE_TOOBIG	-1034	Generated license too long.
RLM_ACT_NO_REHOST	-1035	Counted licenses can't be rehostable.
RLM_ACT_BAD_URL	-1036	License Generator not found on server.

RLM_ACT_NO_LICENSES	-1037	RLM Cloud: No licenses found.
RLM_ACT_NO_CLEAR	-1038	Unencrypted requests not allowed.
RLM_ACT_BAD_KEY	-1039	Bad activation key (illegal char).
RCC_CANT_WRITE_FULFILL	-1040	RLM Cloud: Can't write licf table.
RCC_PORTAL_CANT_WRITE_FULFILL	-1041	RLM Cloud: Can't write licf table.
RLM_ACT_KEY_TOOMANY	-1042	Insufficient count left in activation key.
RLM_ACT_SUB_BADTYPE	-1043	Subscription license not Nodelocked or Single.
RLM_ACT_CONTACT_BAD	-1044	Contact information supplied is bad.

3.6.3 rlm_license_stat() returns RLM_LICENSE errors and status. These are:

Status	Value	Meaning	Full Description
0	0	Success	
RLM_EL_NOPRODUCT	-1	No authorization for product	rlm_checkout() did not find a product to satisfy your request.
RLM_EL_NOTME	-2	Authorization is for another ISV	The license you are requesting is in the license file, but it is for a different ISV.
RLM_EL_EXPIRED	-3	Authorization has expired	The only license available has expired. This error will only be returned for local license lines, never from a license server.
RLM_EL_NOTTHISHOST	-4	Wrong host for authorization	The hostid in the license doesn't match the hostid of the machine where the software is running.
RLM_EL_BADKEY	-5	Bad key in authorization	The signature in the license line is not valid, i.e. it does not match the remainder of the data in the license.
RLM_EL_BADVER	-6	Requested version not supported	Your application tried to check out a license at a higher version than was available, e.g., you specified v5, but the available license is for v4.
RLM_EL_BADDATE	-7	Bad date format - not permanent or dd-mm-yy	The expiration, start, or issued date wasn't understood, eg, 316-mar-2010 or 31-jun-2010. You'd probably never see this in the field unless somebody had tampered with the license file.

RLM_EL_TOOMANY	-8	Checkout request for too many licenses	Your checkout request will never work, because you have asked for more licenses than are issued.
RLM_EL_NOAUTH	-9	No license auth supplied to call	This is an internal error.
RLM_EL_ON_EXC_ALL	-10	On excludeall list	The license administrator has specified an EXCLUDEALL list for this product, and the user (host, etc) is on it.
RLM_EL_ON_EXC	-11	On feature exclude list	The license administrator has specified an EXCLUDE list for this product, and the user (host, etc) is on it.
RLM_EL_NOT_INC_ALL	-12	Not on the includeall list	The license administrator has specified an INCLUDEALL list for this product, and you are not on it.
RLM_EL_NOT_INC	-13	Not on the feature include list	The license administrator has specified an INCLUDE list for this product, and you are not on it.
RLM_EL_OVER_MAX	-14	Request would go over license MAX	The license administrator set a license MAX usage option for a user or group. This checkout request would put this user/-group/host over that limit.
RLM_EL_REMOVED	-15	License (rlm)removed by server	A license administrator removed this license using the rlmremove command or the RLM web interface.
RLM_EL_SERVER_BADRESP	-16	Unexpected response from server	The application received a response from the license server which it did not expect. This is an internal error.
RLM_EL_COMM_ERROR	-17	Error communicating with server	This indicates a basic communication error with the license server, either in a network initialization, read, or write call.
RLM_EL_NO_SERV_SUPP	-18	License server doesn't support this	
RLM_EL_NOHANDLE	-19	No license handle	No license handle supplied to an rlm_get_attr_xxx() call or rlm_license_xxx() call.
RLM_EL_SERVER_DOWN	-20	Server closed connection	The license server closed the connection to the application.

RLM_EL_NO_HEARTBEAT	-21	No heartbeat response received	Your application did not receive a response to a heartbeat message which it sent. This would happen when you call rlm_get_attr_health(), or automatically if you called rlm_auto_hb().
RLM_EL_ALLINUSE	-22	All licenses in use	All licenses are currently in use, and the user did not request to be queued. This request will succeed at some other time when some licenses are checked in.
RLM_EL_NOHOSTID	-23	No hostid on uncounted license	Uncounted licenses always require a hostid.
RLM_EL_TIMEDOUT	-24	License timed out by server	Your application did not send any heartbeats to the license server and the license administrator specified a TIMEOUT option in the ISV server options file.
RLM_EL_INQUEUE	-25	In queue for license	All licenses are in use, and the user requested queuing by setting the RLM_QUEUE environment variable.
RLM_EL_SYNTAX	-26	License syntax error	This is an internal error.
RLM_EL_ROAM_TOOLONG	-27	Roam time exceeds maximum	The roam time specified in a checkout request is longer than either the license-specified maximum roaming time or the license administrator's ROAM_MAX_DAYS option specification.
RLM_EL_NO_SERV_HANDLE	-28	Server does not know this license handle	This is an internal server error. It will be returned usually when you are attempting to return a roaming license early.
RLM_EL_ON_EXC_ROAM	-29	On roam exclude list	The license administrator has specified an EXCLUDE_ROAM list for this product, and the user (host, etc) is on it.
RLM_EL_NOT_INC_ROAM	-30	Not on the roam include list	The license administrator has specified an INCLUDE_ROAM list for this product, and you are not on it.

RLM_EL_TOOMANY_ROAMING	-31	Too many licenses roaming already	A request was made to roam a license, but there are too many licenses roaming already (set by the license administrator ROAM_MAX_COUNT option).
RLM_EL_WILL_EXPIRE	-32	License expires before roam period ends	A roaming license was requested, but the only license which can fulfill the request will expire before the roam period ends.
RLM_EL_ROAMFILEERR	-33	Problem with roam file	There was a problem writing the roam data file on the application's computer.
RLM_EL_RLM_ROAM_ERR	-34	Cannot check out rlm_roam license	A license was requested to roam, but the application cannot check out an rlm_roam license.
RLM_EL_WRONG_PLATFORM	-35	Wrong platform for client	The license specifies platforms=xxx, but the application is not running on one of these platforms.
RLM_EL_WRONG_TZ	-36	Wrong timezone for client	The license specifies an allowed timezone, but the application is running on a computer in a different timezone.
RLM_EL_NOT_STARTED	-37	License start date in the future	The start date in the license hasn't occurred yet, e.g., today you try to check out a license containing start=1-mar-2030.
RLM_EL_CANT_GET_DATE	-38	time() call failure	The time() system call failed.
RLM_EL_OVERSOFT	-39	Request goes over	This license checkout causes the license usage to license soft_limit go over it's soft limit. The checkout is successful, but usage is now in the overdraft mode. RLM_EL_OVERSOFT is also returned if you have a misconfigured token-based license and the server has gone into overdraft due to this. See the note in the token-based license restrictions section.

RLM_EL_WINDBACK	-40	Clock setback detected	RLM has detected that the clock has been set back. This error will only happen on expiring licenses.
RLM_EL_BADPARAM	-41	Bad parameter to rlm_checkout() call	This currently happens if a checkout request is made for < 0 licenses.
RLM_EL_NOROAM_FAILOVER	-42	Roam operations not allowed on failover server	A failover server has taken over for a primary server, and a roaming license was requested. Roaming licenses can only be obtained from primary servers. Re-try the request later when the primary server is up.
RLM_EL_BADHOST	-43	bad hostname in license file or port@host	The hostname in the license file is not valid on this network.
RLM_EL_APP_INACTIVE	-44	Application is inactive	Your application is set to the inactive state (with rlm_set_active(rh, 0), and you have called rlm_get_attr_health().
RLM_EL_NOT_NAMED_USER	-45	User is not on the named-user list	You are not on the named user list for this product.
RLM_EL_TS_DISABLED	-46	Terminal server/remote desktop disabled	The only available license has Terminal Server disabled, and the application is running on a Windows Terminal Server machine.
RLM_EL_VM_DISABLED	-47	Running on Virtual Machines disabled	The only available license has virtual machines disabled, and the application is running on a virtual machine.
RLM_EL_PORTABLE_REMOVED	-48	Portable hostid removed	The license is locked to a portable hostid (dongle), and the hostid was removed after the license was acquired by the application.
RLM_EL_DEMOEXP	-49	Demo license has expired	Detached Demo™ license has expired.
RLM_EL_FAILED_BACK_UP	-50	Failed host back up - failover server released license	If your application is holding a license from a failover server, when the main server comes back up, the failover server will drop all the licenses it is serving, and you will get this status.

RLM_EL_SERVER_LOST_XFER	-51	Server lost it's transferred license	Your license was served by a server which had received transferred licenses from another license server. The originating license server may have gone down, in which case, your server will lose the licenses which were transferred to it.
RLM_EL_BAD_PASSWORD	-52	Incorrect password for product	RLM_EL_BAD_PASSWORD is an internal error and won't ever be returned to the client - if the license password is bad, the client will receive RLM_EL_NO_SERV_SUPP.
RLM_EL_METER_NO_SERVER	-53	Metered licenses	Metered licenses only work with with a license require server server.
RLM_EL_METER_NOCOUNT	-54	Not enough count for meter	There is insufficient count in the meter for the requested operation.
RLM_EL_NOROAM_TRANSIENT	-55	Roaming not allowed	Roaming is not allowed on servers with transient hostids, ie, dongles.
RLM_EL_CANTRECONNECT	-56	Can't reconnect to server	On a disconnected handle, the operation requested needed to reconnect to the server, and this operation failed.
RLM_EL_NONE_CANROAM	-57	None of these licenses can roam	The license max_roam_count is set to 0. This will always be the case for licenses that are transferred to another server.
RLM_EL_SERVER_TOO_OLD	-58	Server too old for this operation	In v10, this error means that disconnected operation (rlm_init_disconn()) was attempted on a pre-v10.0 license server.
RLM_EH_SERVER_REJECT	-59	Server rejected client	Either the server is older than the oldest version allowed, or a generic server is used when the client specifies this is not allowed.
RLM_EL_REQ_OPT_MISSING	-60	Required option missing	A required option was set with the rlm_set_req_opt() call, and this string is not part of the license string. This error will only be reported for nodelocked licenses, the server will always report RLM_EL_NO_SERV_SUPP.

RLM_EL_NO_DYNRES	-61	Reclaim can't find dynamic reservation	The license specified to be reclaimed cannot be found.
RLM_EL_RECONN_INFO_BAD	-62	Reconnection info invalid	This is generally an internal error.
RLM_EL_ALREADY_ROAMING	-63	License already roaming on this host	If you attempt to roam N licenses then later N+X licenses, you will receive this error. The original roam must be returned first.
RLM_EL_BAD_EXTEND_FMT	-64	Bad format for RLM_ROAM_EXTEND	RLM_ROAM_EXTEND format is product:date:extension_code.
RLM_EL_BAD_EXTEND_CODE	-65	Bad extend code	Extension code does not verify correctly.
RLM_EL_NO_ROAM_TO_EXTEND	-66	No roaming license to extend	This is an attempt to extend a non-existant roaming license.
RLM_EL_NESTED_ALIAS	-67	Nested aliases	You cannot define an alias in terms of another alias.
RLM_EL_NO_JSON	-68	No JSON in returned message	This is an internal error in web services processing with RLM Cloud.
RLM_EL_BAD_JSON	-69	Bad JSON in returned message	This is an internal error in web services processing with RLM Cloud.
RLM_EL_BADHANDSHAKE	-70	Bad handshake on web services checkout	This is an internal error in web services processing with RLM Cloud.
RLM_EL_HELPER_ERR	-71	rlm_helper error	This is an internal error in web services processing with RLM Cloud.
RLM_EL_PERS_NOT_ON_LIST	-72	Not on list	The user is not on the personal license list.
RLM_EL_PERS_BADPASS	-73	Bad password	The personal user's password is incorrect.
RLM_EL_PERS_INUSE	-74	License in use	The personal license is in use.
RLM_EL_PERS_HANDLE_ERR	-75	Handle error	RLM handle error (personal license)
RLM_EL_NOTTEMP	-76	Not a temp license	License is not a temporary license.
RLM_EL_NOSERVER	-77	No server	No server to connect to return temp license.

3.7 Basic RLM Server Setup for Windows

If you have just purchased a piece of software that uses the Reprise License Manager (RLM), and you have never used RLM before, this guide will walk you through a basic setup of the license manager on the Windows operating system.

This is meant to be only a very basic guide to get you started; more detailed information on any of the items below can be found in the preceding manual. If you have any installation issues or

questions, please contact your software publisher directly for support.

In the rest of this guide we will refer to the software publisher as the ISV (independent software vendor).

3.7.1 What You'll Need

- The RLM server
- An ISV server
- A license file

The RLM server may or may not be provided to you by your ISV, and will be called *rlm.exe* on Windows. If not provided by your ISV, downloads can be found here: https://reprisesoftware.-com/support/admin/

The ISV server and license file must be provided by your ISV. The ISV server will be either a platform-independent settings file, called <isvname>.set, or a binary called <isvname>.exe on Windows. The license file may be a text file with the extension ".lic" (the format recommended by Reprise), but you will need to refer to the files supplied by your ISV to confirm.

Options File (optional)

If you would like to view historical usage information for the software you have purchased, you will also need an options file. Creating one is easy:

1. Create a new text file and rename it "<isvname>.opt" (New Text Document.txt → <isv-name>.opt).
2. Open <isvname>.opt with Notepad and add a REPORTLOG line:

```
REPORTLOG +C:\\RLM\reportlog.log
```

3. Save and close the file.

For more information see Section 1.9.

3.7.2 Installation

Create a folder that the license manager will run from. This can be called anything, but for the purpose of this guide we will name it \RLM and create it directly on the root of C: (C:\RLM). If you have downloaded a kit, or were given an archive (like .zip) from your ISV, extract this to the location you will install RLM.

Ensure that you have rlm.exe; your license file; and either <isvname>.set or <isvname>.exe in this folder.

Double-clicking the RLM executable will open a command prompt Window, and effectively start your license manager; however, this will need to be started manually every time Windows is restarted, or the command prompt is closed. It is recommended that you install RLM as a service.

Installing the Service

1. Open the start menu, type "cmd," right-click "Command Prompt," and select "Run as administrator."
2. Use the cd command move to the installation directory. For example:

```
cd c:\rlm
```

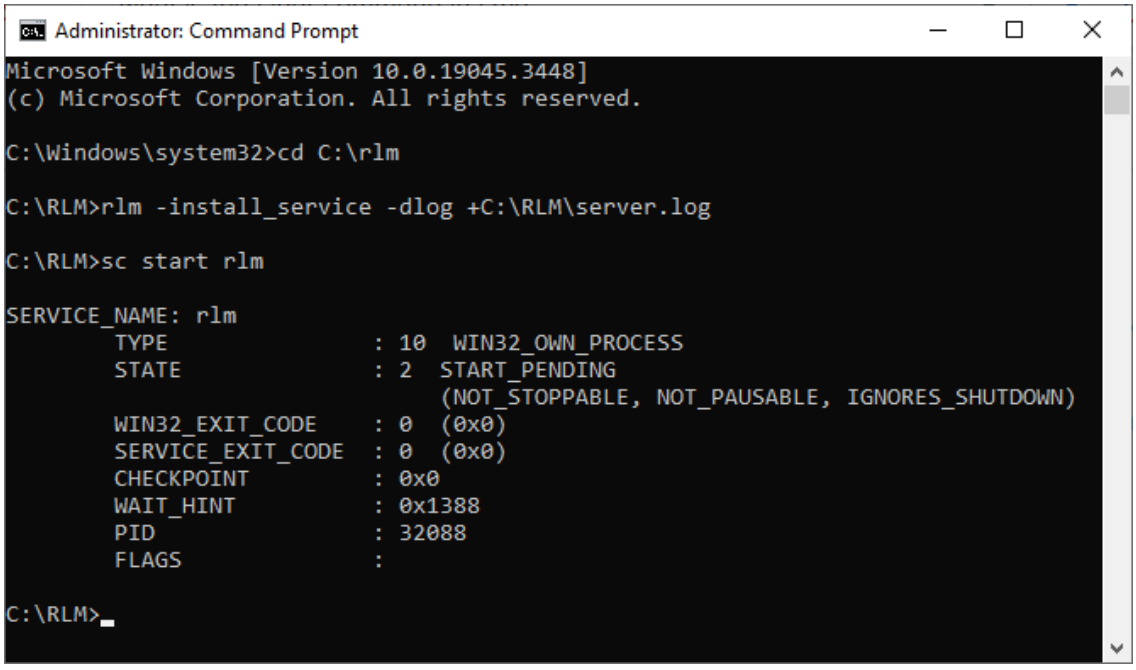
3. Now install the service. Note that the path for the log will need to match your installation directory.

```
rlm -install_service -dlog +C:\RLM\server.log
```

4. If successful you will be taken to a new line with no messages. Now start the service:

```
sc start rlm
```

5. You should now see a message containing various information related to the service, including "Start Pending." You should now see "RLM" in your Windows Services.



For more information and additional installation parameters see Section 1.3.1.

3.7.3 Web Management Interface

The default installation includes a web administration interface that can be accessed through a browser to perform license administration tasks. This can be access using https://<server-hostname>:5054 or https://<server-IP>:5054 (for version 15.2 and below, this may be http only).

The default login and password for the web interface are:

Username:	admin
Password:	admin

The first time you login you will be prompted to change the password.

See Section 1.7.

3.7.4 Important Notes

- RLM uses port 5053, as well as a port for your ISV that is dynamic by default. The ISV port can be set to a static port if you need to create a firewall rule. See: Section 1.4.4.

- The web interface uses port 5054. If the RLM server cannot bind to this port, it will exit. To access the interface outside of the server, clients will need to be able to communicate over this port.
- Ports should be open TCP inbound on the server firewall.