VisionLabs

MACHINES CAN SEE

LUNA Index Module

Installation manual

v.5.84.0

Contents

Default ports for services

Service name	Port
LUNA PLATFORM Python Matcher Proxy	5110
LUNA Index Module Indexer	5180
LUNA Index Module Index Manager	5190
LUNA Index Module Indexed Matcher	5200

Configuration names for services

The table below includes the LIM service names in the Configurator service. Use these parameters to configure your services.

Service	Service name in Configurator
Python Matcher Proxy	luna-matcher-proxy
Index Manager	lim-manager
Indexer	lim-indexer
Indexed Matcher	lim-matcher

Introduction

This document describes the general approach for deploying LUNA Index Module (LIM) in Docker containers.

LIM is an substantive module, so it should be integrated into LUNA PLATFORM 5 of a similar version already deployed. It is necessary to update the LP if the versions differ.

Each LIM service has its own image. Docker images are the basis of containers. Each container includes libraries required for running services and execution parameters for use within a container runtime.

It is considered that installation is performed on the server with Almalinux 8 OS, where LIM was not installed.

Firewall and SELinux should be manually configured on the server by the administrator. Their configuration is not described in this document.

No data backup or databases replication is implemented for LP data in this installation.

For a successful launch, you need to perform the actions from the sections "Before launch" and "Services launch". The section "Additional information" provides useful information on the description of Docker commands, further steps for performing descriptor matching, etc.

This document does not include a tutorial for Docker usage. Please refer to the Docker documentation to find more information about Docker:

https://docs.docker.com

> This document includes an example of LIM deployment. It implements LIM minimum power operating for demonstration purposes and cannot be used for the production system.

> It is recommended to use orchestration services for the commercial usage of LIM. Their utilization is not described in this manual.

> All the provided commands should be executed using the Bash shell (when you launch commands directly on the server) or Putty (when you remotely connect to the server). The provided commands were tested with these utilities only. The use of other shells or emulators can lead to errors when executing commands.

1 Before launch

Make sure that you are the **root** user before launch!

Before launching the LUNA Index Module, you must perform the following actions:

- Unpack the LIM distribution.
- Create symbolic link.
- Create directory to store indexes.
- Configure SELinux and Firewall.
- Login to VisionLabs registry.
- Choose logging method.
- Upload LIM settings to Configurator.
- Update settings in the Configurator.

1.1 Distribution unpacking

The distribution package is an archive **lim_v.5.84.0**, where **v.5.84.0** is a numerical identifier, describing the current LUNA Index Module version.

The archive includes configuration files, required for installation and exploitation. It does not include Docker images for the services. They should be downloaded from the Internet.

Move the distribution package to the directory on your server before the installation. For example, move the files to /root/ directory. The directory should not contain any other distribution or license files except the target ones.

Move the distribution to the /var/lib/luna/ directory.

```
mv /root/lim_v.5.84.0.zip /var/lib/luna
```

Install the unzip archiver if it is necessary.

```
yum install -y unzip
```

Go to the folder with distribution.

```
cd /var/lib/luna
```

Unzip files.

```
unzip lim_v.5.84.0.zip
```

1.2 Symbolic link creation

Create a symbolic link. The link indicates that the current version of the distribution file is used to run LIM.

```
ln -s lim_v.5.84.0 lim-current
```

1.3 Create directory to store indexes

To store indexes and interact with them on the server, you need to create an appropriate directory. This directory will be mounted when the LIM services are launched.

Create folder to store indexes.

```
mkdir -p /var/lib/luna/lim_storage
```

Set appropriate permissions to read/write this folder.

```
chown -R 1001:0 /var/lib/luna/lim_storage
```

1.4 SELinux and Firewall

You must configure SELinux and Firewall so that they do not block LUNA PLATFORM services.

> SELinux and Firewall configurations are not described in this guide.

If SELinux and Firewall are not configured, the installation cannot be performed.

1.5 Login to registry

When launching containers, you should specify a link to the image required for the container launching. This image will be downloaded from the VisionLabs registry. Before that, you should login to the registry.

> Login and password can be requested from the VisionLabs representative.

Enter login <username>.

```
docker login dockerhub.visionlabs.ru --username <username>
```

After running the command, you will be prompted for a password. Enter password.

> In the `docker login` command, you can enter the login and password at the same time, but this does not guarantee security because the password can be seen in the command history.

1.6 Choose logging method

There are two methods to output logs in LUNA PLATFORM:

- Standard log output (stdout).
- Log output to a file.

Log output settings are set in the settings of each service in the `<SERVICE_NAME>_LOGGER` section.

If necessary, you can use both methods of displaying logs.

> For more information about the LUNA PLATFORM logging system, see the "Logging" section in the administrator manual.

1.6.1 Logging to stdout

This method is used by default and requires no further action.

1.6.2 Logging to file

Note: When you enable saving logs to a file, you should remember that logs occupy a certain place in the storage, and the process of logging to a file negatively affects system performance.

To use this method, you need to perform the following additional actions:

- **Before launching the services:** Create directories for logs on the server.
- **After launching the services:** Activate log recording and set the location of log storage inside LP service containers.
- **During the launch of services:** Configure synchronization of log directories in the container with logs on the server using the `volume` argument at the start of each container.

Examples of container launch commands in this documentation contain arguments for synchronizing log directories.

Note that the above steps must be performed before, during and after starting the services. Saving logs to file will not work if you perform all actions after starting the containers.

See the instructions for enabling logging to files in the "Logging the server" section.

1.7 Upload LIM settings to Configurator

To use LIM services with LP 5, you need to upload their settings to the Configurator service using the configuration migration mechanism.

```
docker run \
-v /etc/localtime:/etc/localtime:ro \
--rm \
--entrypoint='' \
--network=host \
dockerhub.visionlabs.ru/luna/lim-configs:v.0.6.0 python3 -m configs.migrate
    head --config_db_url postgres://luna:luna@127.0.0.1:5432/
    luna_configurator
```

Where `--config_db_url postgres://luna:luna@127.0.0.1:5432/luna_configurator` is the flag for specifying the `luna_configurator` database address.

1.8 Update settings in the Configurator

Next, you need to enable the use of the Python Matcher Proxy service and add the matching plugin to the list of plugins used by the Python Matcher Proxy service.

Copy the file with the necessary settings to the Configurator container.

```
docker cp /var/lib/luna/lim-current/example-docker/configs/lim_settings.json
    luna-configurator:/srv/lim_settings.json
```

Update the settings in the Configurator service.

```
docker exec -it luna-configurator python3 ./base_scripts/db_create.py --dump
    -file /srv/lim_settings.json
```

As a result, the following settings will be updated in the Configurator service:

```
LUNA_MATCHER_PROXY_ACTIVE_PLUGINS = ["indexed_matcher"]
ADDITIONAL_SERVICES_USAGE = "luna_matcher_proxy":true
```

2 Services launch

This section gives examples for launching LIM containers.

The sequence of launching LIM services is as follows:

- Python Matcher Proxy
- Indexer
- Index Manager
- Indexed Matcher

To launch LIM services, the LP 5 must be deployed.

It is recommended to launch containers one by one and wait for the container status to become "up" (use the `docker ps` command).

When launching each service, certain parameters are used, for example, `--detach`, `--network`, etc. See the section "Launching parameters description" for more detailed information about all launch parameters of LUNA PLATFORM services and databases.

See the "Docker commands" section for details about working with containers.

2.1 Python Matcher proxy container launch

Use the following command to launch the service:

```
docker run \
--env=CONFIGURATOR_HOST=127.0.0.1 \
--env=CONFIGURATOR_PORT=5070 \
--env=PORT=5110 \
--env=WORKER_COUNT=1 \
--env=RELOAD_CONFIG=1 \
--env=RELOAD_CONFIG_INTERVAL=10 \
--env=SERVICE_TYPE="proxy" \
-v /etc/localtime:/etc/localtime:ro \
-v /tmp/logs/python-matcher-proxy:/srv/logs \
--name=luna-python-matcher-proxy \
--restart=always \
--detach=true \
--network=host \
dockerhub.visionlabs.ru/luna/luna-python-matcher:v.1.11.1
```

2.2 Indexer container launch

Use the following command to launch the service:

```
docker run \
--env=CONFIGURATOR_HOST=127.0.0.1 \
--env=CONFIGURATOR_PORT=5070 \
--env=PORT=5180 \
--env=WORKER_COUNT=1 \
--env=RELOAD_CONFIG=1 \
--env=RELOAD_CONFIG_INTERVAL=10 \
-v /etc/localtime:/etc/localtime:ro \
-v /tmp/logs/lim-indexer:/srv/logs \
-v /var/lib/luna/lim_storage:/srv/local_storage \
--name=lim-indexer \
--restart=always \
--detach=true \
--network=host \
dockerhub.visionlabs.ru/luna/lim-indexer:v.0.6.0
```

> The deployment of the Indexer service should be done on a separate server, because building an
> index takes a lot of resources for a long time. One Indexer instance can only build one index at a
> time, so it is recommended to run multiple indexer instances. The indexer must be also configured
> with storage, which must be large enough.

2.3 Index Manager container launch

Use the following command to launch the service:

```
docker run \
--env=CONFIGURATOR_HOST=127.0.0.1 \
--env=CONFIGURATOR_PORT=5070 \
--env=PORT=5190 \
--env=WORKER_COUNT=1 \
--env=RELOAD_CONFIG=1 \
--env=RELOAD_CONFIG_INTERVAL=10 \
-v /etc/localtime:/etc/localtime:ro \
-v /tmp/logs/lim-manager:/srv/logs \
-v /var/lib/luna/lim_storage:/srv/local_storage \
--name=lim-manager \
--restart=always \
--detach=true \
--network=host \
dockerhub.visionlabs.ru/luna/lim-manager:v.0.6.0
```

> It is recommended to run at least two manager instances for redundancy purposes. Since task management is carried out through the Redis, if one manager is down, the second one will be able to continue its work from the instant step.

2.4 Indexed Matcher container launch

Use the following command to launch the service:

```
docker run \
--env=CONFIGURATOR_HOST=127.0.0.1 \
--env=CONFIGURATOR_PORT=5070 \
--env=PORT=5200 \
--env=WORKER_COUNT=1 \
--env=RELOAD_CONFIG=1 \
--env=RELOAD_CONFIG_INTERVAL=10 \
-v /etc/localtime:/etc/localtime:ro \
-v /tmp/logs/lim-matcher:/srv/logs \
-v /var/lib/luna/lim_storage:/srv/local_storage \
--name=lim-matcher \
--restart=always \
--detach=true \
--network=host \
dockerhub.visionlabs.ru/luna/lim-matcher:v.0.6.0
```

Indexed Matcher does not communicate with other LIM services. It only monitors the storage, and when indices appear it loads them into memory. Since matching requests processing is carried out through the Redis streams, any number of matcher instances could be run without any system config updates. The number of Indexed Matcher instances should be determined by performance requirements.

For each instance of the Indexed Matcher service, you can set an environment variable `VL_LIM_MATCHER_HOST` indicating its IP address. This can be useful for separating monitoring data into different instances. For example, you can determine how many instances are in work, which instance caused the error, which indexes are in operation, etc.

3 Additional information

This section provides the following additional information:

- Steps to perform descriptors matching.
- Useful commands for working with Docker.
- Description of the parameters for launching LIM services and creating databases.
- Actions to enable saving LIM service logs to files.

3.1 Steps to perform descriptors matching

To start the descriptors matching procedure using LIM services, you should perform the following steps:

- select the preferred mode that will be used for the descriptors matching — **one-time** or **automatic**.
- **One-time mode**: Specify the required "list_id" in the request body to the "create task" resource of the Index Manager service and perform the request.
- **Automatic mode for working with specific lists**: Specify the required "list_id" in the "indexing_lists" setting of the "LIM_MANAGER_INDEXING" of the Configurator service.
- **Automatic mode for working with all existing lists**:
  - Specify the "dynamic" value in the "indexing_lists" setting of the "LIM_MANAGER_INDEXING" of the Configurator service.
  - Specify the minimum number of faces in the list in the "min_indexing_list_size" setting of the "LIM_MANAGER_INDEXING" of the Configurator service to index all existing descriptors in the lists.
- Send a request for matching descriptors from the API service.

> For details on LIM service interaction, see the "Service interaction" section of the LIM administrator manual.

3.2 Docker commands

3.2.1 Show containers

To show the list of launched Docker containers use the command:

```
docker ps
```

To show all the existing Docker containers use the command:

```
docker ps -a
```

3.2.2 Copy files to container

You can transfer files into the container. Use the `docker cp` command to copy a file into the container.

```
docker cp <file_location> <container_name>:<folder_inside_container>
```

3.2.3 Enter container

You can enter individual containers using the following command:

```
docker exec -it <container_name> bash
```

To exit the container, use the command:

```
exit
```

3.2.4 Images names

You can see all the names of the images using the command:

```
docker images
```

3.2.5 Delete image

If you need to delete an image:

- Run the `docker images` command.

- Find the required image, for example dockerhub.visionlabs.ru/luna/luna-image-store.
- Copy the corresponding image ID from the IMAGE ID, for example, "61860d036d8c".
- Specify it in the deletion command:

```
docker rmi -f 61860d036d8c
```

Delete all the existing images.

```
docker rmi -f $(docker images -q)
```

3.2.6 Stop container

You can stop the container using the command:

```
docker stop <container_name>
```

Stop all the containers:

```
docker stop $(docker ps -a -q)
```

3.2.7 Delete container

If you need to delete a container:

- Run the "docker ps" command.
- Stop the container (see Stop container).
- Find the required image, for example dockerhub.visionlabs.ru/luna/luna-image-store.
- Copy the corresponding container ID from the CONTAINER ID column, for example, "23f555be8f3a".
- Specify it in the deletion command:

```
docker container rm -f 23f555be8f3a
```

Delete all the containers.

```
docker container rm -f $(docker container ls -aq)
```

3.2.7.1 Check service logs

You can use the following command to show logs for the service:

```
docker logs <container_name>
```

3.3 Launching parameters description

When launching a Docker container for a LUNA Index Module service you should specify additional parameters required for the service launching.

> The parameters specific for a particular container are described in the section about this container launching.

All the parameters given in the service launching example are required for proper service launching and utilization.

3.3.1 Launching services parameters

Example command of launching LP services containers:

```
docker run \
--env=CONFIGURATOR_HOST=127.0.0.1 \
--env=CONFIGURATOR_PORT=5070 \
--env=PORT=<Port_of_the_launched_service> \
--env=WORKER_COUNT=1 \
--env=RELOAD_CONFIG=1 \
--env=RELOAD_CONFIG_INTERVAL=10 \
-v /etc/localtime:/etc/localtime:ro \
-v /tmp/logs/<service>:/srv/logs/ \
--name=<service_container_name> \
--restart=always \
--detach=true \
--network=host \
dockerhub.visionlabs.ru/luna/<service-name>:<version>
```

The following parameters are used when launching LP services containers:

- docker run — Command for running the selected image as a new container.

- dockerhub.visionlabs.ru/luna/<service-name>:<version> — Sets the image required for the container launching.

> Links to download the container images you need are available in the description of the corresponding container launching.

- --network=host — Sets that a network is not simulated and the server network is used. If you need to change the port for third-party party containers, you should change this string to -p 5440:5432. Where the first port 5440 is the local port and 5432 is the port used inside the container. The example is given for PostgreSQL.

- `--env=` — Sets the environment variables required to run the container (see the "Service arguments" section).

- `--name=<service_container_name>` — Sets the name of the launched container. The name must be unique. If there is a container with the same name, an error will occur.

- `--restart=always` — Sets a restart policy. The daemon will always restart the container regardless of the exit status.

- `--detach=true` — Run the container in the background mode.

- `-v` — Enables you to mount the content of a server folder into a volume in the container. Thus their contents will synchronize. The following general data is mounted:

- `/etc/localtime:/etc/localtime:ro` — Sets the current time zone used by the system in the container.

- `/tmp/logs/<service>:/srv/logs/` — Enables copying of the folder with service logs to your server `/tmp/logs/<service>` directory. You can change the directory where the logs will be saved according to your needs.

3.3.1.1 Service arguments

Each service in LUNA PLATFORM has its own launch arguments. These arguments can be passed through:

- Setting a flag for the launch script (`run.py`) of the corresponding service.
- Setting environment variables (`--env`) on the Docker command line.

For example, using the `--help` flag you can get a list of all available arguments. An example of passing an argument to an API service:

```
docker run --rm dockerhub.visionlabs.ru/luna/luna-api:v.6.39.0 python3 /srv/
    luna_api/run.py --help
```

List of main arguments:

Launch flag	Environment variable	Description
`--port`	`PORT`	Port on which the service will listen for connections.
`--workers`	`WORKER_COUNT`	Number of workers for the service.
`--log_suffix` `--log_suffix`	`LOG_SUFFIX LOG_SUFFIX`	Suffix added to log file names (with the option to write logs to a file enabled).

`--config-reload`	`RELOAD_CONFIG`	Enable automatic configuration reload. See "Automatic configurations reload" in the LUNA PLATFORM 5 administrator manual.
`--pulling-time`	`RELOAD_CONFIG_INTERVAL`	Configuration checking period (default 10 seconds). See "Automatic configurations reload" in the LUNA PLATFORM 5 administrator manual.
`--luna-config` `--luna-config`	`CONFIGURATOR_HOST,` `CONFIGURATOR_PORT`	Address of the Configurator service for downloading settings. For `--luna-config` it is sent in the format `http://localhost:5070/1`. For environment variables, the host and port are set explicitly. If the argument is not given, the default configuration file will be used.
`--config`	None	Path to the file with service configurations.
`--<config_name>`	`--EXTEND_CMD=<` `config_name>`	Tag of the specified configuration in the Configurator. When setting this configuration, the value of the tagged configuration will be used. Example: `--LUNA_MONITORING TAG_1` **Note:** You must pre-tag the appropriate configuration in. Configurator. **Note:** Only works with the `--luna-config` flag.
`--tls_cert`	None	Path to the SSL certificate for launching the service using the HTTPS protocol.
`--tls_key`	None	Path to the SSL private key for launching the service using the HTTPS protocol.
`--tls_key_pass`	None	Password for the SSL private key for launching the service using the HTTPS protocol.

The list of arguments may vary depending on the service.

It is also possible to override the settings of services at their start using environment variables.

The `VL_SETTINGS` prefix is used to redefine the settings. Examples:

- `--env=VL_SETTINGS.LUNA_MONITORING.SEND_DATA_FOR_MONITORING=0`.   Using the environment variable from this example will set the "SEND_DATA_FOR_MONITORING" setting for the `LUNA_MONITORING` section to "0".
- `--env=VL_SETTINGS.OTHER.STORAGE_TIME=LOCAL`.   For non-compound settings (settings that are located in the "OTHER" section in the configuration file), you must specify the "OTHER" prefix. Using the environment variable from this example will set the value of the "STORAGE_TIME" setting (if the service uses this setting) to "LOCAL".

Passing flags using environment variable

Flags for which an environment variable is not explicitly allocated can be passed using the environment variable `EXTEND_CMD`.

For example, you can pass the configurations tag in the following way:

```
--env=EXTEND_CMD="--LUNA_MONITORING=TAG_1 --LUNA_EVENTS_DB=TAG_2"
```

- `/var/lib/luna/lim_storage:/srv/local_storage` — enables you to mount a directory for storing indexes in local storage. The location and name of the directory for storing indexes inside LIM containers is set in the "INDEX_STORAGE_LOCAL" settings of the LIM services. Note that the directory must be the same for all three services. The local directory can be changed according to your needs.

3.3.2 Creating DB parameters

Example command of launching containers for database migration or database creation:

```
docker run \
-v /etc/localtime:/etc/localtime:ro \
-v /tmp/logs/<service>:/srv/logs/ \
--rm \
--network=host \
dockerhub.visionlabs.ru/luna/<service-name>:<version> \
python3 ./base_scripts/db_create.py --luna-config http://localhost:5070/1
```

The following parameters are used when launching containers for database migration or database creation:

Here:

- `--rm` — Sets if the container is deleted after all the specified scripts finish processing.

- `python3 ./base_scripts/db_create.py` — Sets Python version and a script `db_create.py` launched in the container. The script is used for the database structure creation.

- `--luna-config http://localhost:5070/1` — Sets where the launched script should receive configurations. By default, the service requests configurations from the Configurator service.

3.4 Logging to server

To enable saving logs to the server, you should:

- Create directories for logs on the server.
- Activate log recording and set the location of log storage inside LP service containers.
- Configure synchronization of log directories in the container with logs on the server using the `volume` argument at the start of each container.

3.4.1 Create logs directory

Below are examples of commands for creating directories for saving logs and assigning rights to them for all LUNA PLATFORM services.

```
mkdir -p /tmp/logs/lim-manager /tmp/logs/lim-indexer /tmp/logs/lim-matcher /
    tmp/logs/python-matcher-proxy
```

```
chown -R 1001:0 /tmp/logs/lim-manager /tmp/logs/lim-indexer /tmp/logs/lim-
    matcher /tmp/logs/python-matcher-proxy
```

3.4.2 Logging activation

To enable logging to file, you need to set the `log_to_file` and `folder_with_logs` settings in the `<SERVICE_NAME>_LOGGER` section of the settings for each service.

Automatic method

To update logging settings, you can use the `logging.json` settings file provided with the distribution package.

Run the following command:

```
docker cp /var/lib/luna/lim-current/example-docker/configs/logging.json luna
    -configurator:/srv/luna_configurator/used_dumps/logging.json
```

Update your logging settings with the copied file.

```
docker exec -it luna-configurator python3 ./base_scripts/db_create.py --dump
    -file /srv/luna_configurator/used_dumps/logging.json
```

Manual method

Go to the Configurator service interface (`127.0.0.1:5070`) and set the logs path in the container in the `folder_with_logs` parameter for all services whose logs need to be saved. For example, you can use the path `/srv/logs`.

Set the `log_to_file` option to `true` to enable logging to a file.

3.4.3 Mounting directories with logs when starting services

The log directory is mounted with the following argument when starting the container:

```
-v <server_logs_folder>:<container_logs_folder> \
```

where `<server_logs_folder>` is the directory created in the create logs directory step, and `<container_logs_folder>` is the directory created in the activate logging step.

Example of command to launch the Index Manager service with mounting a directory with logs:

```
docker run \
--env=CONFIGURATOR_HOST=127.0.0.1 \
--env=CONFIGURATOR_PORT=5070 \
--env=PORT=5190 \
--env=WORKER_COUNT=1 \
--env=RELOAD_CONFIG=1 \
--env=RELOAD_CONFIG_INTERVAL=10 \
-v /etc/localtime:/etc/localtime:ro \
-v /tmp/logs/lim-manager:/srv/logs \
-v /var/lib/luna/lim_storage:/srv/local_storage \
--name=lim-manager \
--restart=always \
--detach=true \
--network=host \
dockerhub.visionlabs.ru/luna/lim-manager:v.0.6.0
```

The example container launch commands in this documentation contain these arguments.