IDA Pro Lumina server Administrator Guide

Table of Contents

Last updated on December 01, 2022 — v8.2

1. Introduction

This manual describes the installation, management, and interaction with an on-the-premises IDA Pro Lumina server.

It is primarily intended for administrators, and will focus on the Lumina server component, that can be purchased for IDA Pro.

While we will (at least superficially) make use of the `lc` command-line client that is used to access/manage the server, this manual will not offer a detailed explanation of its usage: that is the role of the `lc` user manual.

We recommend having the `lc` user manual ready before starting the installation and configuration of the Lumina server.

2. Installing the Lumina server

2.1. Prerequisites

After your purchase of IDA Pro Lumina server licenses, you have received an e-mail that contains links to a download area where you will find, among other things:

- an installer for the Lumina server
- this guide
- a `lumact.key`

NOTE `ida.key` and `lumact.key` may contain the same licenses information.

All those will be necessary, so please go ahead and download them.

You will also need `root` access on the host where you will be installing the Lumina server (to install the server, not to run it).

2.1.1. MySQL server

The Lumina server stores its data in a MySQL DBMS server. It is therefore necessary to have valid credentials to such a server, as well as a fresh, empty database.

NOTE The Lumina server requires a MySQL server version 5.8 or newer.

For illustration purposes, let's assume the MySQL database the server will use, is called: `"lumina_db"`.

2.2. Installation

At installation-time, the Lumina server installer will need information about the MySQL instance the Lumina server will be using (host, port, username, password). Eventually, that information will end up in the `lumina.conf` file, sitting next to the `lumina_server_pro` binary:

```
CONNSTR="mysql;Server=127.0.0.1;Port=3306;Database=lumina;Uid=lumina;Pwd=<snipped>"
```

This chapter explains how to install the Lumina server, and create the first (i.e., administrator) user.

2.2.1. Supported platforms

The Lumina server can be installed on Linux servers. We have tested it on Debian and Ubuntu, but other major flavors of Linux should be fine too.

To install the server, run the Lumina installer as `root` and follow the instructions (the server will not require `root` permissions; only the installer does.)

TIP If your Linux system is based on `systemd` (e.g., Debian/Ubuntu, Red-Hat, CentOS, ...), it is recommended to let the installer create systemd units so that the server will start automatically at the next reboot.

2.2.2. Activating the server license

In order for the Lumina server license to be activated, it must be bound to a Host ID (an Ethernet MAC address.)

From a command prompt, run `/sbin/ifconfig`, and lookup the "ether" address for the network interface through which the server will be accessible.

```
>/sbin/ifconfig
enp4s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        [...snipped...]
        ether bf:e2:91:10:58:d2  txqueuelen 1000  (Ethernet)
```

```
        [...snipped...]
```

In this case, our mac address is: `bf:e2:91:10:58:d2`

Go to https://hex-rays.com/activate , and submit both the `ida.key` file and your MAC address. You will then receive another e-mail with instructions to download the following files:

- `lumina.crt`
- `lumina.key`
- `lumina.lic`

2.2.3. Installing the server license

Those need to be copied in the Lumina installation directory. As `root`:

```
>cd /opt/lumina
>cp .../path/to/lumina.crt .
>cp .../path/to/lumina.key .
>cp .../path/to/lumina.lic .
>chown lumina:lumina lumina.crt lumina.key lumina.lic
>chmod 640 lumina.crt lumina.key lumina.lic
```

2.2.4. Creating the initial database schema

At this point, the server should be ready to run.

CAUTION If your system is already in production and hosts files, skip this section. Using the `--recreate-schema` option as in the example below, will re-create an empty database and lose all data.

For the Lumina server to work, it needs to have a proper database schema to work with (at this point, the MySQL database (i.e., `"lumina_db"`) must already exist but is still empty.)

That is why, on the first install, you will need to initialize the database the server will use:

```
>sudo -u lumina ./lumina_server_pro --config-file lumina.conf --recreate-schema
Hex-Rays Lumina Server Teams v8.0 Hex-Rays (c) 2022
2022-09-02 10:28:30 Database has been initialized; exiting.
```

If you see "Error: Cannot connect to lumina db" please refer to troubleshooting section.

2.2.5. Testing the server

Now that the server is installed and has a database to work with, we can test that it works:

```
>sudo -u lumina ./lumina_server_pro --config-file lumina.conf \
                     --certchain-file lumina.crt \
                     --privkey-file lumina.key
Hex-Rays Lumina Server Teams v8.0 Hex-Rays (c) 2022
2022-09-22 12:14:37 Listening on 0.0.0.0:65432...
```

Good, the server appears to run! (If you are observing more worrying messages than this one, please refer to the troubleshooting section.)

At this point, you may want to either let the server run, or stop it (`Ctrl+C` will do) and restart it using systemd:

```
>systemctl restart lumina.service
```

...and make sure it runs:

```
>ps aux | grep lumina_server_pro
lumina  78812  0.0  0.0 ...
```

If you don't see a running `lumina_server_pro` process, please refer to the `systemd` diagnostic tools (e.g., `journalctl`) for more info.

2.3. Initial configuration

This chapter explains how to perform the initial configuration of the Lumina server, and in particular how to create the first (i.e., "administrator") user.

2.3.1. Creating the administrator

IMPORTANT The very first user to log into the server becomes the first administrator. S/he can create new administrators and otherwise manage the server.

Once the server is up and running, login to it using a username and password of your choice using the `lc` utility.

NOTE `lc` is the Lumina command-line administration client, which comes with the Lumina server installer. We will assume the server has been installed in `/opt/lumina/`, and thus `lc` is present in `/opt/lumina/lc`.

```
>cd /opt/lumina
>./lc -hlumina.acme.com -ualice -psecr3t info
    Hex-Rays Lumina Server v8.0
    Lumina time: 2022-09-01 14:28:02, up since 2022-09-01 14:27:58
    MAC address: <snipped macaddr>
    Client name: alice *ADMIN*
    Client host: 127.0.0.1
```

Since Alice is the first user to login to the server, the credentials she provided, will be used to create the server's primary administrator.

You can verify that you are the only user by checking the user list:

```
>./lc -hlumina.acme.com -ualice -psecr3t users
    LastActive          Adm Login License         User name Email
    ------------------- --- ----- --------------- --------- --------------
    2022-09-01 14:28:04 *   alice AA-A11C-AC8E-01 Alice     alice@acme.com
    # Shown 1 results
```

Useful environment variables

To facilitate using `lc`, you may consider defining the following environment variables:

```
export LUMINA_HOST=lumina.acme.com
export LUMINA_USER=alice
export LUMINA_PASS=secr3t
```

After that, you can connect to the server effortlessly. For example, this command will print information about the server and the client:

```
>./lc info
    Hex-Rays Lumina Server v8.0
    Lumina time: 2022-09-01 14:28:02, up since 2022-09-01 14:27:58
    MAC address: <snipped macaddr>
    Client name: alice *ADMIN*
    Client host: 127.0.0.1
...
```

2.4. Lumina server command-line options

-p … (--port-number …)	Port number (default 65432)
-i … (--ip-address …)	IP address to bind to (default to any)
-c … (--certchain-file …)	TLS certificate chain file
-k … (--privkey-file …)	TLS private key file
-v (--verbose)	Verbose mode
(--recreate-schema …)	Drop & re-create schema. Note that **THIS WILL ERASE ALL DATA**
(--upgrade-schema)	Upgrade database schema; then quit. Only necessary when upgrading to a newer version of the Lumina server.
-C … (--connection-string …)	Connection string
-l … (--log-file …)	Log file
-f … (--config-file …)	Config file
-D … (--badreq-dir …)	Directory holding dumps of requests causing internal errors

2.5. Troubleshooting

2.5.1. The server complains about a "world-accessible" file, and exits

The following files shouldn't be readable by everyone on the system, but only by `root` and `lumina`:

- `lumina.conf`: this file file holds the connection string to the database the server will use, and might contain credentials.
- `lumina.crt`: the certificate chain
- `lumina.key`: the private key file
- `lumina.lic`: the license file

As a precaution, the Lumina server will refuse to start if these files are readable by unauthorized users.

Please make sure they:

- have `lumina:lumina` ownership: `chown lumina:lumina lumina.crt lumina.key lumina.lic lumina.conf`
- are not world-accessible: `chmod 640 lumina.crt lumina.key lumina.lic lumina.conf`

2.5.2. MySQL

Before the first `--recreate-schema` command can succeed, it is necessary to create the MySQL database, as well as the user that it will be accessed as.

2.5.3. "Authentication plugin 'caching_sha2_password' cannot be loaded"

Some recent Linux distributions ship versions of MySQL that use the "caching_sha2_password" password-checking plugin (as opposed to the traditional "native password" one.)

In order for the Lumina server to able to use this strategy, it would have to ship with a `libmysqlclient.so` file that is linked against `libssl.so.3`.

Unfortunately, this library is not yet available on all of the various [versions of] distributions that we currently support, and introducing this dependency would significantly reduce the diversity of platforms on whith the Lumina server can run.

Consequently (for now) we have opted for another approach: the "lumina" MySQL user should use MySQL's "native password" strategy. This can be accomplished by issuing the following command in a MySQL prompt:

```
ALTER USER lumina@localhost IDENTIFIED WITH mysql_native_password BY '<luminauserpassword>';
```

Once `libssl.so.3` is more generally available, Lumina will not require this particular fine-tuning anymore.

2.5.4. MySQL TLS connections

For the same reasons as the `caching_sha2_password` issue, the Lumina server does not use TLS connections and thus will negociate a plain TCP connection to the MySQL server.

Creating the user and database

What follows is an example creating a user + database, on a Debian system:

```
>sudo mysql -uroot -p
[sudo] password for aundro:
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 14306
Server version: 10.1.48-MySQL-0+deb9u2 Debian 9.13

Copyright (c) 2000, 2018, Oracle, MySQL Corporation Ab and others.
```

```
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]> create user lumina@localhost;
Query OK, 0 rows affected (0.00 sec)

MySQL [(none)]> set password for lumina@localhost = PASSWORD('<snipped>');
Query OK, 0 rows affected (0.00 sec)

MySQL [(none)]> grant all on *.* to lumina@localhost;
Query OK, 0 rows affected (0.00 sec)

MySQL [(none)]> ALTER USER lumina@localhost IDENTIFIED WITH mysql_native_password BY '<snipped>';
Query OK, 0 rows affected (0.00 sec)

MySQL [(none)]> create database test_lumina;
Query OK, 1 row affected (0.00 sec)

MySQL [(none)]> [Ctrl+C] Bye
```

"Index column size too large. The maximum column size is 767 bytes."

The Lumina server cannot create its schema due to a particularly stringent limit on "index prefix sizes" in older versions of MySQL.

This limit was increased in MySQL **5.8**, and thus this is the minimum version the Lumina server can work with.

"Error: Cannot connect to lumina db"

In this case, edit the configuration file, by default `/opt/lumina/lumina.conf` and replace `Server=localhost` by `Server=127.0.0.1` in `CONNSTR`.

3. Concepts

3.1. What is the Lumina server

The Lumina server is a "functions metadata" repository.

It is a place where IDA users can **push**, and **pull** such metadata, to ease their reverse-engineering work: metadata can be extracted from existing projects, and re-applied effortlessly to new projects, thereby reducing (sometimes dramatically) the amount of time needed to analyze binaries.

3.1.1. Functions metadata

The Lumina server associates "function metadata" to functions, by means of a (`md5`) *hash* of those functions: whenever it wants to push information to, or pull information from the server, IDA will first have to compute hashes of the functions it wants to retrieve metadata for, and send those hashes to the Lumina server.

Similarly, when IDA **push**es information to the Lumina server, it will first compute hashes for the corresponding functions, extract the metadata corresponding to those from the `.idb` file, and send those hash+metadata pairs to the server.

3.1.2. Metadata contents

Metadata about functions can include:

- function name
- function address
- function size
- function prototype
- function [repeatable] comments
- instruction-specific [repeatable] comments
- anterior/posterior (i.e., "extra") comments
- user-defined "stack points" in the function's frame
- the function frame description and stack variables
- instructions operands representations

3.1.3. Pushing & overriding metadata

When a user pushes metadata about a function whose `md5` hash isn't present in the database, the Lumina server will simply create a new record for it.

However, when a user pushes metadata about a function whose `md5` hash (and associated metadata) is already present in the database, the Lumina server will attempt to "score" the quality of the old metadata and the quality of the new metadata. If the score of the new metadata is higher, the new function metadata will override the previous one.

NOTE When a user asks IDA to push *all* functions to the Lumina server, IDA will automatically skip some functions: those that still have a "dummy" name (e.g., `sub_XXXX`), or that are below a certain size threshold (i.e., 32 bytes) will be ignored.

3.1.4. Metadata history

The Lumina server retains a history of the metadata associated to functions. Using the `lc` utility, it is possible to dig into that history, and view changes (detailed diffs, too.)

3.1.5. File contents

It's worth pointing out that when pushing metadata to the Lumina server, IDA will not push the binary file itself. Only the following metadata about the file itself will be sent:

- the name of the input file

- the name of the IDB file
- a `md5` hash of the input file

The Lumina server cannot therefore be used as a backup/repository for binary files & IDBs