RLM License Administration

RLM v12.0

December, 2015

Contents

RLM License Administration - Copyright (c) 2006-2015, Reprise Software, Inc

RLM - Reprise License Manager - Copyright (c) 2006-2015 Reprise Software, Inc

Section 1 – License Management Introduction

Introduction

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 which uses the Reprise License Manager. This manual explains how to install and configure the licensing system included with your software.

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.

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.

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.

What's New in RLM v12.0

This section lists the new features in RLM v120, along with pointers to the relevant sections in the manual.

- ***The "install Windows Service" menu item in the RLM web interface has been removed.*** To install RLM as a service, use the rlm -install_service command from a command window running as administrator.
- ***RLM v12.0 will sort licenses (within a license file) in the order of the _id keyword.*** Licenses without _id keywords will remain unsorted at the end of all the licenses with _id keywords, which are sorted in increasing numerical order. See License ID on page 26 for more information.

Section 2 – License Administration Basics

This section of the manual contains the information you need to install and manage your purchased applications which use the RLM license manager.

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 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 License Administration Tools on page 38.

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.

Here are the details of how 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. 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.

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.
- NB: 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.

The License Server

The license server consists of at least two processes

- The generic server, called *rlm*
- At least one ISV server, named *isv*

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 The RLM Web Server on page 45.

Note that you should *NEVER* run the rlm servers as a privileged user (root on unix or administrator on Windows).

rlm startup options

The *rlm* command is:

```
% rlm [-c license_file] [-dat] [-dlog [+]logfile]
      [-nows] [-ws port] [-x [rlmdown|rlmremove]]
      [-install_service] [-service_name sname]
      [-user username -password password]
      [-isv_startup_delay seconds] [-v] [-info] [-noudp]
```

The **-c license_file** option specifies which license file to use. This option overrides the setting of the RLM_LICENSE environment variable. Note that the -c option first appeared in RLM v2.0. Beginning in RLM v6.0, 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. This option first appeared in RLM v3.0.

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. This option first appeared in RLM v2.0. (Note: starting in RLM v4.0, 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. This option was introduced in RLM v9.3BL2.

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>", eg ".\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. For details on how to set that property on an account, see this blog: http://blogs.msdn.com/b/ablock/archive/2008/09/18/setting-the-properties-the-log-on-as-a-service-and-allow-log-on-locally.aspx

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 or rlmid2 (hardware keys), 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. This option was introduced in RLM v8.0BL6.

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 it's version and exit. This option was introduced in RLM v9.3BL2.

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.

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 REPORTLOG line in The ISV Options File on page 62 for more information.

Note that if the *rlm* server cannot bind the web server port (5054 by default), it will exit. Note that the web server default port was 9000 prior to RLM v6.0.

Also note that, prior to RLM v3.0, if there was not at least one license file with the current hostname (as returned by gethostname()), or "localhost", the servers would not run. This condition generates a warning in RLM v3.0 and later.

Starting in RLM v10.0, 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.

The Server Debug Log

Both RLM ad 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 The ISV Options File on page 62 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).

RLM debug log is specified.

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.

Example:

```
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 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.
- Beginning in RLM v8.0, when running as service, rlm now changes its working directory to the directory where rlm.exe is installed. This is so that log files will be written there instead of in c:\windows\system32 as in prior versions (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:\windows\system32.
- The web interface service installation is new in RLM v7.0
- Prior to RLM v4.0 when running rlm as a service, it is strongly recommended that you specify a debug log location for each ISV server. This is done in The ISV Options File for each ISV server, using the DEBUGLOG keyword. If no location is specified for the debug log, the ISV server's debug information is lost when running as a service. Starting

in RLM v4.0, the ISV servers place their debug output in the same file as the rlm server's debug log, as specified in the -dlog option, so no DEBUGLOG specification is necessary.

- Starting in v9.4 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.

Starting the *rlm* server at system boot time on Unix systems

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. Note that you must install this startup script as root.

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 -q"
        }

        case "$1" in
          start)
                start
                ;;
          stop)
                stop
                ;;
          restart)
                stop
                sleep 2
                start
                ;;
          *)
         echo $"Usage: $0 {start|stop|restart}"
                exit 1
        esac

        exit 0
```

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, and 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. In rlm v2.0 and later, 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* (by this we mean 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 (in RLM v2.0) licenses that will expire within 14 days. ISV servers do not process single-use (count==*single*) licenses.

Beginning in RLM v5.0, 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.

ISV server open file limits

Beginning in RLM v6.0, ISV servers on Unix platforms will attempt to increase their open file limit. If a server is able to increase it's 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

How Licenses are Pooled by the ISV Server

When the ISV server processes all it's 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 (ie, 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 non-zero id will only pool with the same same ID# (assuming all the other attributes make it eligible to pool).

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 License Administration Tools on page 38. In addition, options can be specified for each ISV server in The ISV Options File. You can restrict access to administration commands via The RLM Options File.

The License File

The license file contains information which configures the license servers and describes all the licenses granted from the ISV to their customer. License Files have 3 types of lines:

1. HOST Lines that specify the license server host
2. ISV Lines which specify the ISV's license server information, and
3. LICENSE Lines which describe license grants from the ISV to the customer.

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

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

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 four characters are illegal in data items in the license (and options) file: "<", ">", "&", and double-quote ("). Note: single quote (') and back-quote (`) were illegal prior to RLM v8.0. ISV license names cannot begin with the characters "rlm_".

Note that 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.

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

- *isv-binary-pathname* on ISV lines [Note: case-sensitive on Unix systems only]
- *options-file-filename* on ISV lines [Note: case-sensitive on Unix systems only]
- short (~62-character) license keys (keys with bits/character of 6)

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.

The 4 types of license lines (HOST, ISV, LICENSE, and UPGRADE) are described below.

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, so you can change it at any time without affecting your licenses.

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 is not an input to the license key signature algorithm, so it can be changed to any available port.

For a description of the various hostids that RLM supports, see RLM hostids on page 104.

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" – they are 100% equivalent.

ISV line

Format (pre-RLM v9.0):

ISV *isvname* [*isv-binary-pathname* [*options-file-filename* [*port-number*]]]

Format (RLM v9.0+):

ISV *isvname* [*isv-binary-pathname* [*options-file-filename* [*port-number*]]] [binary=*isv-binary-pathname*] [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*.

The *isvname* is the name assigned by Reprise Software to the ISV and does not change.

Beginning in RLM v6.0, the ISV server can be delivered as either an executable (as in all older versions of RLM) or as a small, platform-independent ISV server **settings** file (named *isvname*.set by default).

The *isv-binary-pathname* is the filesystem location of the ISVs license server binary. This can be any accessible location. The *isv-binary-pathname* is not an input to the license key signature algorithm, so you can change it to relocate the ISV server at any time. Starting in RLM v3.0, the ISV pathname can be omitted if the isv server is located in the same directory as the rlm binary. If omitted, RLM will first attempt to open an ISV server settings file (*isvname*.set), and if that fails, will attempt to open a license server binary (*isvname*.exe on windows, or *isvname* on unix).

The third (optional) parameter specifies whether an options file is to be used for this license server. If you would like to specify options (see The ISV Options File on page 62), either specify the location of the file containing these options here, or name the ISV options file *isvname*.opt and place it in the directory which contains the license file which the server reads.

The fourth (optional) parameter specifies the port # which the ISV server will use. This should normally be omitted, but can be used if you need to access the ISV server across a firewall and the firewall needs to be configured to allow access to the port. Note that you must specify an options file if you want to specify an ISV server port number.

The fifth (optional) parameter (new in RLM v9.2) specifies a license password to be applied to all LICENSE or FEATURE lines which **follow** the ISV line in the license file. If an individual LICENSE line has a password, the password from the LICENSE line is used.

In the old format, the parameters are strictly positional, and, for example, to specify a port #, the ISV server binary and options file must both be specified. However, in the new format, 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 that, in the new format, 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:

ISV reprise /home/reprise/reprise /home/reprise/reprise.opt (old format)

ISV reprise options=/home/reprise/reprise.opt binary=/home/reprise/reprise (new format)

ISV reprise /home/reprise/reprise port=8765 (new format)

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

In these examples, the license server for ISV *reprise* is located at /home/reprise/reprise and (in the first 2 examples) an options file is located at /home/reprise/reprise.opt. In the 3rd example an ISV server port # is specified. In the fourth example, the ISV server binary name /a/b/reprise will be used instead of /home/reprise/reprise.

Note: The keyword "VENDOR" can be used instead of "ISV" – it is 100% equivalent.

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, ie, less than 62-character, license keys), and none may be modified by the License Administrator with the exception of optional parameters whose keywords begin with the underscore ('_') character.

Types of Licenses

While there is a single format for the LICENSE line, the licenses you receive from your software suppliers can have many different meanings. The following licensing attributes are present in all licenses, and define the major meaning of the license itself:

- **Locking: Node-locked** (counted or uncounted), **Username-locked** (counted or uncounted), or **Floating licenses**.

  RLM can lock a license in a variety of ways:

  1. A license can be *node-locked*. 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 hostids on page 104.
     - A node-locked license can be either *counted*, *uncounted*, or *single*. If it is *uncounted* or *single*, then the software only need 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 that *single* licenses are checked locally to ensure that only one instance is running.)
     - To create a node-locked license, add the keyword **hostid=..** at the end of the license line. See the description of the LICENSE line for more information.
  2. 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 that any white space in a username is converted to the underscore ('_') character.
  3. 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.

- **Expiration: Expiring or non-expiring licenses**

  All licenses have a *expiration date*. If the license uses the special expiration date of ***permanent***, they never expire (any date with a year of 0 is also non-expiring, e.g. 1-jan-0).

- **Version: By version number or by product release date.**

  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:

  * Your ISV could make all your software request version 1.0 with all your licenses issued for version 1.0, and the version would never be an issue, unless and until they wanted to obsolete all the old licenses on a new release.

  * Or, your ISV could put the product's version number in the rlm_checkout() call, then licenses for an older version of the product will not work with a newer version of the product.

  * Or, your ISV can use a date-based version. To do this, they might put the year of release into the *rlm_checkout()* call, then when you issue licenses, issue them either for this year, or some year in the future. This allows you, the customer, to use products released on or before the year in the license. Alternately, the ISV could make the version be "year.month".

- **License Count**: Each license has a count of available licenses. If this count is "0" or "uncounted", the license count is not enforced. Otherwise, only the specified number of license checkouts are allowed.

Fixed (positional) parameters

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

isv

isv is the name of the ISV granting the rights.

product

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

version

version is the highest-numbered product version supported by this license, in the form "N.M". For example, 1.0, 2.37, or 2006.12

exp-date

exp-date is the date the license expires, in the form dd-mmm-yyyy, for example, 1-jul-2007. Any license with either a year of "0" (ie, "1-jan-0"), or the word "permanent" does not expire.

count

count is the number of licenses granted. **0** or **uncounted** means an uncounted, node-locked license. **uncounted** and **0** are 100% equivalent.

single means a node-locked, single-use license. **single** 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, nodelocked 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 for your Software Provider to issue you a single-use license without you, the License Administrator, having to configure a license server.

token, **token_locked**, and **token_unlocked** are used to specify a Token-Based License; this license must also have the **token=...** optional parameter (see Token-Based Licenses on page 75). 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-key

license-key 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 it's *license-key*. The *license-key* will have "sig=" prepended after the license has been signed by the *rlmsign* utility.

Note that if the *license-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 per license. Note that parameter names which begin with the underscore ('_') character can be either added or modified by the License Administrator without invalidating the license signature.

akey=activation-key

The RLM Activation Pro license generator can include the akey= keyword with the activation key used to fulfill the license, if specified by your Software Publisher. This parameter is unused by RLM. akey= first appeared in RLM v11.0.

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 (note: the list must be in quotes if more than one item is specified):

- **TerminalServer** - disable use of Windows Terminal Server or Remote Desktop.
- **VM** - Disable use on Virtual Machines.

Example:
disable=TerminalServer
disable=vm
disable="vm TerminalServer"

disable= first appeared in RLM v4.0, with the single environment *TerminalServer*. *VM* was added in RLM v5.0.

_failover_host=hostname

This field is used only on an "rlm_failover" or "rlm_failover_server" license, and is set by the License Administrator to the failover server's hostname. See Failover License Servers on page 81 for more information. This field has no meaning on any license other than an "rlm_failover" or "rlm_failover_server" license.

hold=n

hold specifies that a license is to be "held" checked-out by the license server after the application performs a checkin call or exits. The license will remain checked-out for the number of seconds *n* specified after application checkin or exit. *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.

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 after the application checks it in, 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. See also min_checkout on page 27.

host_based[=n]

host_based indicates that the specified number of licenses (or all licenses) must be reserved to hosts 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 refuse to serve the license. Also note that licenses reserved to a HOST_GROUP are not counted. Thus, all reservations must be to individual hosts.

hostid=hostid-string

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 hostids on page 104.

Starting in RLM v5.0, 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 hosts:

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

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 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, 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.

issued=issue-date

The optional *issued=issue-date* attribute is used in conjunction with the *replace* keyword. See the description of *replace* for a description of how the issue-date affects license replacement.

max_roam=days

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.

max_roam is used to specify the maximum number of *days* which a license can be held by the server and used on a disconnected system.

If your Software Provider does not specify *max_roam* in an individual license, RLM limits the maximum number of days that a license can roam to 30 days. If *max_roam* is set to -1, roaming is disabled on that particular license.

Also note that if your ISV specifies *max_roam* on the *rlm_roam* license itself, this *max_roam* specification will apply to all of your products which do not have *max_roam* specifications.

max_roam_count=count

max_roam_count specifies the maximum number of licenses which can roam. If unspecified, all licenses are allowed to roam. If set to 0, no licenses are allowed to roam. If 2 licenses are pooled, their max_roam_count values are added. Finally, you can lower this value by using the

ROAM_MAX_COUNT option, however ROAM_MAX_COUNT will never raise this value. max_roam_count is new in RLM v8.0.

min_checkout=n

min_checkout specifies that a license is to be "held" checked-out by the license server after the application performs a checkin call or exits if the license did not remain checked out for the minimum interval specified by *n*. 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 after the application checks it in, 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. See also hold on page 25.

Example 1:

hold=30
application checks license out for 5 minutes
application checks license in
at this point, the server keeps the license checked out for an additional 30 seconds.

Example 2:

min_checkout=30
application checks license out for 5 minutes
application checks license in
at this point, the license will be checked in by the server with no extra hold time.

Example 3:

min_checkout=400
application checks license out for 5 minutes
application checks license in
at this point, the server keeps the license checked out for an additional 100 seconds.

NOTE: The license server scans all held licenses once per minute, so the exact time of checkin can be rounded up to the next 60 seconds. So, in example 3 above, the checkin could happen any time between 100 seconds and 120 seconds after the rlm_checkin() call.

NOTE: The minimum checkout time specification will be ignored for any license which is roaming.

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 is disabled on this license.

min_timeout=n

You 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 checkin of the license and informs the application (if it is still running).

min_timeout specifies the lowest value, in seconds, you can set the *TIMEOUT or TIMEOUTALL* value for a license. If not specified in the license, the RLM default of 3600 seconds (1 hour) is used. Note that licenses NEVER time out on their own, you must set a TIMEOUT or TIMEOUTALL option in the options file to have them time out.

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

_password=password-text

The *_password* specification limits who can use this license to users who know the password.

You can assign a password to a license in order to restrict the ability to check out, or even see the license.

To do this, specify:

    _password = password-string

in the license.

If specified, a license password restricts access to this license to requests which have specified the same password-string. The password-string is specified with the RLM_LICENSE_PASSWORD environment variable, or in the RLM web interface for viewing licenses.

Note that 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.

Also note that if you do assign a password to a license, and the application fails to supply the correct password, the server will return a "License Server does not support this product" error (status -18) when a checkout is attempted.

Beginning in RLM v9.2, license passwords can be specified on the ISV line, with the new "password=password-text" option. When specified this way, the password on the ISV line applies to all LICENSE or FEATURE lines for the ISV which **follow** the ISV line in the license file. If an individual LICENSE line specifies a password, the password from the LICENSE line is used in place of a password on the ISV line.

platforms=platform-list

The *platforms* specification limits the types of computers which can use this license.

RLM allows your ISV 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.

The *platform-list* consists of a list of RLM-defined platform names, which consist of a machine architecture and an operating system version/revision, as in the following table:

Platform	RLM Platform name	string in *platforms=*
HP-UX on PA-Risc	hp_h1	hp_h
HP-UX 64-bit on PA-Risc	hp64_h1	hp64_h
IBM AIX 32-bit	ibm_a1	ibm_a
IBM AIX 64-bit	ibm64_a1	ibm64_a
Linux on Intel X86	x86_l1, x86_l2	x86_l
Linux 64-bit on Intel	x64_l1	x64_l
MAC on Intel X86	x86_m1	x86_m
MAC on PPC	ppc_m1	ppc_m
Solaris 64-bit on Intel	x64_s1	x64_s
Solaris on Sparc	sun_s1	sun_s
Solaris (64-bit) on Sparc	sun64_s1	sun64_s
Windows on Intel X86	x86_w1	x86_w
Windows 64-bit on Intel X86	x64_w1	x64_w

replace=[product-list]

The *replace* specification causes this license to replace other license(s).

In order to render ineffective one or more licenses which have already issued to you, the ISV uses the *replace[=product-list]* option in the new license. *replace=* causes RLM to ignore the "replaced" license(s).

Note: If your ISV uses *replace*, they must also have specified either *start=* or *issued=*.

***replace* operates as follows:**

- licenses from the *product-list* will be replaced. 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.

share=UHI[:nnn]

share specifies how a license is shared between separate clients (processes).

Licenses can be shared between separate running processes. To do so, your ISV will have put 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. Share is specified 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.

For example, if share is specified as *share=UH*, then both the username and the hostname of a request must match an existing checked-out license in order to share that existing checked-out license. If share is specified as *share=u*, then only the usernames must match on two processes in order for them to share the license.

The share= keyword accepts an optional maximum process count which can share the license:

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 that 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 it's 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 checkin time.

soft_lmit=n

A license can have a *soft_limit* that is lower than it's count of available licenses. Once usage exceeds the *soft_limit*, checkout requests will return the RLM_EL_OVERSOFT status instead of a 0 status. The application's behavior in this case is entirely up to your ISV.

Note that 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*. Also note that licenses which do not specify a *soft_limit* use the license *count* as their *soft_limit*.

start=start-date

start= specifies a *start-date.* This license cannot be used before the specified date. The format is the same as the expiration date.

timezone=timezone-spec

timezone= allows your ISV to specify a set of valid timezones for the client machine that performs the license checkout. If timezone= is present in the license, there is a timezone restriction. The

timezone-spec is a 24-bit HEX number, with one bit set for each timezone your ISV wishes 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 allow the license to be used in PST only, the timezone spec would be as follows, specifying timezones 7 and 8 west of GMT:

timezone=180

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

token= specifies the list of products which are checked out when a *token-based* license is requested. See Token-Based Licenses on page 75 for a complete description.

user_based[=n]

user_based indicates that the specified number of licenses (or all licenses) must be reserved to users 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. Thus, all reservations must be to individual users.

The following fields are not used by RLM, but are present to identify licenses:

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.

customer=who

customer= is to identify the customer of the software. *customer* is not used by RLM.

issuer=who

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

_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. _line_item first appeared in RLM v3.0.

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

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, in the form dd-mmm-yyyy	11 characters
count	positive integer	2**31 - 1
hold	positive integer - seconds	2*31 - 1
host_based	int	count of licenses which must be reserved
hostid (single)	string	72 characters
hostid (list)	space-separated, quoted string	200 characters, max of 25 hostids
issued	string, in the form dd-mmm-yyyy	11 characters
_line_item	string, License Administrator defined	64 characters
max_roam	positive integer - days	2**31 - 1
max_roam_count	positive 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
password	string	32 characters
platforms	string	80 characters
share	enumerated	3 ("uhi") + :integer
soft_limit	integer	2**31 - 2
start	string of the form dd-mmm-yyyy	11 characters
timezone	int	bitmap with bits 0-23 set
user_based	int	count of licenses which must be reserved
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" and/or "beta"	14 characters

Examples:

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

LICENSE reprise calc 1.0 1-aug-2008 5 9874359783459734573498753987495873459873 45

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, subject to the rules for combining licenses listed in How Licenses are Pooled by the ISV Server on page 17.

UPGRADE line

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, ie, 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 that 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 **upgrade**d.

Certain licenses can never be upgraded. In particular, token-based licenses (the token definitions

themselves) will never be upgraded. Also, named-user licenses are not eligible for upgrades.

In order for a license to be upgraded, the *base license* and the *upgrade license* must be eligible to be pooled by the license server as described in How Licenses are Pooled by the ISV Server on page 17.

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 that 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.

Note that the 3rd case above is the most useful - if your ISV wanted to upgrade 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, ie, the *upgrade license*, by itself, grants no rights.

When a license is **upgrade**d by the license server, the new licenses will have their parameters modified as follows:

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 -
max_roam_count	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	If licenses have a specification, upgrade spec is preserved for the new version; base license spec	- undefined -

(partial upgrade)	*delta* (count-user_based) is preserved (minimum value 1) for the old version	
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. Note that these are the same as the 6 fixed parameters of the LICENSE line, with the version replaced by a pair of versions:

isv

isv is the name of the ISV granting the rights.

product

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

from-version

from-version is the lowest-numbered product version which is eligible for an **upgrade**, in the form "vN.M". For example, 1.0, 2.37, or 2006.12

to-version

to-version is the highest-numbered product version supported by the license once it is **upgrade**d, in the form "vN.M". For example, 1.0, 2.37, or 2006.12

exp-date

exp-date is 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" (ie, "1-jan-0"), or simply the word "permanent".

count

count is 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 the LICENSE line on page 22 for more information.

token, **token_locked**, and **token_unlocked** are not allowed in an UPGRADE license. All other optional parameters are ignored.

upgrade-key

upgrade-key is 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 that 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 the License Line for information on the optional parameters.

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 (eg, 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

license_spec1:license_spec2:license_spec3: .... :license_specN (UNIX)

license_spec1;license_spec2;license_spec3; .... ;license_specN (Windows)

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 that the separator character is ':' for Unix systems and ';' for Windows systems.

Also note that the license file cannot be placed in a path where any component of the pathname contains the '@' character.

Example

% setenv 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.

% 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".

Note: The ability to specify the license directly was added in RLM v3.0.
Note: The ability to add a directory to the RLM_LICENSE environment variable was added in RLM v6.0.

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 The RLM Web Server on page 45.

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: *rlmdebug*, *rlmdown*, *rlmhostid*, *rlmnewlog*, *rlmremove*, *rlmreread*, *rlmstat*, *rlmswitch*, and *rlmswitchr*. On UNIX platforms, *rlmutil* is an all-in-one utility which is installed with hard links to all the utility program names. On windows, separate .exe files are provided for each utility.

RLM has the ability to restrict usage of the **remove**, **reread**, **shutdown**, **status** and **option editing** requests. All restrictions are done via The RLM Options File. For a description of how to restrict command usage, see The RLM Options File on page 57.

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. Note that prior to RLM v4.0 the RLM_LICENSE environment variable takes precedence over a *-c license_spec* specification. Starting in RLM v4.0, the -c option overrides RLM_LICENSE.
-dat	use *.dat as the license file instead of *.lic (new in rlm v3.0)
-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 (new in RLM v9.4)

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 file *logfile.anon*.

There is a one-to-one mapping from a particular user or host name to it's 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.

rlmanon first appeared in RLM v4.0.

rlmdebug - display debugging information about products

Usage: **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
```

rlmdown - shuts down the license server

Usage: **rlmdown** [-q] [isv]

rlmdown shuts down the first license server in the license path RLM_LICENSE. If the *-q* option is specified, the shutdown happens without a confirming 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 that (on Unix systems only) the servers can also be shut down by sending a SIGTERM signal to the *rlm* process. SIGTERM shuts down all the servers, including rlm.

rlmhostid - print the hostid of this machine

Usage: **rlmhostid** [[-]32|ether|ip|internet|host|user] [-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 on page 104.

Note: Beginning in RLM v4.0, if the rlmhostid command prints "(Virtual: some text)" after the hostid, this means that the command is run on a virtual system which is not supported for running an RLM license server. In the case of Solaris systems, license servers can only be run on zone 0 (the "Global" zone).

rlmnewlog - moves the old report log info to a new file

Usage: **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 that 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.

rlmremove - remove a checked-out license from a user

Usage: **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
```

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

Usage: **rlmreread** [*isv*]

rlmreread causes the specified ISV server *isv* to reread it's 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 it's license file.

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

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

Note also that (on Unix systems only) the servers will do a reread if a SIGHUP signal is sent to the *rlm* process.

rlmstat - obtains status from the license servers

Usage: **rlmstat** [-a] [-i [isv]] [-l [isv]] [-n [host]] [-p [product]] [-u [user]]

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)
-i	display this *isv* only	Display license checkout info from ISVs
- 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

Example rlmstat output

```
% 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.

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

rlmstat -avail command

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

Usage:

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.

Note that 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.

Also note that 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

```
% rlmstat -avail -i reprise
rlmstat v9.1
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)

```
% rlmstat -avail -i reprise -b
rlmstat v9.1
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
```

rlmswitch - switches the debug log info to a new file

Usage: **rlmswitch** [*isv*] *new-log-file-name*

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

rlmswitchr - switches the report log info to a new file

Usage: **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*.
Beginning in RLM v9.1, *rlmswitchr* will fail if the server is not currently writing a report log.

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.

In addition, the web server allows you to edit server option files (if you have access to the **edit_options** capability - for ISV servers, or the **edit_rlm_options** capability - for rlm itself.) Also, the web interface allows you to view the recent debug log information from any of the servers if you have access to the **status** capability. Finally, access to the **status**, **reread**, and **shutdown** commands is controlled by the appropriate capability as specified in The RLM Options File on page 57 (or by the login credentials for RLM v10.0 or newer servers).

The web server is started automatically on port 5054 when rlm is started. To use the web server, simply point your browser to: *http://ServerHostName:5054* and select the operation you would like to perform. You will be prompted for any required information. (Note: RLM ran the web server on port 9000 prior to v6.0).

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.

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

Access Control to the RLM Web Interface

Beginning in RLM v10.0, it is possible to require users to log in to the RLM Web Interface.

The login capability is provided via the *rlm password file*, named *rlm.pw*. If this file exists in the directory with the rlm binary, then the RLM Web Interface will require users to log in before they can perform any actions. Reprise Software recommends that you protect access to this file so that ordinary users can't write it. The *RLM password file*, as well as the directory which contains it, must be read-write to the rlm process.

The *RLM password file* has one line for each user, formatted as follows:

username:password:list-of-permissions

The *username* must not contain a ':' character.

If the *password* field is blank, then the user can log in without supplying a password. To change their password, select the "Change Password" menu item once logged in as that user. The

password field is an encrypted hash of the actual password (similar to the Unix password file).

The *list-of-permissions* field is a comma-separated list of the various privileges which you can assign to this user. These names are the same names you would use in the RLM options file if you were controlling access without logins enabled, with the addition of the special "all" permission, which enables all operations. If you use the *RLM password file* to control access, you should not use the RLM options file to control access.

Beginning in rlm v10.0, you can control the appearance and defaults of the “Activate License” command with the rlm “-activate” options in the rlm options file.

RLM privileges assignable in the RLM password file

Privilege	Meaning	Notes
all	Special privilege name, enables all privileges	
edit_meter	Allows modifying count for meter counters	Enables "status" privilege even if not present
edit_options	Allows editing options files for ISV servers	Enables "status" privilege even if not present
edit_rlm_options	Allows editing license files and options files for the rlm server	Enables "status" privilege even if not present
edit_xfer	Allows editing server-server license transfer settings for ISV servers	Enables "status" privilege even if not present
logfiles	Enables the functions which change log files - switch, switchr, newlog	
remove	Allows the user to remove a license from a running process	Enables "status" privilege even if not present
reread	Allows access to the functions which do reread commands on license servers	
shutdown	Allows access to the functions which shut down license servers	Enables "status" privilege even if not present
status	Allows display of status and debug log information from the license servers	

A user with no privileges assigned will have access to the "Activate License", "Diagnostics", "RLM Manual...", "System Info", "About", "Change Password", and "Logout" commands.

A couple of example password line entries shown here:

```
tom:$5ukMApW1jixwcrGqRALO91:all
harry::edit_options,edit_rlm_options,reread
```

User "tom" has a password assigned, and can perform all actions with the web interface.
User "harry" has no password (therefore no password is required to log in), and has the edit options, edit rlm options, and reread privileges assigned. He will also be able to view status.

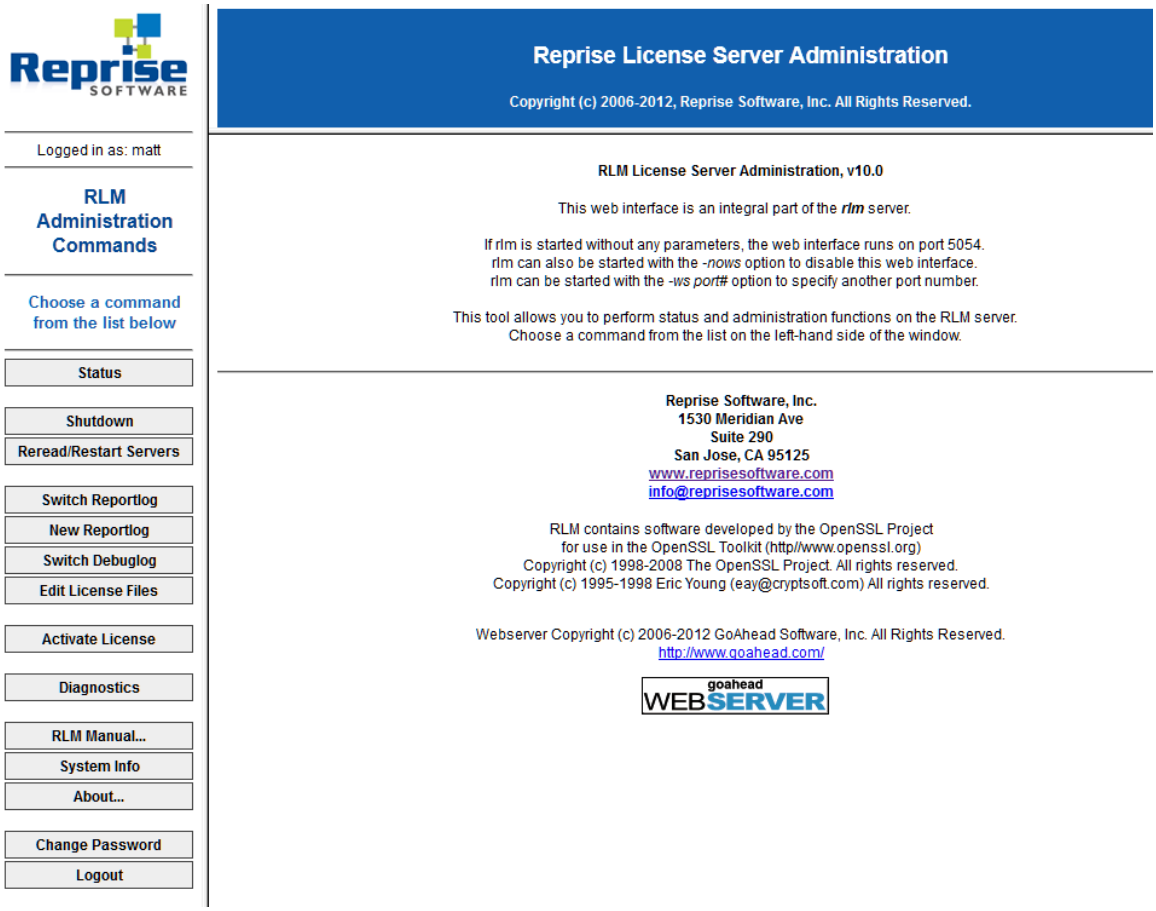
Intro Screen

The intro screen of the RLM web server is shown below. There are 3 sections to the rlm web interface:

- a top banner with the Reprise logo and title
- a command area on the lower left, and
- a general view area in the main lower-right hand side of the screen.

The top section of the view area displays some general information about rlm command options to run the web server. On the left-hand side is a list of administration commands which are discussed later.

Note that, beginning in RLM v10.0, every user will not see all the commands on the left-hand side of the menu, depending on the privileges assigned to that user in the password file or in the RLM options file.



Main Status screen

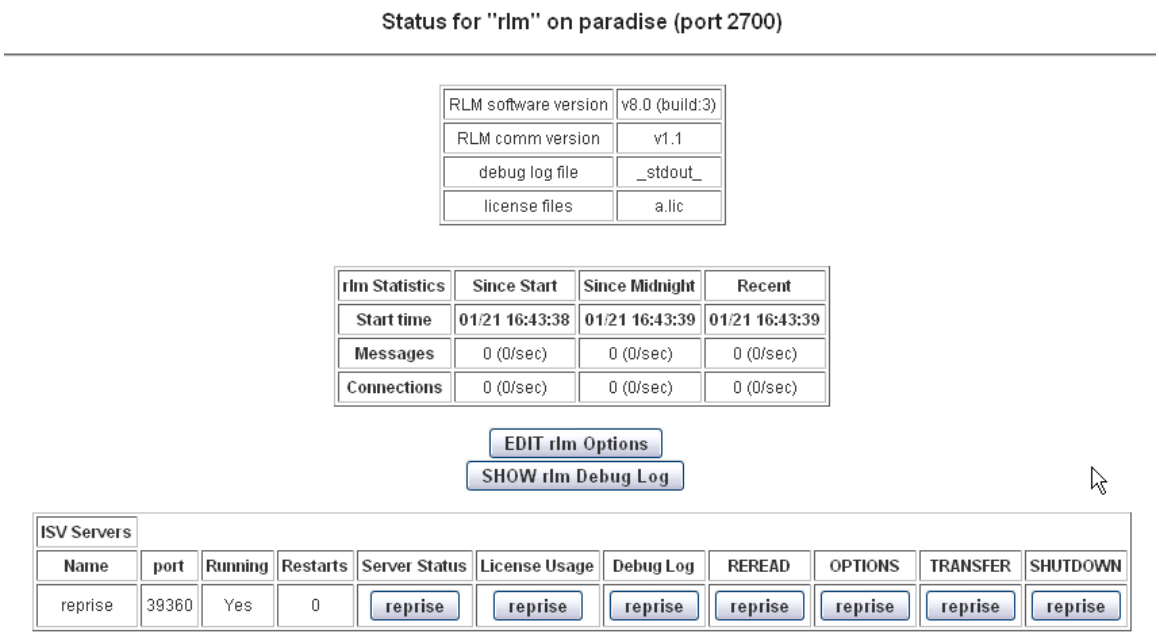
If you select *Status* from the menu on the left, the main status screen is displayed in the view area as shown below.

The top section displays the host information where the rlm server is running - host name and port #.

Below this is status of the RLM server itself, followed by buttons to edit rlm options and display the last few lines of the rlm debug log.

Next is a table of ISV servers, one per line, with a number of buttons on the right-hand side of each line to retrieve ISV server status, license status, display the last few lines of the debug log, reread or restart the server, edit server options, or shut down the ISV server. Note that these buttons (and the corresponding columns) will only appear if the user running the web server has access to these functions, as specified in the rlm options file.

The status screen provides access to the shutdown and reread/restart commands for all the ISV servers, as well as option file editing and debug log viewing for both rlm and the ISV servers.



Server Status

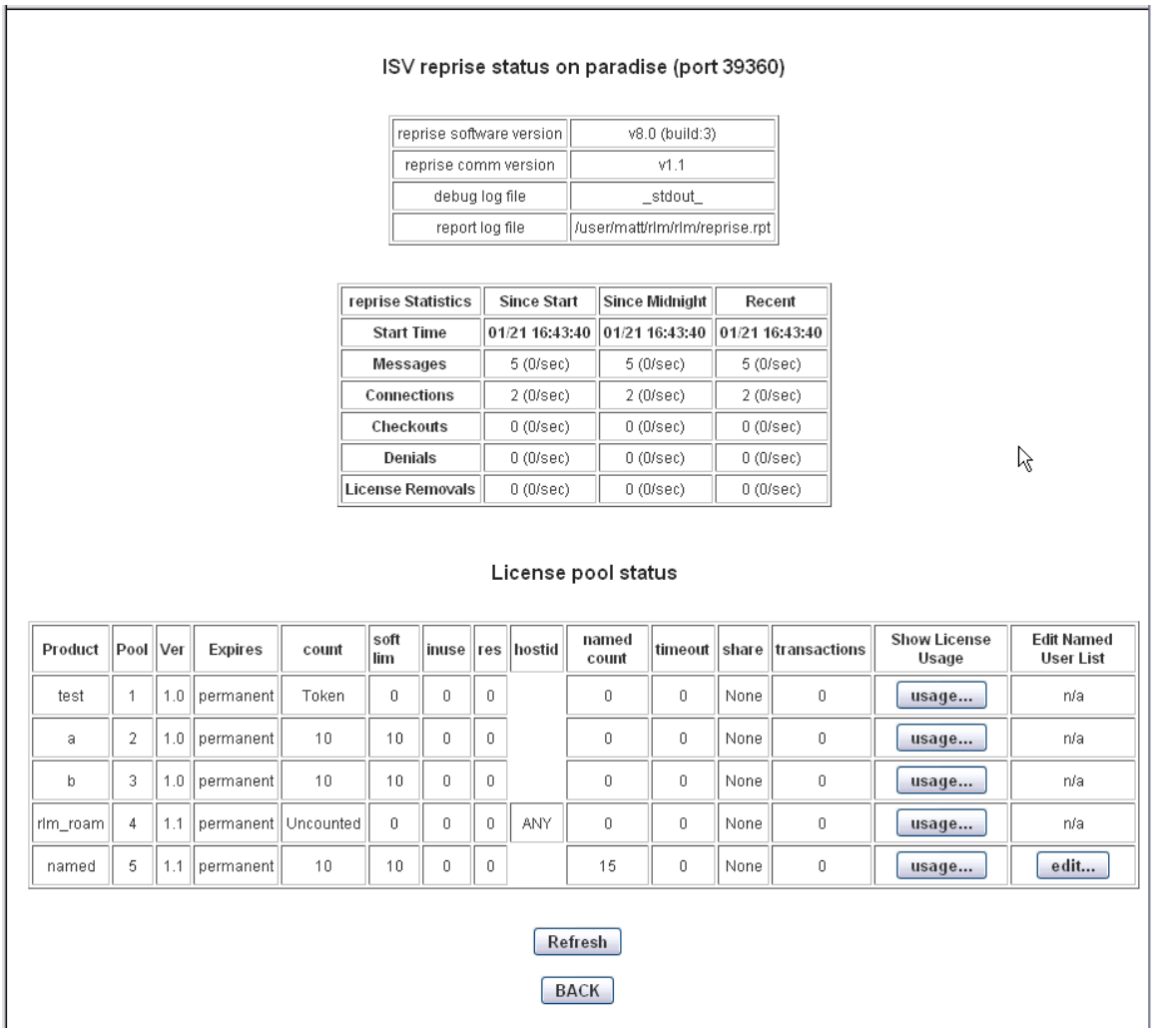
If you click on an ISV button in the *Server Status* column in the ISV server status display, you will see the detailed status display for this ISV server (shown below) in the view area. This display shows some server statistics in a table at the top, followed by a table of all the licenses which this ISV server is serving.

There are several columns in this table which will appear or not, depending on the particulars of the licenses which this server is serving. For example, there are columns for *hostid* (in the case of

node-locked licenses), *roaming* (in the case where some licenses are roamed out to disconnected systems), and *named count* (named user count - in the case of named user licenses). In the example shown here, there are no node-locked licenses, and no licenses are roaming, so these 2 columns do not appear.

Also, please note that 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.

At the far right-hand side of each license line, there are 2 columns. The first column has buttons which, if pressed, will generate a list of users of that product. The second column has buttons which are used to maintain the named user list for named user licenses. Note that if this server is not serving any named user licenses, the 2nd column will not appear. Also, only named user licenses will have edit buttons in this column. In the example below, only the first license is a named user license.



ISV reprise status on paradise (port 39360)

reprise software version	v8.0 (build:3)
reprise comm version	v1.1
debug log file	_stdout_
report log file	/user/matt/rlm/rlm/reprise.rpt

reprise Statistics	Since Start	Since Midnight	Recent
Start Time	01/21 16:43:40	01/21 16:43:40	01/21 16:43:40
Messages	5 (0/sec)	5 (0/sec)	5 (0/sec)
Connections	2 (0/sec)	2 (0/sec)	2 (0/sec)
Checkouts	0 (0/sec)	0 (0/sec)	0 (0/sec)
Denials	0 (0/sec)	0 (0/sec)	0 (0/sec)
License Removals	0 (0/sec)	0 (0/sec)	0 (0/sec)

License pool status

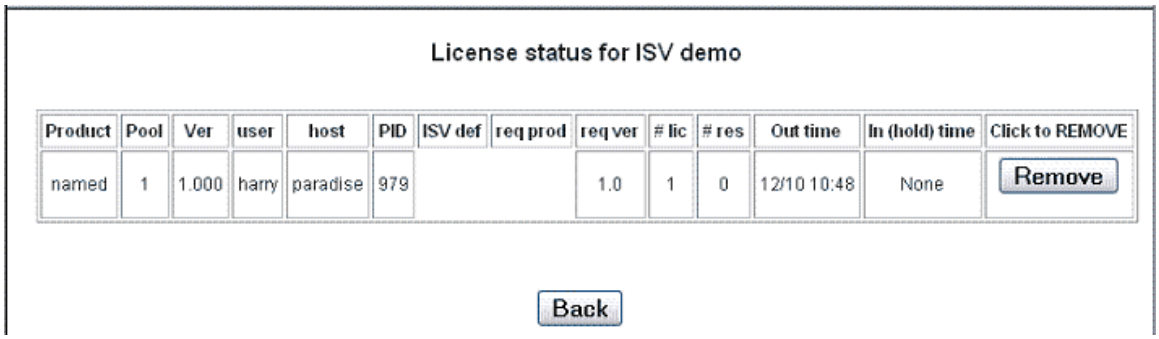
Product	Pool	Ver	Expires	count	soft lim	inuse	res	hostid	named count	timeout	share	transactions	Show License Usage	Edit Named User List
test	1	1.0	permanent	Token	0	0	0		0	0	None	0	usage...	n/a
a	2	1.0	permanent	10	10	0	0		0	0	None	0	usage...	n/a
b	3	1.0	permanent	10	10	0	0		0	0	None	0	usage...	n/a
rlm_roam	4	1.1	permanent	Uncounted	0	0	0	ANY	0	0	None	0	usage...	n/a
named	5	1.1	permanent	10	10	0	0		15	0	None	0	usage...	edit...

Refresh

BACK

License Status

If you click on the **usage...** button in the "Show License Usage" column above, you will see the license status screen, as shown below.



Maintaining Named User Licenses

If you click on the **edit...** button in the "Edit Named User List" column above, you will see the "Edit Named User Definitions" screen, as shown below. This form contains a table of all the named users for this license, as well as a list of recently deleted named users. You can delete any named user from the list by pressing the **Delete** button to the right-hand side of their name. Pressing this button will present a confirmation screen, which then allows you to remove that user from the list. Note that the user cannot be removed from the list if he/she currently has any licenses checked out at the server (including roaming licenses).

Once deleted, a user must remain off the list for a minimum amount of time as specified in the license.

At the bottom of this screen are 2 buttons for adding named users to the list. The first button **Add Group...** brings up a form which has a choicelist of all GROUP definitions from this ISV server's options file. If you select a group to add, group members will be added to the named user list until the list is full, or the group is exhausted.

Below the **Add Group...** button is an **Add New User...** button, which is used to add an individual user to the named user list.

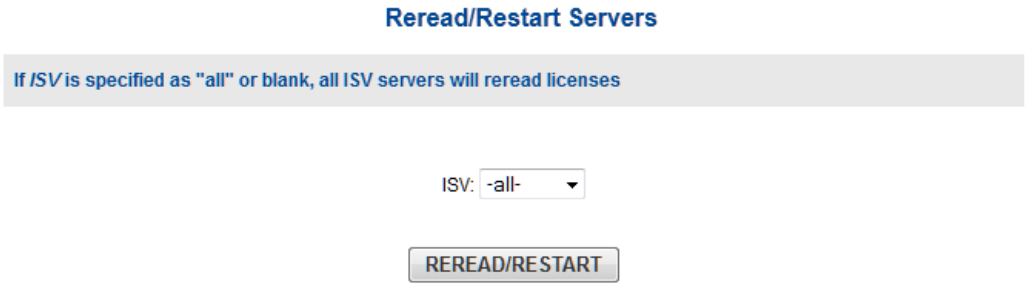
Press the **Back** button if you do not wish to make any changes to the named user list.

Server Shutdown

If you select *Shutdown* from the menu on the left (or from the Shutdown column in the ISV server status display), you will see the **Shutdown License Server** screen below in the view area. If you enter an ISV name that particular ISV server will be shut down. If you leave the ISV name blank or enter "all", all ISV servers will be shut down. Note that you cannot shut down rlm from the shutdown screen. The shutdown will happen when you click the **SHUT DOWN SERVER** button. If you do not wish to shut down any servers, use the browser **back** button, or select a different command from the list on the left.

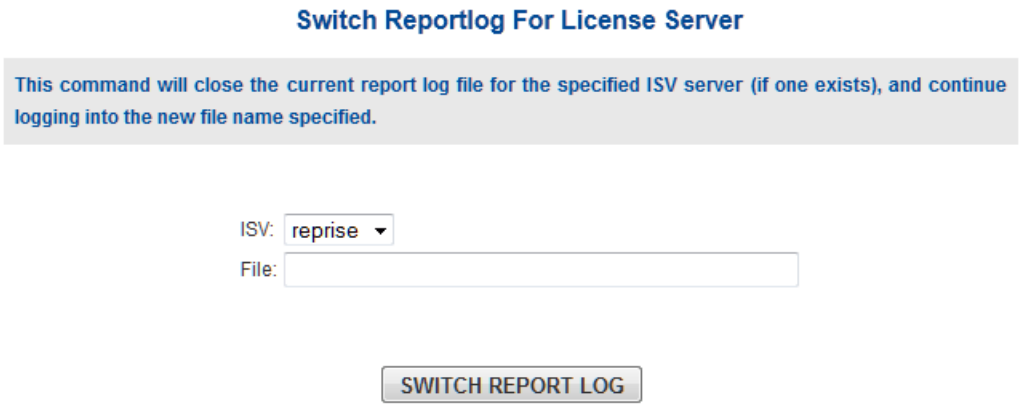
Server Reread/Restart

If you select //Reread/Restart *from the menu on the left (or from the* REREAD/RESTART// column in the ISV server status display), you will see the **Reread/Restart Servers** screen below in the view area. If you enter an ISV name that particular ISV server will be restarted if it is not running, or it will be sent a reread command if it is running. If you leave the ISV name blank or enter "all", all ISV servers will be restarted or reread their license files, as appropriate. If you select *rlm*, the rlm server itself will reread it's license and option files. The reread/restart will happen when you click the **REREAD LICENSES** button. If you do not wish to send the reread command to any servers, use the browser **back** button, or select a different command from the list on the left.



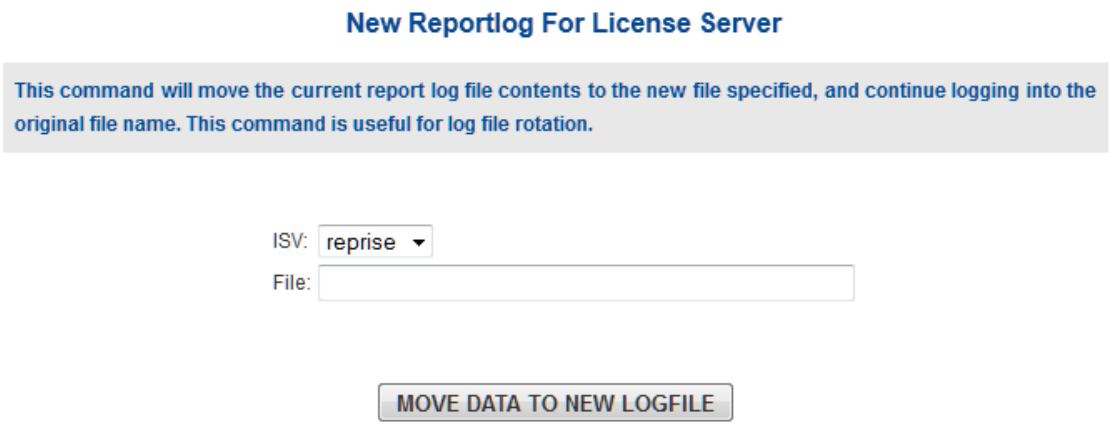
Switch ISV server Reportlog

If you select *Switch Reportlog* from the menu on the left, you will see the **Switch Reportlog For License Server** screen below in the view area. Enter an ISV name and a new filename for the reportlog, then that particular ISV server will begin writing it's reportlog to the filename specified. The switch command will be sent when you click the **SWITCH REPORT LOG** button. If you do not wish to switch the report log, use the browser **back** button, or select a different command from the list on the left.



New ISV server Reportlog

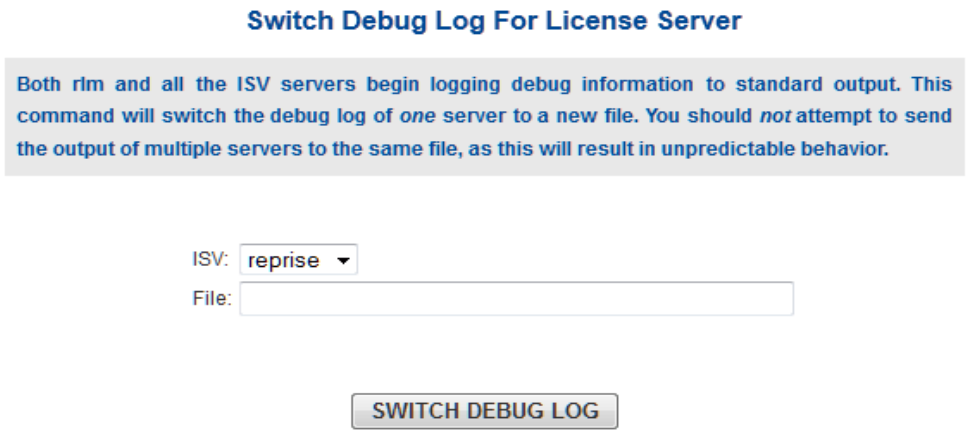
If you select *New Reportlog* from the menu on the left, you will see the **New Reportlog For License Server** screen below in the view area. Enter an ISV name and a new filename for the reportlog, then that particular ISV server will rename the current reportlog to the filename specified, and continue logging to the original reportlog filename. The command will be sent when you click the **MOVE DATA TO NEW LOGFILE** button. If you do not wish to rename the report log, use the browser **back** button, or select a different command from the list on the left.



Switch Debug Log for ISV Server or rlm

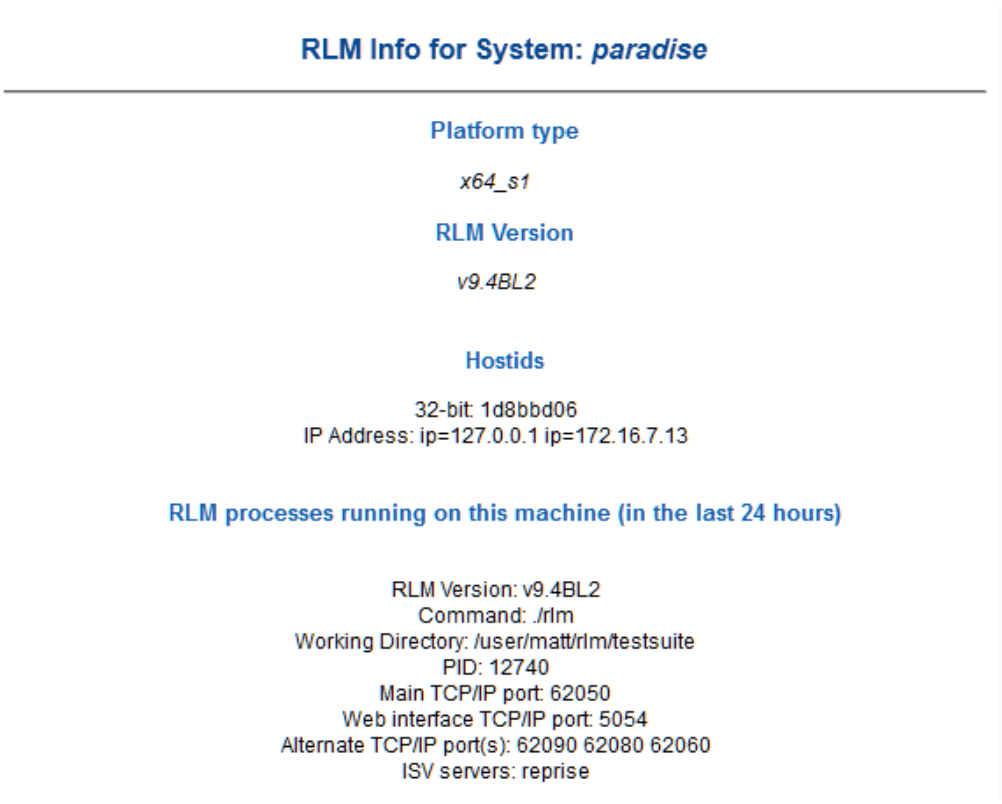
If you select *Switch Debuglog* from the menu on the left, you will see the **Switch Debug Log For License Server** screen below in the view area. Enter an ISV name (or *rlm*) and a new filename for the debug log, then that particular ISV server (or rlm) will begin writing it's debug log to the filename specified. The switch command will be sent when you click the **SWITCH DEBUG LOG** button. If you do not wish to switch the debug log, use the browser **back** button, or select a different command from the list on the left.

Note that 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.



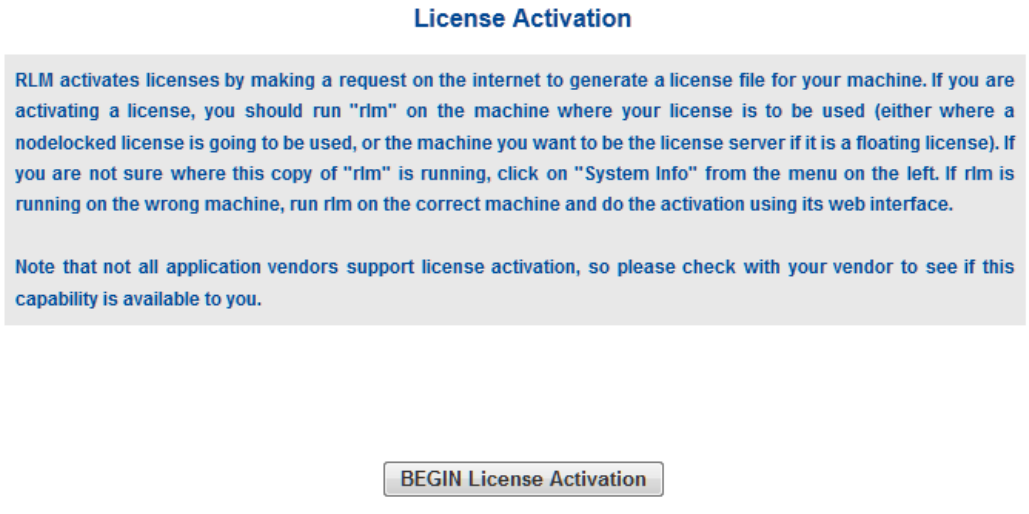
RLM System Info

If you select *System Info* from the menu on the left, you will see the RLM system information screen below in the view area. 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). In addition, starting in RLM v9.3BL2, this screen will 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).



Activate License

If you select *Activate License* from the menu on the left, you will be prompted for the information necessary to activate a license from an ISV server's internet site. The information is collected in several steps. The initial screen is shown below. Click on **BEGIN License Activation** to step through the screens which will collect the data required to activate a license.



RLM Manual/About...

These last two commands display RLM information. The **RLM Manual** button displays the latest version of this manual from the Reprise Software, Inc. website at http://www.reprisesoftware.com/RLM_License_Administration.html

The **About...** command displays the intro screen seen at the beginning of this section.

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.

RLM privileges controlled by the RLM options file

Privilege	Name to use in RLM options file	Meaning
edit_meter	edit_meter	Allows modifying count for meter counters
edit_options	edit_options	Allows editing options files for ISV servers
edit_rlm_options	edit_rlm_options	Allows editing options files for the rlm server and license files
edit_xfer	edit_xfer	Allows editing server-server license transfer settings for ISV servers
logfiles	logfiles	Enables the functions which change log files - switch, switchr, newlog
manage_service	manage_service	Allows editing windows service setup
remove	remove	Allows the user to remove a license from a running process
reread	reread	Allows access to the functions which do reread commands on license servers
shutdown	shutdown	Allows access to the functions which shut down license servers
status	status	Allows display of status and debug log information from the license servers

The RLM options file syntax is a subset of The ISV Options File syntax. The **privilege** names **status**, **reread**, **shutdown**, **logfiles**, **edit_meter**, **edit_options**, **edit_rlm_options**, **manage_service** and **edit_xfer** are used where a product name would be used in an ISV options file. By default, all privileges are granted to all users unless otherwise restricted in the rlm options file.

A user with no privileges assigned will have access to the "Activate License", "Diagnostics", "RLM Manual...", "System Info", and "About" commands.

Note that the RLM web interface does not have access to the username or hostname (the rlmutil utilities do pass the username and hostname), so, to be most effective, command restrictions should be done based on IP addresses. By default, all commands are enabled (unless disabled with the -x rlmdown or -x rlmremove rlm startup options, in which case rlm options have no effect.).

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 ("). Note: single quote (') and back-quote (`) were illegal prior to RLM v8.0.

Note that 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.

The ACTIVATE option controls whether the "Activate License" button is present, and, if present, the default activation URL and ISV name.

There is one additional option available in the ISV options file: NO_OLD_RLMUTIL. 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 everything in the RLM options file is case-insensitive.

The RLM Options File first appeared in RLM v4.0. The remove privilege and NO_OLD_RLMUTIL were added in RLM v9.0. The logfile privilege was added in RLM v9.1. The EDIT_METER privilege was added in RLM v9.3.

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. Note that 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 that **privilege** should be one of "**status**", "**shutdown**", "**reread**", "**edit_options**", or "**edit_rlm_options**":

ACTIVATE [off | url URL | isv ISVNAME]

The ACTIVATE line allows you to disable the "Activate License" command, or to set the defaults for the URL and ISV name for activation.

The 3 forms of the ACTIVATE line are:

ACTIVATE off
ACTIVATE url URL
ACTIVATE url ISVNAME

In the first form, the "Activate License" menu item is disabled and will not appear in the menus.

In the second form "URL" is the default URL used for activation. For example:
ACTIVATE url www.reprisesoftware.com

In the third form "ISVNAME" is the default ISV name used for activation. For example:
ACTIVATE isv reprise

ACTIVATE is new in RLM v10.0

EXCLUDE *privilege* [user|host|group|host_group|internet|project] *who*

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 '*' which matches any address, e.g. 172.16.7.*

For a list of the *privileges* available, see the table at the beginning of this chapter.

EXCLUDEALL [user|host|group|host_group|internet] *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 GROUP or HOST_GROUP line in the RLM options file.

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

For a list of the *privileges* available, see the table at the beginning of this chapter.

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. Prior to RLM v9.0, only the first definition of a particular group name is used. Starting in RLM v9.0, multiple lines that specify the same GROUP name will have their lists of usernames concatenated.

Example:

GROUP engineers tom dick harry

Example (rlm v9.0+). This example results in a group of 6 users:

GROUP engineers tom dick harry

GROUP engineers larry curly moe

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. Prior to RLM v9.0, only the first definition of a particular group name is used. Starting in RLM v9.0, 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

Example (rlm v9.0+). 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

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.*

Example 2. 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

INCLUDE *privilege* [user|host|group|host_group|internet] *who*

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 defi*ned by pr*ivilege.

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

For a list *of the pri*vileges available, see the table at the beginning of this chapter.

INCLUDEALL [user|host|group|host_group|internet] *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.*

For a list of the *privileges* available, see the table at the beginning of this chapter.

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.

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.

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 (RESERVE), and to either allow (INCLUDE or INCLUDEALL) or disallow (EXCLUDE or EXCLUDEALL) license use. Additionally, options are provided to create groups of users (GROUP) or hosts (HOST_GROUP), or IP addresses (INTERNET_GROUP), and to control the maximum number of licenses a user or group can check out (MAX).

Logging of the license servers is controlled by the (DEBUGLOG) and (REPORTLOG) options. In addition, you can suppress debug logging of checkin/checkout/denied entries with the NOLOG option. Automatic report log file rotation is supported in ISV servers via the ROTATE option (new in RLM v4.0).

Control over license roaming is provided with the INCLUDEALL_ROAM, EXCLUDEALL_ROAM, ROAM_MAX_DAYS, and ROAM_MAX_COUNT options.

License timeout is controlled via the TIMEOUT and TIMEOUTALL options.

License timezone restrictions are controlled with the TIMEZONE option.

Client license caching is controlled with the CLIENT_CACHE option.

License queuing behavior can be modified with the EXPRESS and PRIORITY options.

There is one additional option available in the ISV options file: NO_OLD_RLMUTIL. 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.

The ISV options file is specified on the ISV line in The License File.

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.

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 ("). Note: single quote (') and back-quote (`) were illegal prior to RLM v8.0.

Note that 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.

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

Note that 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 the section below.

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.

Examples:

CLIENT_CACHE 0

- disables client caching for all licenses

This is logged:

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

CLIENT_CACHE 200 foo

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

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.

CLIENT_CACHE first appeared in RLM v10.1

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.

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 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.

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.

Alternately, 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.*

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: *internet* and *project* were added in RLM v2.0. *noproject* was added in RLM v5.0. *internet_group* was added in RLM v10.0. The id option was added in RLM v11.1

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.*

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

Note: *internet* and *project* were added in RLM v2.0. *noproject* was added in RLM v5.0.

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: internet and project were added in RLM v2.0. *internet_group* was added in RLM v10.0

EXPRESS ON|OFF [*product*]

The EXPRESS line controls queuing behavior for one or all *product*s. 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.

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. Prior to RLM v9.0, only the first definition of a particular group name is used. Starting in RLM v9.0, multiple lines that specify the same GROUP name will have their lists of usernames concatenated.

Example:

GROUP engineers tom dick harry

Example (rlm v9.0+). This example results in a group of 6 users:

GROUP engineers tom dick harry

GROUP engineers larry curly moe

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. Prior to RLM v9.0, only the first definition of a particular group name is used. Starting in RLM v9.0, 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

Example (rlm v9.0+). 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

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.*

Example 2. 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

INCLUDE *product* userhost|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.*

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

Note: internet and project were added in RLM v2.0. *internet_group* was added in RLM v10.0 The id option was added in RLM v11.1

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.*

Note: internet and project were added in RLM v2.0. *internet_group* was added in RLM v10.0

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.*

Note: internet and project were added in RLM v2.0. *internet_group* was added in RLM v10.0

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.*

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

Note: internet and project were added in RLM v2.0. *internet_group* was added in RLM v10.0
The id option was added in RLM v11.1

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.

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.

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 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”.

Note: PRIORITY was added in RLM v2.0. *internet_group* was added in RLM v10.0. The id option was added in RLM v11.1

REPORTLOG [+]*file_path* [std | small | detailed] [auth]

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 checkin/checkout events down to the tenth of a millisecond

The fourth optional argument, if present, specifies that the reportlog is to be authenticated. This parameter should be the string **auth**, and if it is to be used, the third parameter (reportlog format) must be present as well. Note that this parameter is no longer required in RLM v4.0, as all reportlogs are authenticated.

If your server is writing a reportlog, it is important to shut the server down gracefully (ie, 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 sever 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 Appendix A - Reportlog File Format on page 94.

Note - the authentication parameter first appeared in RLM v3.0.
Note - the authentication parameter is no longer required in RLM v4.0 - all report logs are authenticated.
Note - prior to RLM v9.3, if a detailed reportlog was specified on Windows, the std format was used. In v9.3, rlm will produce a detailed reportlog, however, the fractional seconds field will always be 0 on Windows.

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.*

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

Note: internet and project were added in RLM v2.0. *internet_group* was added in RLM v10.0. The id option was added in RLM v11.1.

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.

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*. Note that if you specify ROAM_MAX_DAYS for the *rlm_roam* license, this will limit roaming on all products to the number of days specified.

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.

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.

Note that 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.)

The ROTATE option first appeared in RLM v4.0.

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 that 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.

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 that a license could have a min_timeout specification, in which case if you specify a TIMEOUTALL value that is lower, the license's 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.

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 prod

would cause the second TIMEZONE option (for product *prod*) 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.

Section 3 – Advanced Topics

This section of the manual contains topics that may be of use if you or your ISV are doing a more advanced implementation of licensing.

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 customer 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 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 four ways:

- The count field contains one of the 3 token keywords (**token**, **token_locked**, 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 option processing is different for *token-based* licenses. See below.

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 sig=xxxx token="<dollars 2.0 5>"
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 sig=xxxx token="<dollars 2.0 5> <cents 3.4 53>"
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 token-based* license is checked out, the RLM functions return information about the first license in the list only. In this example, RLM functions would return information about the *dollars* license.

The License Count Keywords

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

- token
- token_locked
- 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 that *token* and *token_unlocked* are 100% equivalent.

token_locked means that the *token-based* license includes the license server hostid in it's signature, and is valid only in *this* license file.

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 license turns into checkouts of:

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

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 that 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.

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.
- Queuing is not allowed for *compound token-based* licenses.

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 heartbeat. In this sense, the *token-based* license works just like a regular, non *token-based* license.

Report Log

When a *token-based* license is checked out, the name of the license requested (and it's version) is logged in the checkout record for each resulting *primary* license checkout.

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.

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 it's license if it is not able to check the license out an 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 that this capability depends on your Software Provider having coded their application to handle the QUEUED status return from the license server. Note that if you have set **RLM_ROAM**, the setting of **RLM_QUEUE** is ignored.

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 the EXPRESS option in The ISV Options File.

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.

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.

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 node-locked *rlm_roam* license in it.

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 it's 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.

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**.

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.

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: Starting in RLM v4.0, **RLM_ROAM** no longer needs to be set on the disconnected system. Note that 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.

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 checkin), the roamed license will be returned to the license pool on the server. Please note that 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.

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).

Failover License Servers

RLM provides the capability for a license server to take over the license compliment of another server which has gone down. This server which takes over the license load of the failed server is called the *failover license server*. The server whose licenses are being taken over is called the *primary 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. This is an unsupported configuration.

The ability for a server to take over the load of another server is selected on an ISV-by-ISV basis, and enabled by an *rlm_failover* or *rlm_failover_server* license in the license file of the server that is to take over.

While the *failover license server* can be serving it's own compliment of licenses, Reprise Software recommends that it should have no licenses of it's own, but simply be standing by, waiting for one (or more) other server(s) to fail so that it can take over.

In order to enable a failover license server, the ISV issues you an *rlm_failover* or *rlm_failover_server* license with the following characteristics:

- count=some non-zero value
- hostid=<hostid of the *primary server*>
- _failover_host=<hostname of the *primary server*> in the case of an *rlm_failover* license, OR
- _failover_host=<port@host of the *primary server rlm* process> in the case of an *rlm_failover_server* license.

(Note: beginning in RLM v9.2, the **_failover_host** keyword has been replaced with the **_primary_server** keyword. **_failover_host** will continue to work, but new software installations should use **_primary_server** instead.)

The difference between an *rlm_failover* license and an *rlm_failover_server* license is whether rlm checks for the machine being down (*rlm_failover*) or the rlm process on the machine being down (*rlm_failover_server*) before serving the licenses from the failed server.

Note: Reprise Software recommends that ISVs set _primary_server (_failover_host) to **localhost** (or **5053@localhost**) when issuing the *rlm_failover or rlm_failover_server* license. This value would be changed by you, the License Administrator, to the name of the *primary server* upon installation. Setting this to **localhost** insures that the *failover license server* will not be activated by mistake if the license is not edited.

The *rlm_failover* license must be a **counted** license; otherwise the server will not use it. (If it were not a counted license, the *rlm_failover or rlm_failover_server* license can be used on any server, which could result in multiple servers taking over **at the same time** for the *failed server*.)

In order to enable the *failover license server*, the *rlm_failover* (or *rlm_failover_server*) license needs to be in one of the license files it is using, and the license file(s) for the *failed server* also need to be processed by the *failover license server*.

When a license server encounters a *rlm_failover* or *rlm_failover_server* license, it does several things:

- Starts a separate thread in the license server to periodically monitor the health of the *failed server*. This is done by attempting TCP/IP connection(s) to ports on the *failed server* in order to determine whether the server is up or down. No attempt is made to determine whether the license server is up - if the computer is up, the *failover license server* will not take over serving licenses.
- If the *primary server* should go down, enables all licenses in license files for the *failed server* by performing the equivalent of an *rlmreread* command.
- When the *failed server* comes back up, disables all licenses in license files for the *failed server* by performing the equivalent of an *rlmremread* command.

Configuring *Failover License Servers*

Reprise Software recommends configuring *failover license servers* as stand-alone servers that do not serve their own compliment of licenses. In other words, Reprise recommends configuring a license server that has only *rlm_failover* (and/or *rlm_failover_server*) 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 that 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 failed 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 "main_server"):

```
HOST failover_host 11111111 1700
ISV reprise
LICENSE reprise rlm_failover 1.0 permanent 1 hostid=12345678
_primary_server=main_server sig="x"
```

Alternate LICENSE line:

```
LICENSE reprise rlm_failover_server 1.0 permanent 1 hostid=12345678
_primary_server=5053@main_server sig="x"
```

Note that the hostid in the LICENSE line for the *rlm_failover* or *rlm_failover_server* license is the hostid of the server on the node **main_server**).

Installing *Failover License Servers*

When you receive the *rlm_failover* license file, do the following to install your *failover license server*:

1. Install *rlm* and your *ISV server* on the *failover license server* node.
2. Install all license files from the *primary server* on the failover license server node.
3. Edit the license file with the *rlm_failover* or *rlm_failover_server* license to put the hostname of the *primary server* in the *_primary_server=* (or *_failover_host=*) field, and the hostname of the *failover license server* on the HOST line. (Put the port@host of the rlm process on the *primary server* in the *_primary_server=* (or *_failover_host=*) field for the case of *rlm_failover_server* licenses).
4. Install any ISV options in an options file, and make it accessible to the *ISV server* either through it's license file or by giving it the default options filename of *isv.opt*.
5. Insure that *rlm* will process all the license files above, and start *rlm*.
6. Modify your user's RLM_LICENSE environment variables to include *port@host* of the *failover license server*. OR be sure to include the *failover license server's license file* where your application will find it. If you omit this step, the *failover license server* will take over, but your application will not be able to check out a license from the *failover server*.

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 it's 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.

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 definition**s. 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 isvname
```

(Note that 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:

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 communications 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 it's 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 it's 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.

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.

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 has to be up to return the license, anyway.

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.

License Transfer is new in RLM v7.0. ISV-defined transfer types are new in RLM v7.0BL4. Disconnected (roamed) transfers are new in RLM v10.0.

Section 4 – Reference Material

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.

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 that 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.

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.

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.

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.

RLM_DIAGNOSTICS was introduced in RLM v8.0.

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. RLM_EXTENDED_ERROR_MESSAGES first appeared in RLM v3.0.

RLM_LICENSE

The RLM_LICENSE variable specifies the path to one or more license files and/or license servers. See The License Environment on page 36 for a complete description of how to use RLM_LICENSE.

RLM_LICENSE_PASSWORD

The RLM_LICENSE_PASSWORD variable specifies a password for access to particular licenses which have the _password attribute. See The License File on page 19 for a complete description of how to use RLM_LICENSE_PASSWORD.

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.

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. RLM_PATH_RANDOMIZE first appeared in RLM v2.0.

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.

RLM_QUEUE

RLM_QUEUE informs an application that it should queue for a license is none are available. If RLM_QUEUE is set, any subsequent application started will queue for it's license if none are available. RLM_QUEUE first appeared in RLM v1.1. If RLM_ROAM is set, RLM_QUEUE is ignored (starting in RLM v12.0).

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).

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

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 first appeared in RLM v1.1.
RLM_DISCONNECTED was removed in v3.0, as it was no longer used.

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**.

To download the RLM License Administration bundle, go to the Reprise Website Download area, select the License Administration bundle, click "I agree" to the license terms, 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:

Platform	OS Version	Platform Name	Kit file name
HP-UX on PA-Risc	HP/UX 11.0 or later, 32-bit	hp_h1	hp_h1.admin.tar.gz
HP Itanium	HP/UX Itanium, 64-bit	ia64_h1	ia64_h1.admin.tar.gz
IBM AIX	IBM AIX RS/6000	ibm_a1	ibm_a1.admin.tar.gz
Linux on Intel X86	Linux, 32-bit	x86_l2	x86_l2.admin.tar.gz
Linux x64	Linux, 64-bit	x64_l1	x64_l1.admin.tar.gz
Linux Itanium	Linux Itanium, 64-bit	ia64_l2	ia64_l2.admin.tar.gz
Linux PPC	Linux PPC, 64-bit	ppc64_l1	ppc64_l1.admin.tar.gz
Mac on Intel X86	Mac OS 10.4.11 or later, 32-bit	x86_m1	x86_m1.admin.tar.gz
Mac on PPC	Mac OS 10.4 or later. 64-bit	ppc_m1	ppc_m1.admin.tar.gz
NetBSD	NetBSD v4.0 or later, 32-bit	x86_n1	x86_n1.admin.tar.gz
Solaris on Intel	Solaris 10 or later, 64-bit	x64_s1	x64_s1.admin.tar.gz
Solaris on Sparc	Solaris 9 or later, 32-bit	sun_s1	sun_s1.admin.tar.gz
Windows on Intel X86	Windows 7, 8, Vista, XP, Server 2003, 2008, 2012 (32-bit)	x86_w1	rlm.vx.yBLz.admin.exe

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_w1 for 32-bit or x64_w1 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.

The Tests

rlmtests performs 2 categories of tests:

- checkout performance tests
- server capacity tests

The checkout performance tests consists 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-processes all succeed.) During the course of this test, the checkout performance time is reported as each sub-process completes.

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* is the number of simulated clients which each subprocess will run in the server capacity tests. The maximum value is 2000.

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 checkin records, as described below. Common to all formats are the classes of data logged, and the data for all except checkin 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 (added in RLM v3.0)
- checkin
- checkout
- dequeue (added in RLM/reportlog v1.1)
- isv-specific data (added in RLM v2.0)
- 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 (added in RLM/reportlog v1.1)
- roam extend
- server shutdown
- server reread of license/option file
- support for a product (feature) - one for each *license pool* at log file start time

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. This is new in RLM v9.0.

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, checkins, 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.

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.

checkin

"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. This field is new in reportlog v1.1

The *why* parameter is one of:

why value	Reason
1	"Normal" checkin by application
2	Application exited, automatic checkin
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 it's transferred licenses
10	Meter ran out of count during a periodic decrement
11	Client failed to send heartbeat within “promise” interval.

Note: *why* values of 4, 5 and 6 are new in v1.1, value 7 is new in RLM v5.0, values 8 and 9 are new in RLM v7.0. 10 is new in RLM v9.3.

The *cur_use* and *cur_resuse* fields indicate the current number of free licenses in use and reservations in use after this checkin. These fields are new in reportlog v2.0.

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”.

On Windows, *tenths_of_msec* will always be 0.

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

"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. This field is new in reportlog v1.1

The *cur_use* and *cur_resuse* fields indicate the current number of free licenses in use and reservations in use after this checkout. These fields are new in reportlog v2.0.

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. This field is new in reportlog v1.1. Note that 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. This data is new in reportlog v2.0 (RLM v2.0).

The *process_id* field is the PID of the process requesting the license. This data is new in reportlog v2.0 (RLM v2.0).

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:
"x86_w2 5.1" - a windows system running Windows XP.

This field is new in reportlog v8.0 (RLM v8.0).

The *application argv0* field is the argv[0] of the product requesting the license. This field is new in reportlog v8.0 (RLM v8.0).

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. This field is new in reportlog v9.0 (RLM v9.0). 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 (ie, 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). This field is new in reportlog v9.0 (RLM v9.0)

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.

The dequeue record is new in v1.1. Note that the dequeue record is identical to the checkin 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.

On Windows, *tenths_of_msec* will always be 0.

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.

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 RLM Status Values on page 111.

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 *isv_def* field contains the value of the optional ISV-defined field. This field is new in reportlog v1.1

The detailed format is new in reportlog v4.0. Prior to v4.0, it was identical to the standard and small formats.

The *pid* field is new in reportlog v11.3.

On Windows, *tenths_of_msec* will always be 0.

license in use

All formats:

- INUSE product version pool# user host "isv_def" count server_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.

log file start

All formats:

- 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
- 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**". (Note: Authentication first appeared in RLM v3.0, reportlog format 3.0. Starting in v4.0, 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 LICENSE FILE line(s) are new in reportlog v2.0. There will be one LICENSE FILE line for each license file which the server is processing.

log file end

All formats:

- SWITCH to *filename* (if an rlmswitch was done)
- END mm/dd/yyyy hh:mm

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.

meter decrement records are new in RLM v9.3.

periodic timestamp

All formats:

- TIMESTAMP mm/dd/yyyy hh:mm
  Timestamps are performed by the ISV servers every night just after midnight.
  Beginning in RLM v9.3, timestamps are added to the log file every 30 minutes

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.

Note: the queue message is new in RLM v1.1 (reportlog v1.1).
Note: the "requested product" and "requested version" fields were added in RLM v5.0 (reportlog v5.0).

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.

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.

Roam extend records are new in RLM v10.0.

server shutdown

All formats:

- SHUTDOWN *user host* mm/dd hh:mm:ss

server reread of license/option file

All formats:

- REREAD *user host* mm/dd hh:mm:ss

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*

NOTE: The *soft_limit, contract, customer, and issuer* fields are new in the v1.1 reportlog. The *hostid* and *pool#* fields are new in the v2.0 reportlog. The *line_item* field is new in the v3.0 reportlog. The *options*, *share*, *max_share*, *type*, and *named_user_count* are new in the RLM v5.0 reportlog. The five *meter_* parameters are new in RLM v9.3.

count is the number of free licenses (ie, 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.

Reportlog version change history

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

RLM hostids

RLM supports several different kinds of identification for various computing environments, as well as some generic identification which are platform-independent.

RLM's host identification (hostid) types are:

hostid 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=WD-WX60AC946860	Windows only
gc	Google Compute Engine	gc=3797742226458986650.k6qt9v5h38w2adwqgc9fdhdf3w0m761p	Linux only. Introduced in RLM v11.1 See note below.
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="
ether	Ethernet MAC address	hostid=ether=00801935f2b5	always printed without leading "ether="
rlmid1	External key or dongle	rlmid1=9a763f21	External key or dongle
user	User name	hostid=USER=joe	
host	Host name	hostid=host=melody	

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

Note: The RLMID series of hostids are optional products, and will often require other software to be installed on the system on which they are to be used. For these devices, see Optional hostid Installation Instructions on page 106.

Note: Beginning in RLM v3.0, IP address hostids can contain the wildcard ('*') character in any position to indicate that any value is accepted in that position.

The serial number (sn) and rlmid1 hostid types first appeared in RLM v5.0.

A note about Google Compute Engine, and the gc= hostid type

license servers cannot serve nodelocked licenses that are locked to gc hostids

The gc= hostid type is determined via a call to an http server. As such, RLM makes an effort to avoid the calls which could take a while to process if the application is not running on google compute engine (Note that RLM first attempts to determine that it is running on google compute engine before making the http calls, but this determination can yield a false positive). This means that RLM clients will not attempt to determine the gc hostid unless they are processing a nodelocked license that is locked to this hostid type. The practical result of this is that ***license servers cannot serve nodelocked licenses that are locked to gc hostids,*** since the client will not transmit this hostid type to the server.

If the RLM algorithm to determine that it is running on google compute engine does not detect google compute engine for any reason, you can set the RLM_GOOGLE_CLOUD environment variable (to any value) to indicate to RLM that it is running on Google Compute Engine.

Optional hostid Installation Instructions

Certain hostids in the RLMID family (RLMID1) require device-specific installation on the target computer.

Installing RLMid1 Devices

The RLMid1 device is a hardware key manufactured by Aladdin Knowledge Systems (now Safenet, Inc). Installation on a target system can be accomplished in two ways:

- use Windows "Found New Hardware" to automatically load the drivers (preferred), or
- use the RLMID1 driver installer (from the Reprise Software website) to do the driver installation

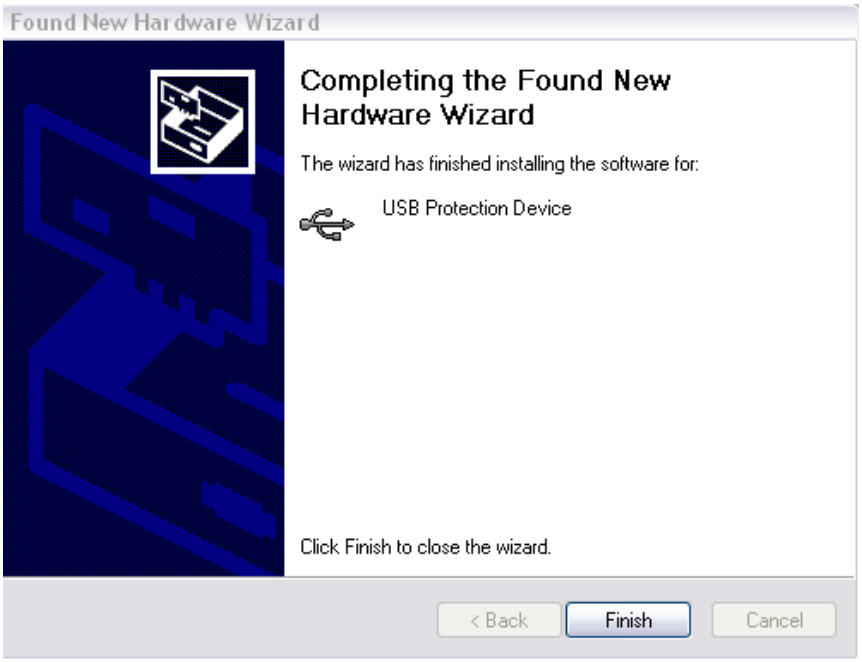
Using Windows to automatically install drivers

In order to use Windows to automatically do the driver installation, simply plug the device into the computer, and Windows will detect the new device. If you first get a permissions screen asking if it is OK to use Windows Update to locate the driver, indicate that permission is granted.

You will get the "Found New Hardware" wizard which will install the drivers for the "USB Protection Device" for you. (Note: the device may alternately be called "HASP HL 3.xx" or "Aladdin USB Key" or "Safenet Inc. USB Key").

Select "Install the Software Automatically (recommended)", and click "Next". Windows will locate the driver and install it.

You will then get the "Completing the Found New Hardware Wizard", shown below; click "Finish".



That is all there is to it.

Installing the drivers using the installation program.

If for some reason Windows fails to update the driver automatically, or if the target system is not connected to the internet, use the driver installer located at:

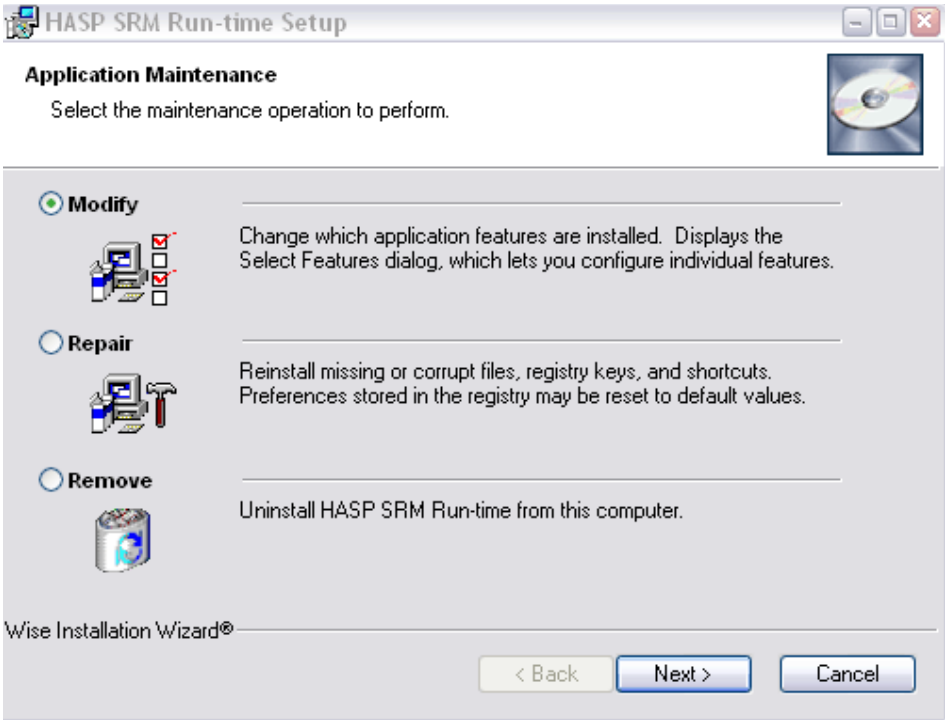
http://www.reprisesoftware.com/drivers/rlmid1.zip

Or you can download the driver directly from the SafeNet site at:

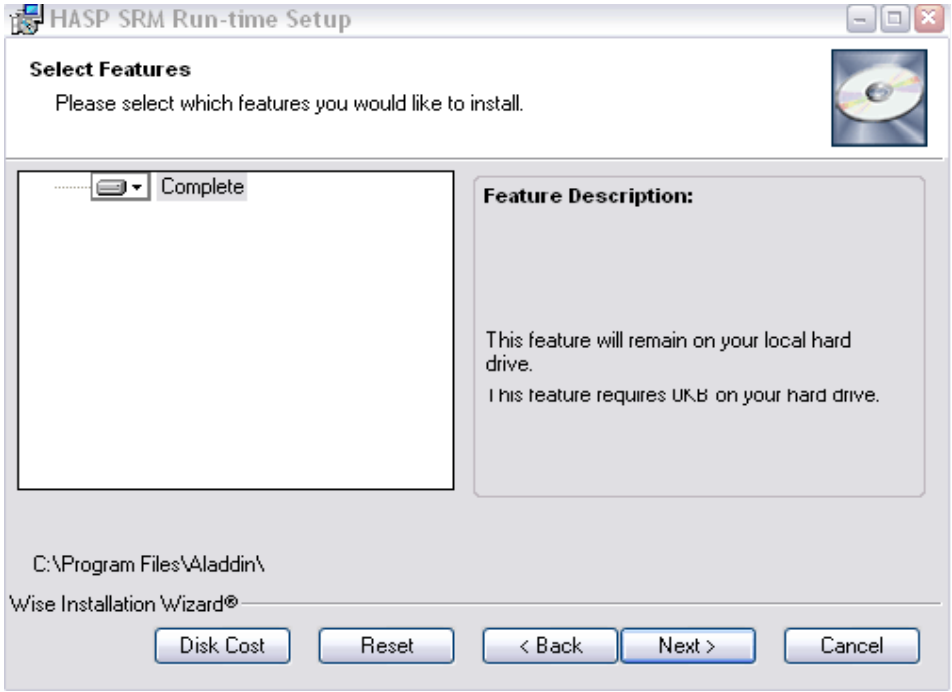
ftp://ftp.aladdin.com/pub/hasp/Sentinel_HASP/Runtime_%28Drivers%29/Sentinel_HASP_Runtime_setup.zip

To run the installer:

- save the installer (rlmid1.zip) to disk
- extract the zip file
- navigate into the rlmid1 directory, and run the RLMID1 installer application
- when the installer comes up, leave "Modify" selected (or select it) and press "Next->"



- On the "Select Features" screen, click "Next->"



- On the "Ready to Modify the Application" screen, click "Next->"

You will now see the "HASP SRM Run-time has been installed" screen. Click "Finish".

The drivers are installed and you are ready to use the RLMID1 devices.

Note that an RLMID1 device can be used by any RLM-licensed application on the system, in other words, there is nothing ISV-specific about the device.

Installing RLMid1 Devices on Linux

To install the necessary drivers for RLMid1 devices on linux, follow these steps:

1. Browse to http://sentinelcustomer.safenet-inc.com/sentineldownloads/. Look for the "Sentinal HASP/LDK Rutime Installer" for Linux. There are several options, depending on Linux variant and the style of installer you want (GUI, RPM, script).

2. Download the appropriate installer and install. Note that you will have to execute the installation as root.

   The runtime installer sets up a daemon that is used to access the hardware key.

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 and x64_w3 kits, 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 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 an ISV server built with a _w3 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 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.

RLM Status Values

General Licensing Errors:

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
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_DEMO_EXISTS	-137	Demo already created on this system
RLM_EH_DEMO_WRITEERR	-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

Internet Activation 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_FUFILL	-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 (Activation Pro)
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)
RL_ACT_DB_READERR	-1026	Activation database read error (Activation Pro)
RLM_ACT_GEN_PARAM_ERR	-1027	Generating license - bad parameter
RLM_ACT_UNSUPPORTED_CMD	-1028	Unsupported command to license generator

License Checkout Errors:

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	The start date in the license hasn't occurred yet,

		in the future	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 license soft_limit	This license checkout causes the license usage to go over it's soft limit. The checkout is successful, but usage is now in the overdraft mode.
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[tm] license has expired.
RLM_EL_FAILED_BACK_UP	-50	Failed host back up - failover server released license	If you 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	RLM_EL_BAD_PASSWORD is an internal error and won't ever be returned to the client - if

		product	the license password is bad, the client will receive RLM_EL_NO_SERV_SUPP
RLM_EL_METER_NO_SERVER	-53	Metered licenses require server	Metered licenses only work with with a license 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_EH_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 Version History

V11.3 - April, 2015

Features Added
Application PID added to DENY records in the reportlog
In rlm web interface and rlmstat, the field previously labeled "transactions" is now labeled "checkouts"

V11.2 - November, 2014

Features Added
Version restrictions on roaming data have been relaxed to allow the “oldest compatible” roam file to be used. As of 11.2, the oldest compatible roam file is 11.0
Client-cached licenses can now be roamed before the end of the cache period.

V11.1 - June, 2014

Features Added
RLMid1 dongles now supported on linux (x86_l2, x64_l1)
Setting RLM_ROAM=today causes a roam to end today at midnight
Licenses now have an optional _id= parameter to identify for options file use
For metered licenses, the user can now decrement the counter in the rlm web interface.
RLM now supports Google Compute Engine hostids (gc=)

V11.0 - February, 2014

Features Added
Ipv6 support

akey= license attribute
Windows clients auto-detect proxy servers

V10.1 - July, 2013

Features Added
CLIENT_CACHE license admin option added
Server logs unreadable license files

V10.0 - Jan, 2013

Features Added
Disconnected server-server license transfer
RLM will now broadcast to find a server on the local network.
Roaming is disabled if the license server uses a transient hostid
rlm web interface now supports user login, with access rights.
rlm web interface only displays commands which the user can execute.
rlm web interface doesn't display “Manage Windows Service” on non-Windows systems
If rlm processes multiple license files, it will attempt to find a good ISV server path
Browsers connecting on rlm's main port are redirected to the webserver port
Report log logs all licenses in use both at start and at the end.
Roamed license time extension logged in report log (and debug log)
RLM web interface allows editing license files
RLM checks that the debug log is writable when installing service
INTERNET_GROUP option

v9.4 - July, 2012

Features Added
Hostname hostid types now accept wildcards
RLM utilities now accept the -z password option
When installing RLM as a service on Windows, the installation now starts and stops the service to trigger firewall prompts.

v9.3 – February, 2012

Features Added
Client-side diagnostics now list all embedded string licenses in addition to other node-locked licenses.
Server-side diagnostics now output the rlm and isv server option file info.
RLM now enumerates the ethernet devices on linux rather than using eth0-7

v9.2 - September, 2011

Features Added
disksn hostid (disk hardware serial number) added on Windows
License Passwords can now be specified on the ISV line
"_primary_server" keyword added for rlm_failover licenses

v9.1 - May, 2011

Features Added
Passwords on individual LICENSE lines
disable=TerminalServerAllowRD attribute
The LOGFILES privilege has been added to the RLM options file

v9.0 - December, 2010

Features Added
rlmstat reports on expiration dates
New license checkout debugging capability/utility
Multiple GROUP lines now concatenate in OPTIONS files
Checkout records in the debug log now contain information on roaming licenses)
RLM servers log information about which licenses were replaced (in the debug log
new ISV line format with optional keyword=value parameters
The REMOVE privilege has been added to the RLM options file.
rlm and ISV servers can now disable older versions of rlmutil

v8.0 - Jan, 2010

Features Added
Optimized license sharing
client and server side diagnostics to aid solving problems
When running as a service, rlm changes working directory to binary directory
rlm logs the client machine's OS to the report log
rlm logs the client's argv[0] to the report log
rlm web interface shows all license file and log file paths
rlm web interface puts all activated license files into the directory specified with -c
single-quote and back-quote characters are now legal in license and option files
max_roam_count license keyword

v7.0 - June, 2009

Features Added
Server-Server license transfers
failover servers no longer pool licenses from failed servers
license line checksum (_ck=)
RLM_EL_FAILED_BACK_UP status when failed server restarts

v6.0 - January, 2009

Features Added
Platform-independent ISV server settings and the Generic ISV server
rlmid2 hardware key
ISV servers increase their open file limit
ISV lockfile in C:\rlm removed
port@host can be specified as host@port
RLM_LICENSE environment and the -c option can contain directories
RLM default port # changed from 28000 to 5053
RLM admin port # changed from 9000 to 5054
UPGRADE licenses
min_checkout

v5.0 - May, 2008

Features Added
Serial Number hostid type

rlmID1 hardware key
hostid lists
ISV servers don't exit on reread if no license file exists
Virtual machine detection in ISV servers
disable= now accepts VM keyword to disable licenses on Virtual Machines
refresh buttons added to web interface
options= license attribute
multiple instances of a single ISV-defined hostid type allowed
license administration NOPROJECT keyword for EXCLUDE and EXCLUDEALL
At least 5 IP addresses now supported for hostids (previously only one)
The rlmver command-line utility has been removed

v4.0 - December, 2007

Features Added
report log anonymizer (rlmanon) added
rlm web interface allows editing option files
rlm web interface displays debug log
report log detailed format adds seconds, tenths of seconds for Denials
Automatic report log rotation
rlm options file controls access to administration functions
RLM web interface displays recent debug log information
RLM web interface allows editing server options file
RLM_ROAM no longer needs to be set on the disconnected system
-c overrides RLM_LICENSE for rlmutil
Named User licensing
disable=TerminalServer license attribute
multiple ethernet device support on linux and mac
ethernet address is default hostid on linux and mac
Windows volume serial number hostid added
Windows volume serial number is default hostid

v3.0 - June, 2007

Features Added
Internet Activation
rlm -dat command-line option
rlmtests performance tests
rlm servers ignore hostnames in license file

The rlm web interface now reports the Process ID (PID) of licenses in use
rlm logs status requests in the debug log
client node can access license server by any name
ISV server pathname optional on ISV line
RLM_CONNECT_TIMEOUT environment variable
RLM_EXTENDED_ERROR_MESSAGES environment variable
maximum license share count
_line_item license keyword
license in a string
improved error messages in web interface and rlmsign
PID of process using license is displayed in web interface
Wildcards allowed in IP addresses used as a hostid

v2.0 - Dec, 2006

Features Added
Failover License Servers
Token-based licensing
user/host based licenses
Nodelocked, single-use licenses (no server)
options to disable rlmdown and rlmremove
RLM_PATH_RANDOMIZE environment variable
ISV servers notify of licenses expiring within 14 days
rlm binds all TCP/IP ports in all license files
rlm -c license_file command-line option
rlm runs as a service on Windows
rlmstat -avail reports on license availability
transient attribute on ISV-defined hostids
System Info in rlm web interface
min_remove license keyword
rlm_products() API call
rlm_log(), rlm_dlog() API calls
PRIORITY license administration option
TIMEZONE license administration option
MAX accepts '*' for all users
license administration license management by PROJECT
MINREMOVE license administration option

v1.1 - July, 2006

Features Added
Held licenses
Shared licenses
License Replacement
License timeout
Roaming licenses
Intelligent license queuing
ISV-defined hostids
contract= license attribute
customer= license attribute
issued= license attribute
issuer= license attribute
platforms= license attribute
soft_limit= license attribute
start_date= license attribute
timezone= license attribute
type= license attribute

v1.0 - May, 2006

This version contains the basic RLM functionality:

Features
Node-locked licenses
Floating licenses
Expiration dates
Transparent multiple server connections
Public-Key authentication

Revision History

v11.3 – April, 2015 – v11.3 release (BL1)

v11.2 – November, 2014 – v11.2 release (BL2)

v11.1 – June, 2014 – v11.1 release (BL2)

v11.0 – Feb, 2014 – v11.0 release (BL2)

v10.1 – July, 2013 – v10.1 release (BL2)

v10.0 – Jan 16, 2013 – v10.0 release (BL2)

v9.4 – July 24, 2012 – v9.4 release (BL2)

v9.3 – February 15, 2012 – v9.3 release (BL2)

v9.2 – 28-Sept-2011 - v9.2 release (BL2)

v9.1 – 2-May-2011 - v9.1 release (BL3)

v9.0 – 15 -Dec-2010 - v9.0 release (BL2)

v8.0 - 26-Jan-2010 - v8.0 release (BL3)

v7.0 - 11-Jun-2009 - v7.0 release (BL3)

v6.0 - Jan-2009 - v6.0 release (BL2)

v5.0 - 1-Jul-2008 - v5.0 release (BL2)

v4.0 - 18-Jan-2008 - v4.0 release (BL4)

v3.0 - 24-Jul-2007 - v3.0 release (BL3)

v2.0 - 13-Dec-2006 - v2.0 release (BL4)

v1.1 - 31-Jul-2006 - v1.1 release (BL4)

v1.0 - 8-May-2006 - v1.0 release