



**materialise**

innovators you can count on

# Materialise Floating License Server

Administrator Guide

## Contents

1 Introduction .....	4
2 Materialise Floating License Server Overview .....	4
2.1. Processing license requests by Proxy Service .....	5
2.2. Licensing API Server .....	5
2.2.1 Health check .....	5
2.2.2 License usage status .....	5
3 Installing Materialise Floating License Server .....	7
3.1. New installation .....	7
3.2. Upgrading earlier version .....	9
3.3. License server components .....	10
4 Managing License Server .....	11
4.1. Command line interface .....	11
4.1.1 Show help .....	11
4.1.2 Register service .....	11
4.1.3 Unregister service .....	11
4.1.4 Start service .....	11
4.1.5 Stop service .....	11
4.1.6 Add key file .....	12
4.1.7 Show system ID .....	12
4.1.8 Show registered license keys .....	12
4.1.9 List license key users .....	13
4.1.10 Delete license key .....	13
4.2. Configuration file .....	13
4.2.1 Overview .....	14
4.2.2 General settings .....	14
4.2.3 License key access permissions .....	14
4.2.4 License key reservation .....	18
4.2.5 Logging settings .....	20
4.2.6 SSL settings .....	21
5 Managing Proxy Service .....	22
5.1. Command line interface .....	22
5.1.1 Register service .....	22
5.1.2 Unregister service .....	22

5.1.3 Start service .....	22
5.1.4 Stop service .....	22
5.2. Configuration UI tool .....	22
5.2.1 Control buttons.....	23
5.2.2 Generic .....	25
5.2.3 Log settings.....	25
5.2.4 Automatic licenses renew settings.....	25
5.2.5 Mail settings .....	26
6 Managing License API Server .....	27
6.1. Command line interface .....	27
6.1.1 Show help .....	27
6.1.2 Register service .....	27
6.1.3 Unregister service .....	27
6.1.4 Start service .....	27
6.1.5 Stop service .....	27
6.2. Configuration file.....	28
6.2.1 Overview.....	28
6.2.2 General settings.....	28
6.2.3 License Server connection settings.....	29
6.2.4 Logging settings.....	29
6.2.5 SSL settings.....	30
7 Generating license reports with LicReport Tool .....	32
7.1 Introduction.....	32
7.2 LicReport usage .....	32
7.2.1 License module usage report .....	33
7.2.2 Log entries report.....	36
7.2.3 License list report.....	38
7.3 Command line parameters list .....	40
Appendix A: Cipher List Format.....	42

## 1 Introduction

This document describes how to use the Materialise Floating License Server for license administrators.

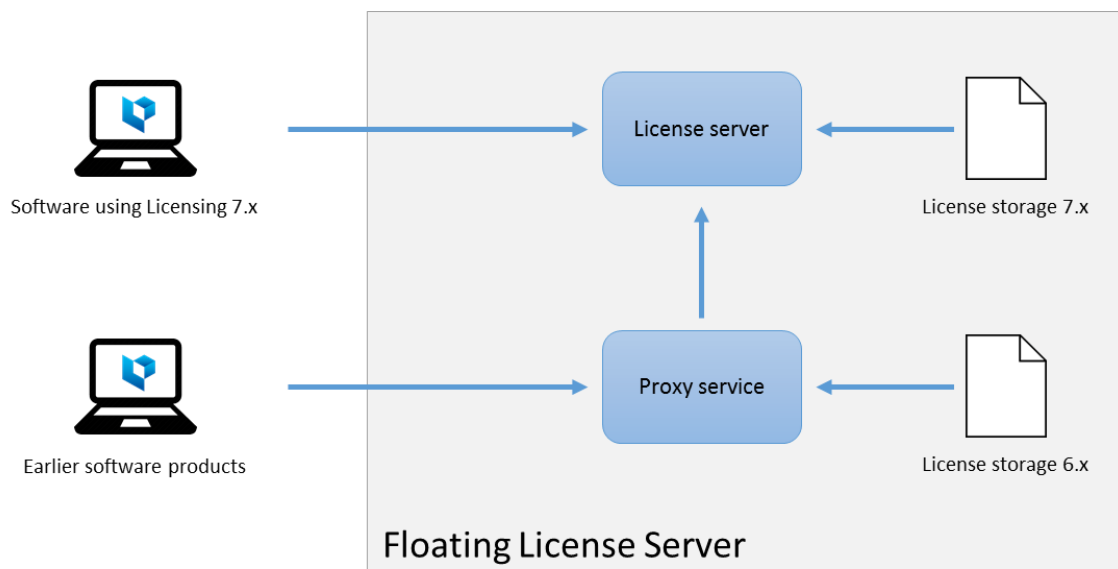
## 2 Materialise Floating License Server Overview

The Materialise Floating License Server provides the licenses to Materialise software products via a network. The server receives requests from the software products, verifies the validity of requested license keys and returns the response data.

The floating licenses installed on the server contain the following information:

- Numeric module ID, which identifies the software product's functionality that can be used if this license valid;
- Maximal allowed version number;
- Validity period of the license module;
- Product key (CCKey or voucher key) the license module is generated for; □Maximum number of concurrent connections.

The main components of the Materialise Floating License Server are depicted on the following diagram:



The following components are present on the Materialise Floating License Server:

- **License server** – a service, which processes the license requests from software products, which use the Licensing 7.x component (Magics 23 and later software).
- **License storage 7.x** – a storage, which keeps the license keys used by the License server component.
- **Proxy service** – a service, which processes the license requests from earlier software products, which do not use the Licensing 7.x components; this proxy service remains for the backwards compatibility purposes.

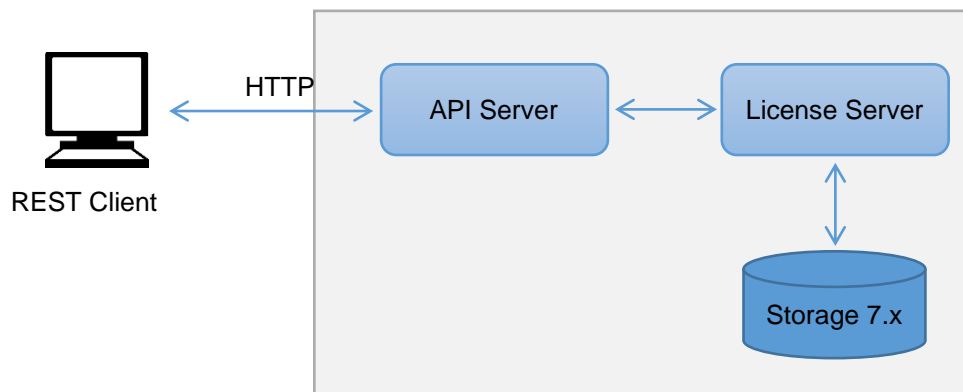
- **License storage 6.x** - a storage, which keeps the license keys used by the Proxy server component.

## 2.1. Processing license requests by Proxy Service

The following logic is used by the Proxy Service when processing the license requests from the software products:

1. Forward request to the License Server component.
2. If the License Server can locate the requested license, forward the response to the software product.
3. If the License Server cannot locate the requested license at step 1, try to locate it in the License Storage 6.x.
4. Send the response to the software product.

## 2.2. Licensing API Server



License API Server is a service that provides REST-interface for querying license server state (e.g. licenses usage). It acts as a proxy for a License Server exposing its status via HTTP protocol.

### 2.2.1 Health check

License API Server provides **/health** endpoint for checking API service liveness.

This endpoint accepts GET requests and replies with 200 status if API Server is working properly, 500 otherwise.

### 2.2.2 License usage status

License API Server provides **/status** endpoint for querying License Server licenses usage.

This endpoint accepts GET requests and replies with a licenses usage status data structure in a JSON format, which looks like this:

```
{
  "vendors": [
    {
      "name": "string",

```

```
"features": [  
  {  
    "name": "string",  
    "totalLicenses": int(64),  
    "usedLicenses": int(64),  
    "type": "string",  
    "sessions": [  
      {  
        "user": "string",  
        "host": "string",  
        "handle": "string",  
        "version": "string",  
        "licenses": int(64)  
      }  
    ]  
  }  
]
```

In default configuration License API Server returns all available (non-expired) licenses on the server (features) for a single vendor called 'Materialise Floating License Server'.

Each feature object contains information about maximum amount of users for a feature, actual current users, license type and lists of current user sessions.

Each session object contains information about user and host name, used feature version, session handle and amount of used licenses.

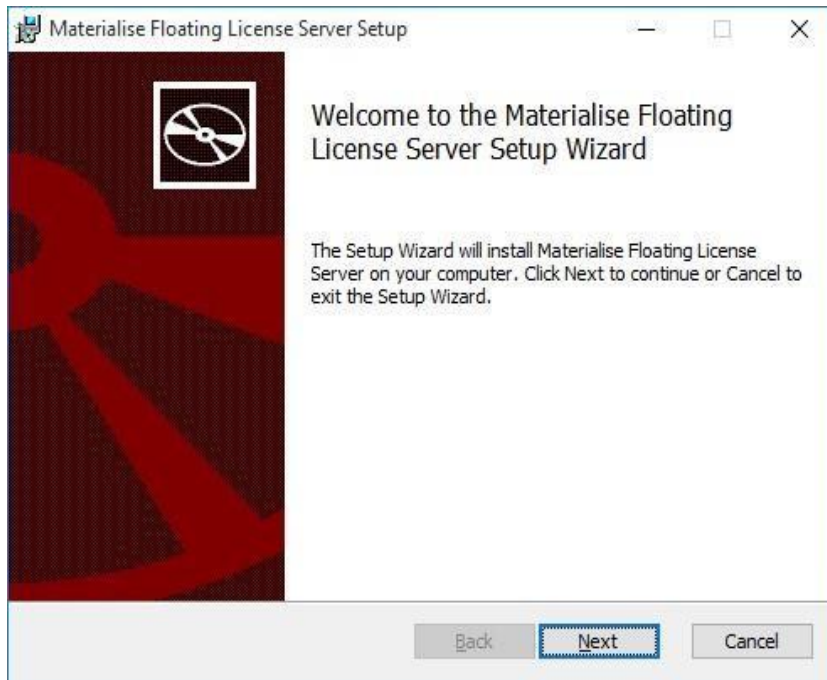
Server gives a 200 status code on successful query and 500 on error.

## 3 Installing Materialise Floating License Server

### 3.1. New installation

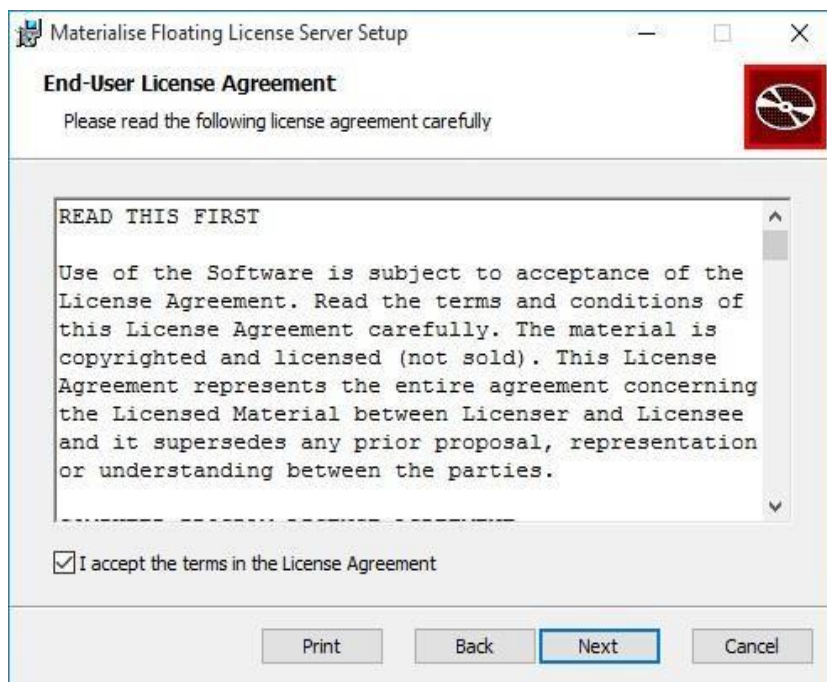
Installation of the Floating License Server is done with the help of the FloatingLicenseServer.msi installer package.

1. Start the installation

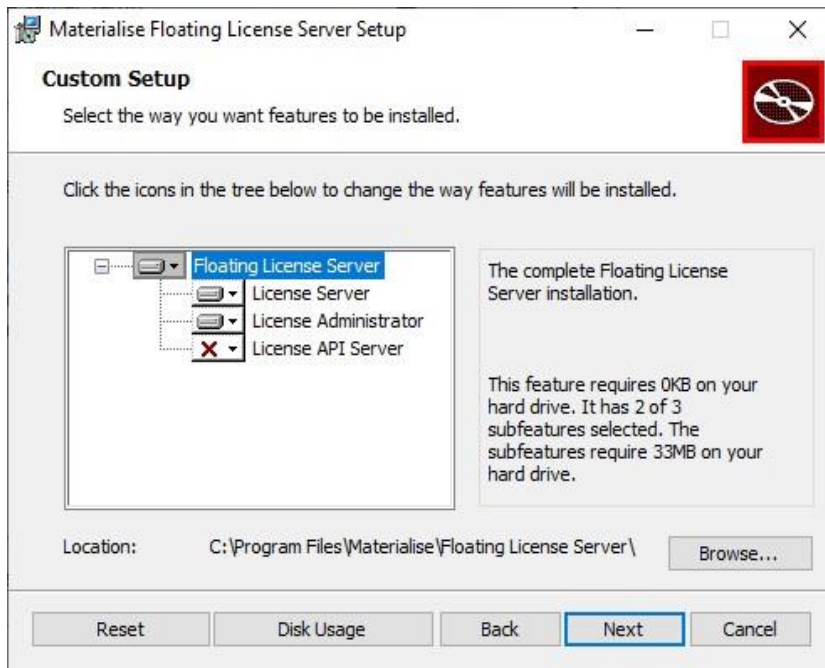


Press Next to proceed to the next step.

2. Accept the license agreement in order to proceed further



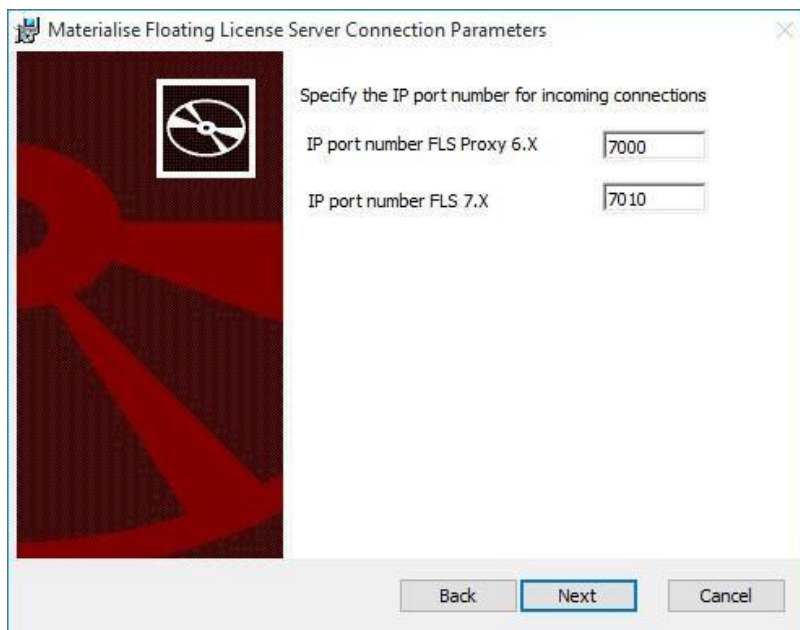
3. Select the components to install



The installation package proposes installing the following components:

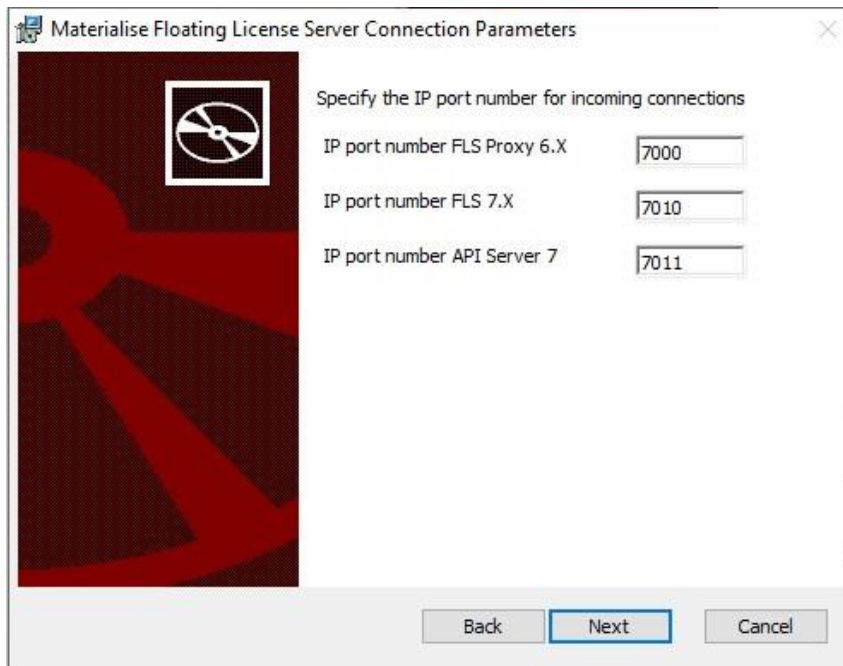
- **License Server** – in this case both the License Server and the Proxy Service will be installed;
- **License Administrator** – a UI tool, which allows viewing the license keys installed on this and other servers, as well as other operations, like registering new license keys;
- **License API Server** – a server providing REST-interface to query License Server status, licenses usage, etc.

4. Specify the IP port numbers, used by the FLS components.

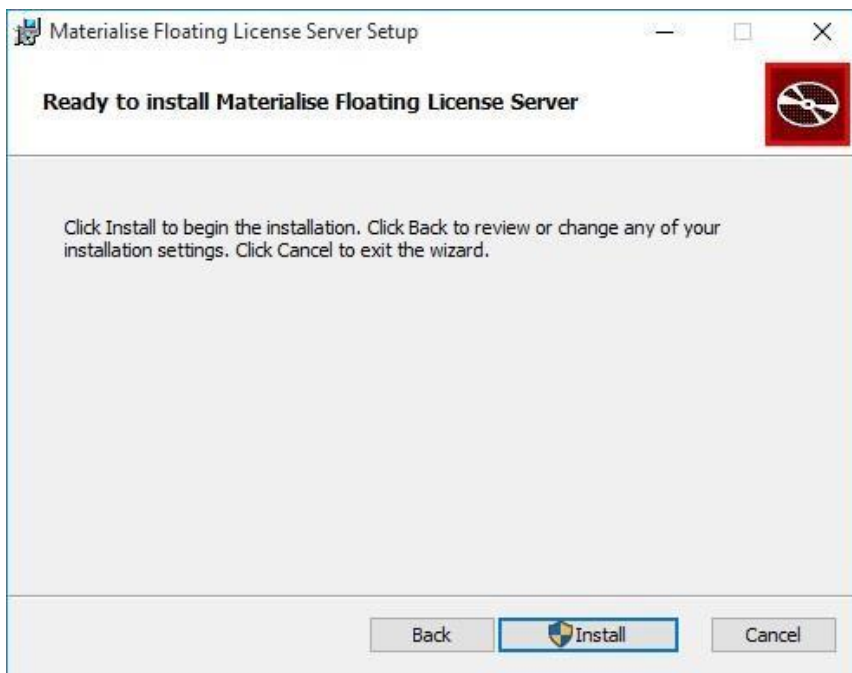


5. When enable License API Server you also need to specify its port.





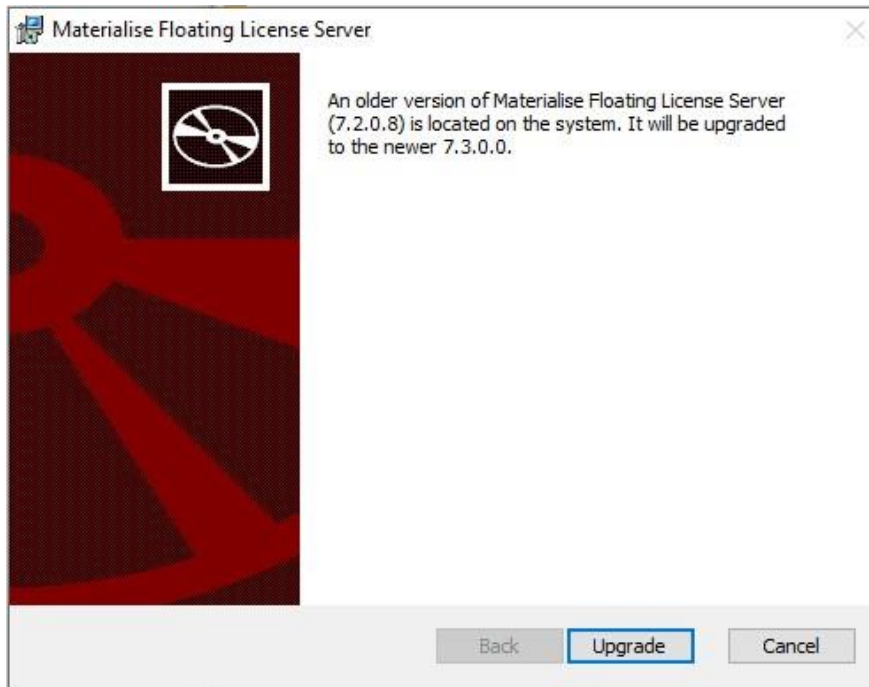
6. Confirm selection and proceed to the installation



### 3.2. Upgrading earlier version

Upgrading the installation of the Materialise Floating License Server can also be done using the `FloatingLicenseServer.msi` package.

When starting the package on the host, where an earlier version of the license server is installed, the installer provides a user with the notification about the upgrade.



After a user presses Upgrade, the installation wizard follows the flow described in the section 3.1. New installation of this document.

### 3.3. License server components

By default the Materialise Floating License Server is installed in the following folder:

```
%PROGRAMFILES%\Materialise\Floating License Server
```

The following components are installed:

- `MatLicenseServer.exe` (7.0 subfolder)– the License Server executable file, it is also registered as a Windows Service named “Materialise Floating License Server 7”;
- `LicSrvFS60.exe` – the Proxy Service executable file, also registered as a Windows Service named “Materialise Floating License Server 6.0”;
- `LicSrvConfig.exe` – configuration UI tool for the Proxy Service component;
- `LicAdmin.exe` (LicAdmin subfolder) – the License Administrator UI tool;
- `MatLicenseAPIServer.exe` (APIServer subfolder) – the License API Server executable file, it is also registered as a Windows Service named “Materialise License API Server 7”. This component is installed if License API Server feature is enabled.



## 4 Managing License Server

### 4.1. Command line interface

A certain set of operations on the License Server can be done using the command line interface provided by `MatLicenseServer.exe`. This section describes the available commands.

#### 4.1.1 Show help

Shows the list of available commands with a brief description for each.

Syntax:

```
MatLicenseServer.exe --help
```

#### 4.1.2 Register service

Registers the `MatLicenseServer.exe` as a “Materialise Floating License Server 7” Windows service.

Syntax:

```
MatLicenseServer.exe --register
```

```
MatLicenseServer.exe -r
```

The command requires administrator rights to execute.

#### 4.1.3 Unregister service

Unregisters the “Materialise Floating License Server 7” Windows service. Syntax:

```
MatLicenseServer.exe --unregister
```

```
MatLicenseServer.exe -u
```

The command requires administrator rights to execute.

#### 4.1.4 Start service

Starts the “Materialise Floating License Server 7” Windows service. Syntax:

```
MatLicenseServer.exe --start
```

The command requires administrator rights to execute.

#### 4.1.5 Stop service

Stops the “Materialise Floating License Server 7” Windows service.

Syntax:

```
MatLicenseServer.exe --stop
```

The command requires administrator rights to execute.

Please note that the License Server component cannot be stopped without first stopping the Proxy Service component.

#### 4.1.6 Add key file

Adds license keys delivered in a key file to the license storage.

Syntax:

```
MatLicenseServer.exe --add <path_to_keyfile>
```

During the execution this command displays the progress of adding the modules from the key file:

```
D:\Temp>"C:\Program Files\Materialise\Floating License Server\7.0\MatLicenseServer.exe" --add rsm.matkey
Thu Jun 27 14:43:50 2019|PID 12852|THR 19540|Licensing|INFO|Connected to localhost:7010 from port 56560
Thu Jun 27 14:43:50 2019|PID 12852|THR 19540|Licensing|INFO|added module 483 version 5.7
```

#### 4.1.7 Show system ID

Shows the system ID value of the host where the license server is installed.

Syntax:

```
MatLicenseServer.exe --sysid Example:
```

```
C:\Program Files\Materialise\Floating License Server\7.0>MatLicenseServer.exe --sysid
Thu Jun 27 14:47:23 2019|PID 3672|THR 14164|Licensing|INFO|Connected to localhost:7010 from port 56592
AC1FD733-53XE-0011-7E34A39795D204CD
```

#### 4.1.8 Show registered license keys

Shows the list of the license keys registered on the license server Syntax:

```
MatLicenseServer.exe --list
```

The following data is shown for each key in the list:

- License module name;
- Maximum allowed version number;
- Validity start date;
- Validity end date;
- Counter (not used);
- Maximum number of users/number of current users;
- CCKey the license key has been generated for;
- License key unique identifier (assigned by the license server after registration)

Example:

```

C:\Program Files\Materialise\Floating License Server\7.0>MatLicenseServer.exe --list
Thu Jun 27 14:52:05 2019|PID 19632|THR 16760|Licensing|INFO|Connected to localhost:7010 from port 56673
Name | Version | Start Date | End Date | Counter | Active/MaxUsers | CCKey | Unique ID
21 | 23.1 | 12-06-2019 02-00-00 | 14-12-2019 15-52-05 | - | 0/20 | C685-82C8-49B4-8D33 | b47149d6-bb94-421d-be5c-df4cf33cfb58
1105 | 8.9 | 12-06-2019 02-00-00 | 14-12-2019 15-52-05 | - | 0/20 | C685-82C8-49B4-8D33 | 213078d4-3937-4aae-aadf-b68b8160f085
520 | 23.1 | 12-06-2019 02-00-00 | 14-12-2019 15-52-05 | - | 0/20 | C685-82C8-49B4-8D33 | 68ce2313-160d-442e-a879-7078951eb7bb
1102 | 8.9 | 12-06-2019 02-00-00 | 14-12-2019 15-52-05 | - | 0/20 | C685-82C8-49B4-8D33 | 4f95d3ef-9473-449e-84d0-47bc436a09f2
1101 | 8.9 | 12-06-2019 02-00-00 | 14-12-2019 15-52-05 | - | 0/20 | C685-82C8-49B4-8D33 | dcef256f-3034-48d2-bcf7-7735ac4c185b
1109 | 8.9 | 12-06-2019 02-00-00 | 14-12-2019 15-52-05 | - | 0/20 | C685-82C8-49B4-8D33 | fa69875c-89a9-4fde-adc3-90c665b4e7cd
1103 | 8.9 | 12-06-2019 02-00-00 | 14-12-2019 15-52-05 | - | 0/20 | C685-82C8-49B4-8D33 | 90f7700f-447c-4b7f-a3b7-7770461c83ba
1111 | 8.9 | 12-06-2019 02-00-00 | 14-12-2019 15-52-05 | - | 0/20 | C685-82C8-49B4-8D33 | e6ef28d4-f58d-47b4-96a9-1494e43f5dfd
1104 | 8.9 | 12-06-2019 02-00-00 | 14-12-2019 15-52-05 | - | 0/20 | C685-82C8-49B4-8D33 | a93a5be3-d0a2-4b5c-ab2a-ae402e0b356f
1117 | 8.9 | 12-06-2019 02-00-00 | 14-12-2019 15-52-05 | - | 0/20 | C685-82C8-49B4-8D33 | f0a25cbb-db00-4338-b196-abd387b3c68a
1120 | 8.9 | 12-06-2019 02-00-00 | 14-12-2019 15-52-05 | - | 0/20 | C685-82C8-49B4-8D33 | b68e5575-998b-48c5-ba9e-f7f041489496
1121 | 8.9 | 12-06-2019 02-00-00 | 14-12-2019 15-52-05 | - | 0/20 | C685-82C8-49B4-8D33 | c0e1b422-9cea-4fd8-9132-b0afaa348eb2
1124 | 8.9 | 12-06-2019 02-00-00 | 14-12-2019 15-52-05 | - | 0/20 | C685-82C8-49B4-8D33 | 8948c513-b040-421d-846f-0fd5883ed7d4
1125 | 8.9 | 12-06-2019 02-00-00 | 14-12-2019 15-52-05 | - | 0/20 | C685-82C8-49B4-8D33 | eb5f50e7f-f521-4e3e-ab84-10a08c819141
1126 | 8.9 | 12-06-2019 02-00-00 | 14-12-2019 15-52-05 | - | 0/20 | C685-82C8-49B4-8D33 | 55bc284f-6a8c-43a8-be08-2b0e63a868f3
1127 | 8.9 | 12-06-2019 02-00-00 | 14-12-2019 15-52-05 | - | 0/20 | C685-82C8-49B4-8D33 | ec0bd2a2-e85d-4c91-a15a-61a696ca9bb0
1128 | 8.9 | 12-06-2019 02-00-00 | 14-12-2019 15-52-05 | - | 0/20 | C685-82C8-49B4-8D33 | 098820e9-3391-4110-ba57-48b781577f6d
1129 | 8.9 | 12-06-2019 02-00-00 | 14-12-2019 15-52-05 | - | 0/20 | C685-82C8-49B4-8D33 | 981d3b4d-c28b-4888-9a0f-8ae72b853826
1130 | 8.9 | 12-06-2019 02-00-00 | 14-12-2019 15-52-05 | - | 0/20 | C685-82C8-49B4-8D33 | 25245b91-885c-4023-9e79-6c0768c383e
1131 | 8.9 | 12-06-2019 02-00-00 | 14-12-2019 15-52-05 | - | 0/20 | C685-82C8-49B4-8D33 | 810ae6ad-6bb9-4834-86ec-5d91e1b54675
1132 | 8.9 | 12-06-2019 02-00-00 | 14-12-2019 15-52-05 | - | 0/20 | C685-82C8-49B4-8D33 | 444336fd-59cf-4724-8279-6f8b67cc1ae0
1133 | 8.9 | 12-06-2019 02-00-00 | 14-12-2019 15-52-05 | - | 0/20 | C685-82C8-49B4-8D33 | 64eb415a-be2e-4000-bf77-abd00f195a46
1201 | 8.9 | 12-06-2019 02-00-00 | 14-12-2019 15-52-05 | - | 0/20 | C685-82C8-49B4-8D33 | b205427-c50f-427e-87c0-4da95e32db2e
1202 | 8.9 | 12-06-2019 02-00-00 | 14-12-2019 15-52-05 | - | 0/20 | C685-82C8-49B4-8D33 | 470b30b-fa9b-4d74-a8cb-d9e8bc360b39
25 | 7.0 | 30-04-2010 02-00-00 | 31-10-2019 15-52-05 | - | 0/3 | F963-83C5-EC6F-C1E7 | 53230811-69fd-44f3-a2ce-1d6b87b73f6c
396 | 23.1 | 30-04-2019 02-00-00 | 31-10-2019 15-52-05 | - | 0/3 | F963-83C5-EC6F-C1E7 | 40900116-886d-4fa7-8a7c-c0ed88fd09b9
399 | 7.0 | 30-04-2019 02-00-00 | 31-10-2019 15-52-05 | - | 0/3 | F963-83C5-EC6F-C1E7 | 351805df-4381-4ccf-b5fa-4dfde1bf4fd9
483 | 5.7 | 26-06-2019 02-00-00 | 28-12-2019 15-52-05 | - | 0/1 | 2793-9D44-E367-1302 | bad3f8dc-b08b-4aab-8d28-b048bd24f86d

```

#### 4.1.9 List license key users

Shows the list of the users and hosts, which are currently using the specified license key.

Syntax:

```
MatLicenseServer.exe --list-users <unique_id>
```

The command requires the license key's unique ID as a parameter, it can be obtained via output of the `--list` command.

Example:

```

C:\Program Files\Materialise\Floating License Server\7.0>MatLicenseServer.exe --list-users b47149d6-bb94-421d-be5c-df4cf33cfb58
Thu Jun 27 15:07:10 2019|PID 1380|THR 16044|Licensing|INFO|Connected to localhost:7010 from port 57015
PC Name | User Name
host1 | user1
host2 | user2
host3 | user3

```

#### 4.1.10 Delete license key

Deletes a license key from the storage.

Syntax:

```
MatLicenseServer.exe --delete <unique_id>
```

The command requires the license key's unique ID as a parameter, it can be obtained via output of the `--list` command.

Example:

```

C:\Program Files\Materialise\Floating License Server\7.0>MatLicenseServer.exe --delete bad3f8dc-b08b-4aab-8d28-b048bd24f86d
Thu Jun 27 15:09:59 2019|PID 8876|THR 20676|Licensing|INFO|Connected to localhost:7010 from port 57046
Thu Jun 27 15:09:59 2019|PID 8876|THR 20676|Licensing|INFO|removed license module: bad3f8dc-b08b-4aab-8d28-b048bd24f86d
license successfully deleted

```

## 4.2. Configuration file



### 4.2.1 Overview

The configuration of the License Server component is defined in the `settings.ini` file, which has to be located in the same folder as `MatLicenseServer.exe`.

If there is no configuration file available at the moment of License Server start, it is created by the server and filled in with default values. In order to apply changes in the file to the License Server, the latter must be restarted.

All values in the configuration file are case-insensitive.

### 4.2.2 General settings

The general configuration settings of the License Server component must be defined under the `[general]` section in the configuration file.

Below you can find an example of the configuration file content with the default general settings values.

```
[general]
port=7010
uselogfile=0
loglevel=1
logfilepath=C:\ProgramData\Materialise\LicenseFiles\LicenseServer7.log
```

#### **port**

The IP port used by the License Server for incoming connection.

Example:

#### **uselogfile**

A flag indicating if the License Server must generate log file during its work.

#### **logfilepath**

Path to the log file created by the License Server. This parameter is only valid if the `uselogfile` parameter is set to 1.

#### **loglevel**

The level of log detail (0 – trace, most thorough; 1 – info; 2 – warning; 3 – error). Default is 1.

### 4.2.3 License key access permissions

Materialise Floating License Server allows configuring access permissions to the license keys for specific users and hosts.

Access permissions can be configured using white- and blacklist modes:

- Blacklist defines the users/hosts, which are not allowed using the license keys; users, which are not in the blacklist, can use the requested license key;
- Whitelist defines the users/hosts, which are explicitly allowed using the license keys; users, which are not in the whitelist, cannot use the requested license key.

Both blacklists and whitelists can be defined for the following entities:

- Entire server
- Specific license modules
- Specific CCKeys

Specification of a blacklist has a precedence over a whitelist, e.g. if the same user/host is listed in both blacklist and whitelist for the same entity, the license server denies accessing the requested license key.

The following access permission check logic is used a software product requests a license key:

1. If the user or the host is in the blacklist for the entire server, access is denied. Otherwise proceed to step 2.
2. If a whitelist is defined for the entire server and neither the user nor the host is in this whitelist, access is denied. Otherwise proceed to step 3.
3. If the user or the host is in the blacklist for the requested module ID, access is denied. Otherwise proceed to step 4.
4. If a whitelist is defined for the requested module ID and neither the user nor the host is in this whitelist, access is denied. Otherwise proceed to step 5.
5. If the user or the host is in a blacklist for the CCKey the requested module belongs to, access is denied. Otherwise proceed to step 6.
6. If a whitelist is defined for the CCKey the requested module ID belongs to and neither the user nor the host is in this whitelist, access is denied. Otherwise access is granted.

### User and host groups definition

Access permissions to the license keys can be configured for specific users, host, but also user and host groups.

Configuration of user groups is done under the `[usersgroups]` section of the configuration file. Below you can find an example of defining the user groups named `u_group1` and

`u_group2`:

```
[usersgroups]
u_group1=john, steve u_group2=jane, sam
```

Configuration of host groups is done in a similar way, different section `[hostsgroups]` is used. Below you can find an example of defining the host groups `h_group1` and `h_group2`:

```
[hostsgroups]
h_group1=host1, host2 h_group2=host3, host4
```

## Server access permissions

Access permissions to the entire license server can be defined with the help of settings under the [general] section.

### **blacklistusers**

The list of users or user groups, which are prohibited from accessing any of the license keys registered on the license server.

Example:

```
[general]
blacklistusers=user1, u_group1
[usersgroups]
u_group1=user101, user102
```

### **whitelistusers**

The list of users or user groups, which are allowed to use license keys registered on the license server. Users, which are not in this list, are prohibited from using the license keys on the server.

Example:

```
[general]
whitelistusers=user1, user2, u_group2
[usersgroups]
u_group2=user201, user202
```

### **blacklisthosts**

The list of hosts or host groups, which are prohibited from accessing any of the license keys registered on the license server.

Example:

```
[general]
blacklisthosts=host1, h_group1
[hostsgroups]
h_group1=host101, host102
```

### **whitelisthosts**

The list of hosts or host groups, which are allowed to use license keys registered on the license server. Software products on hosts, which are not in this list, are prohibited from using the license keys on the server.

Example:

```
[general]
whitelisthosts=host1, host2, h_group2
[hostsgroups]
h_group2=host201, host202
```



### License module access permissions

Access permissions to specific hosts can be defined under the following sections:

- `blacklistmoduleusers` – defines users and user groups, which are not allowed using the specified license module;
- `whitelistmoduleusers` – defines users and user groups, which are explicitly allowed using the specified license module;
- `blacklistmodulehosts` – defines hosts and host groups, which are not allowed using the specified license module;
- `whitelistmodulehosts` – defines hosts and host groups, which are explicitly allowed using the specified license module.

In order to define access permissions to a specific module a setting must be added under one of sections listed above. The name of this setting must be the respective module ID, the value – a list of the users or hosts in the black- or whitelist respectively.

Example of blacklist definitions:

```
[usersgroups]
u_group1=user101, user102

[hostsgroups]
h_group1=host101, host102

[blacklistmoduleusers]
21=user1, user2, u_group1
22=user3

[blacklistmodulehosts]
399=host20, h_group1
```

Example of whitelist definitions:

```
[usersgroups]
u_group1=user101, user102

[hostsgroups]
h_group1=host101, host102

[whitelistmoduleusers]
21=user1, user2, u_group1
22=user3

[whitelistmodulehosts]
399=host20, h_group1
```

### CCKey access permissions

Access permissions to specific hosts can be defined under the following sections:

- `blacklistproductkeyusers` – defines users and user groups, which are not allowed using the specified license module;

- `whitelistproductkeyusers` – defines users and user groups, which are explicitly allowed using the specified license module;
- `blacklistproductkeyhosts` – defines hosts and host groups, which are not allowed using the specified license module;
- `whitelistproductkeyhosts` – defines hosts and host groups, which are explicitly allowed using the specified license module.

In order to define access permissions to a specific CCKey a setting must be added under one of sections listed above. The name of this setting must be the respective module ID, the value – a list of the users or hosts in the black- or whitelist respectively.

#### Example of blacklist definitions:

```
[usersgroups]
u_group1=user101, user102

[hostsgroups]
h_group1=host101, host102

[blacklistproductkeyusers]
C6B5-82C8-49B4-8D33=user1, user2, u_group1

[blacklistproductkeyhosts]
F963-83C5-EC6F-C1E7=host20, h_group1
```

#### Example of whitelist definitions:

```
[usersgroups]
u_group1=user101, user102

[hostsgroups]
h_group1=host101, host102

[whitelistproductkeyusers]
C6B5-82C8-49B4-8D33=user1, user2, u_group1

[whitelistproductkeyhosts]
F963-83C5-EC6F-C1E7=host20, h_group1
```

#### 4.2.4 License key reservation

The Materialise Floating License Server allows reserving certain number concurrent licenses for certain users or hosts. If a license is reserved in such a way for a user, other users cannot use it and therefore it always remains available for that user. For example, if a license module 21 registered on a license server has maximum number of connections 5, and 2 of them are reserved for USER1, other users can only consume 3 remaining connections even if the USER1 is not currently using this license module.

The license reservations can be defined in the configuration file with the help of reservation groups. Each such group is defined as a section according to the following format:

```
[reservationgroup <group_id>]
```

Each such section can contain settings with the following names.

### **modules**

Defines the list of modules, which must be reserved.

### **productkeys**

Defines the list of CCKeys, which must be reserved. If the one is defines, all the modules with this CCKey are reserved for specified users/hosts.

### **users**

Defines the list of users and user groups the specified modules must be reserved for.

### **hosts**

Defines the list of host and host groups the specified modules must be reserved for.

### **size**

The number of connection that must be reserved for specified users and modules. Default value is 1.

In the following example 2 connections to the license module 21 are reserved for user1, user2, user101, user102:

```
[usersgroups]
u_group1=user101, user102

[reservationgroup r1]
module=399
users=user1, user2, u_group1 size=2
```

If reservations for same license keys are defined in different groups, they add up. For example, license module 21 is registered as a part of the CCKey C6B5-82C8-49B4-8D33. Following reservations are defined:

1. Module 21, 2 connections reserved for user1, and user2.
2. CCKey C6B5-82C8-49B4-8D33, 3 connections reserved for user1 and user3.

In this case 5 connections will be reserved for user1.

If a reservation group contains both list of users and hosts, this reservation applies to users logged on these hosts. For example, the configuration file contains the following reservation definition:

```
[reservationgroup r2]
module=25
users=user1, user2 hosts=host1, host2
size=2
```

In this case the reserved licenses can be used by:

1. user1 on host1
2. user1 on host2
3. user2 on host1
4. user2 on host2

#### 4.2.5 Logging settings

Extended logging settings are configured via the [logger] section. This settings take precedence over the settings in the [general] section.

Below you can find an example of the configuration file content with the default logging settings values.

```
[logger]
uselogfile=1
loglevel=1
logfilepath=C:\ProgramData\Materialise\LicenseFiles\MatLicenseServer7_logs\Li
censeServer_%Y%m%d_%H%M%S_%5N.log
rotatelog=1
logfilerotationsize=134217728 ; 128MB
maxtotallogssize=2147483648 ; 2GB
minfreediskspace=2147483648 ; 2 GB
```

#### UseLogfile

A flag indicating if the License Server must generate log file during its work.

#### LogFilepath

Path to the log file created by the License Server. Supports specifying path as a pattern with counter (%N) and date time (%Y,%m,%d,%H,%M,%S) placeholders

#### LogLevel

The level of log detail (0 – trace, most thorough; 1 – info; 2 – warning; 3 – error). Default is 1.

**RotateLogs**

Whether to enable log rotation based on the log files size.

**LogFileRoationSize**

The size of the file at which rotation should occur, in bytes.

**MaxTotalLogsSize**

The maximum total size of log files in folder, in bytes. If the threshold is exceeded, the oldest file(s) is deleted.

**MinFreeDiskSpace**

Minimum free space on disk, in bytes. If the threshold is exceeded, the oldest file(s) is deleted to free space.

#### 4.2.6 SSL settings

HTTPS SSL/TLS settings are configured via the [ssl] section. Below you can find an example of the configuration file content with the default settings values.

```
[ssl]
cipherlist= ALL:!ADH:!LOW:!EXP:!MD5:!RC4:!3DES:@STRENGTH
```

**CipherList**

Specifies the supported ciphers in OpenSSL notation (see [Appendix A](#)).

Default cipher list string means using all available ciphers except ciphers containing ADH,LOW,EXP,MD5,RC4 and 3DES algorithms; the ciphers are ordered/prioritized by the encryption algorithm key length, so by their reliability.

After updating this value the services MatFloatingLicenseServer60 and MatLicenseServer7 require a restart.

## 5 Managing Proxy Service

### 5.1. Command line interface

A certain set of operations on the License Server can be done using the command line interface provided by `LicSrvFS60.exe`. This section describes the available commands.

#### 5.1.1 Register service

Registers the `LicSrvFS60.exe` as “Materialise Floating License Server 6.0” Windows service.

Syntax:

```
LicSrvFS60.exe /r
```

The command requires administrator rights to execute.

#### 5.1.2 Unregister service

Unregisters the “Materialise Floating License Server 6.0” Windows service. Syntax:

```
LicSrvFS60.exe /u
```

The command requires administrator rights to execute.

#### 5.1.3 Start service

Starts the “Materialise Floating License Server 6.0” Windows service. Syntax:

```
LicSrvFS60.exe /start
```

```
LicSrvFS60.exe /s
```

The command requires administrator rights to execute. Please note that the Proxy Service component cannot be started without first starting the License Server component.

#### 5.1.4 Stop service

Stops the “Materialise Floating License Server 6.0” Windows service. Syntax:

```
LicSrvFS60.exe /stop
```

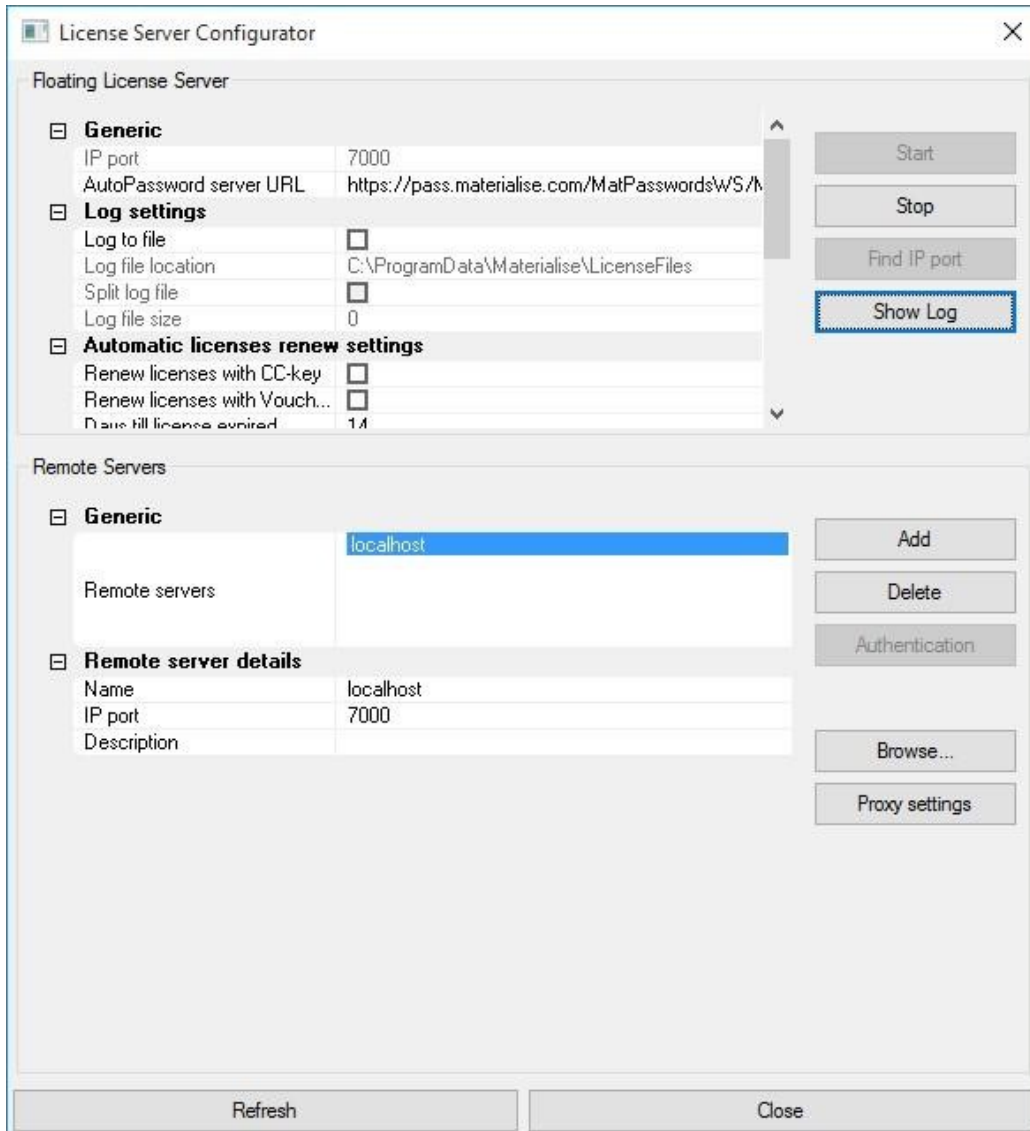
The command requires administrator rights to execute.

### 5.2. Configuration UI tool

The Proxy Service configuration tool provide evolved from the License Server Configurator application installed next to the license server (both local and floating) version 6.x. At the

moment it can only be used to configure the settings of the Proxy Server, the settings of the License Server can be configured via its INI-file.

The configuration tool can be launched with the help of the `LicSrvConfig.exe` executable file installed next to the `LicSrvFS60.exe`.



The main screen contains two major sections:

1. Floating License Server contains the parameters of the Server Proxy.
2. Remote Servers can be used to edit the connections to the license servers. These settings are used by the software products installed on the same host.

This chapter contains the description of the parameters in the Floating License Server section.

### 5.2.1 Control buttons

The following control buttons are available in the Floating License Server section.



### Start

Starts the Proxy Service. The button is disabled if the service is already running.

### Stop

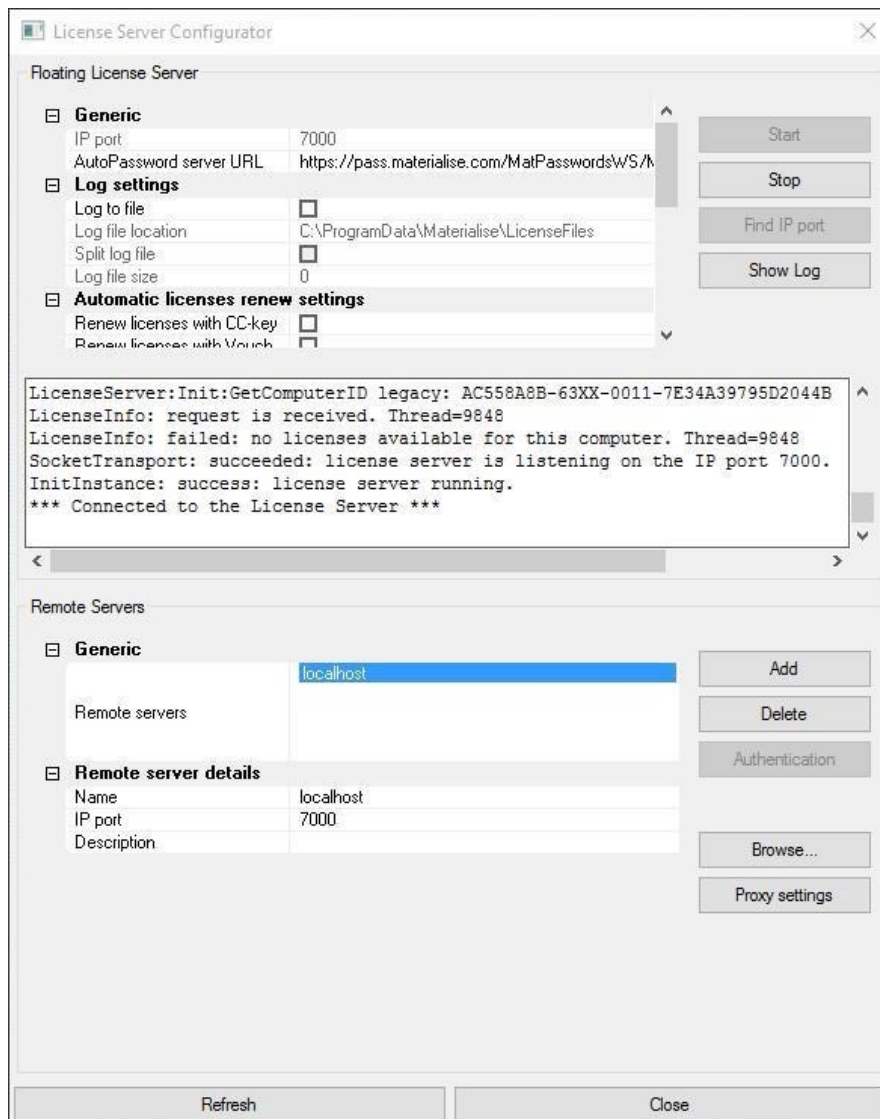
Stops the Proxy Service. The button is disabled if the service is not running.

### Find IP port

Automatically finds available IP port that can be used by the Service Proxy. The button is disabled if the service is running.

### Show Log

Toggles showing the Service Proxy log output.







## 5.2.2 Generic

### **IP port**

The value of an IP port used by the Proxy Service, the parameter can only be edited if the Proxy Service is stopped.

### **AutoPassword server URL**

The URL of the AutoPassword server, which the Proxy Service connects to when performing the license keys automatic renewals operation.

## 5.2.3 Log settings

This section contains parameters used by the Proxy Service for generating the log file.

### **Log to file**

A flag indicating if logging to a file must be turned on.

### **Log file location**

Path to a folder where the log file will be created. Only used if the logging to file is on.

### **Split log file**

A flag indicating if the log file must be split if it reaches some maximum size value. Only used if the logging to file is on.

### **Log file size**

Maximum size of the generated log file. If the file becomes bigger, it's being rolled over.

## 5.2.4 Automatic licenses renew settings

This section contains parameters for the automatic renewal of the license keys using the AutoPassword Service.

### **Renew licenses with CC-key**

Turn the automatic renewal of the license keys generated for CCKeys ON or OFF.

### **Renew licenses with Vouchers**

Turn the automatic renewal of the license keys generated for vouchers ON or OFF.

### **Days till license expired**

Number of days before the license key expiration when the auto-renewal will be initiated.



## 5.2.5 Mail settings

This section contains the settings used by the Proxy Service to use the license key notification e-mails.

### **Send e-mail notification**

Turn the e-mail notifications ON or OFF.

### **Days till license expired**

The number of days till the license keys expiration when an e-mail notification will be sent. Only used when e-mail notifications are turned on.

### **SMTP server**

The address of the SMTP server used to send the e-mails. Only used when e-mail notifications are turned on.

### **SMTP port**

The IP port number on the SMTP server, which is used to send the e-mails. Only used when e-mail notifications are turned on.

### **Security type**

The security type setting for the SMTP server. Three options are available:

1. No security
2. SSL
3. TLS

Only used when e-mail notifications are turned on.

### **Username**

User name required to logon to the SMTP server. Only used when e-mail notifications are turned on.

### **Password**

The password required to logon to the SMTP server. Only used when e-mail notifications are turned on.

### **From**

Sender's e-mail address. Only used when e-mail notifications are turned on.

**To** Receiver's e-mail address. Only used when e-mail notifications are turned on.

### **Subject**

Text used as notification e-mails subject. Only used when e-mail notifications are turned on.



## 6 Managing License API Server

### 6.1. Command line interface

A certain set of operations on the License Server can be done using the command line interface provided by `MatLicenseAPIServer.exe`. This section describes the available commands.

#### 6.1.1 Show help

Shows the list of available commands with a brief description for each.

Syntax:

```
MatLicenseAPIServer.exe --help
```

#### 6.1.2 Register service

Registers the `MatLicenseAPIServer.exe` as a “Materialise License API Server 7” Windows service. Syntax:

```
MatLicenseAPIServer.exe --register
```

```
MatLicenseAPIServer.exe -r
```

The command requires administrator rights to execute.

#### 6.1.3 Unregister service

Unregisters the “Materialise License API Server 7” Windows service. Syntax:

```
MatLicenseAPIServer.exe --unregister
```

```
MatLicenseAPIServer.exe -u
```

The command requires administrator rights to execute.

#### 6.1.4 Start service

Starts the “Materialise License API Server 7” Windows service. Syntax:

```
MatLicenseAPIServer.exe --start
```

The command requires administrator rights to execute.

#### 6.1.5 Stop service

Stops the “Materialise License API Server 7” Windows service. Syntax:

```
MatLicenseAPIServer.exe --stop
```

The command requires administrator rights to execute.

### 6.1.6 Restart service

Stops and starts the “Materialise License API Server 7” Windows service. Syntax:

```
MatLicenseAPIServer.exe --restart
```

The command requires administrator rights to execute.

## 6.2. Configuration file

### 6.2.1 Overview

The configuration of the License API Server component locates in the

`MatLicenseAPIServer.ini` file in the same folder as `MatLicenseAPIServer.exe`.

If there is no configuration file available at the moment of License API Server start, it is created by the server and filled in with default values. To apply changes in the file to the License API Server, you need to restart the service.

All keys in the configuration file are case-insensitive.

### 6.2.2 General settings

The general configuration settings of the License API Server component are defined under the [general] section in the configuration file.

Below you can find an example of the configuration file content with the default general settings values.

```
[general]
port=7011
uselogfile=1
loglevel=1
logfilepath=C:\ProgramData\Materialise\LicenseFiles\LicenseAPIServer7
```

#### **port**

The IP port used by the License Server for incoming connections.

#### **uselogfile**

A flag for enabling log file generation by the License API Server.

#### **logfilepath**

Path to the log file to be created by the License API Server. This parameter is only valid if the `uselogfile` parameter is set to 1.

#### **loglevel**

Specifies generated log verbosity level. 0 – most detailed logs (trace), 1 – info, 2 – warning, 3 – only errors. Default value is 1.

### 6.2.3 License Server connection settings

License API Server works as a proxy for License Server and requires an established connection to work properly. This connection configuration is defined under the [licenseserver] section in the configuration file.

Below you can find an example of the configuration file content with the default general settings values.

```
[licenseserver]
port=7010
host=localhost
vendorname=Materialise Floating License Server
```

#### **port**

The IP port used to connect to the License Server.

#### **host**

The hostname used to connect to the License Server.

#### **vendorname**

Vendor name of the License Server. Used in the server API model.

### 6.2.4 Logging settings

Extended logging settings are configured via the [logger] section. This settings take precedence over the settings in the [general] section.

Below you can find an example of the configuration file content with the default logging settings values.

```
[logger]
uselogfile=1
loglevel=1
logfilepath=C:\ProgramData\Materialise\LicenseFiles\MatLicenseServer7_logs\LicenseServer_%Y%m%d_%H%M%S_%5N.log

rotatelogs=1
logfilerotationsize=134217728 ; 128MB
maxtotallogssize=2147483648 ; 2GB
minfreediskspace=2147483648 ; 2 GB
```

### UseLogfile

A flag indicating if the License Server must generate log file during its work.

### LogFilePath

Path to the log file created by the License Server. Supports specifying path as a pattern with counter (%N) and date time (%Y,%m,%d,%H,%M,%S) placeholders

### LogLevel

The level of log detail (0 – trace, most thorough; 1 – info; 2 – warning; 3 – error). Default is 1.

### RotateLogs

Whether to enable log rotation based on the log files size.

### LogFileRoationSize

The size of the file at which rotation should occur, in bytes.

### MaxTotalLogsSize

The maximum total size of log files in folder, in bytes. If the threshold is exceeded, the oldest file(s) is deleted.

### MinFreeDiskSpace

Minimum free space on disk, in bytes. If the threshold is exceeded, the oldest file(s) is deleted to free space.

## 6.2.5 SSL settings

HTTPS SSL/TLS settings are configured via the [ssl] section.

```
[ssl]
certificate=... ; path to file
private_key=... ; path to file
private_key_password=... ; path to file
ca_certificate=... ; path to file
verify=false
verification_depth=9
cipherlist= ALL:!ADH:!LOW:!EXP:!MD5:!RC4:!3DES:@STRENGTH
```

### certificate

Contains the path to the certificate file (in PEM format).



**private\_key**

Contains the path to the private key file used for encryption. If empty, default private key file is used.

**private\_key\_password**

Contains password to the private key file used for encryption.

**ca\_certificate**

Contains the path to the file or directory containing the CA/root certificates.

**verify**

Specifies whether peer verification should be enabled.

**verification\_depth**

Sets the upper limit for verification chain sizes. Verification will fail if a certificate chain larger than this is encountered.

**CipherList**

Specifies the supported ciphers in OpenSSL notation (see [Appendix A](#)).

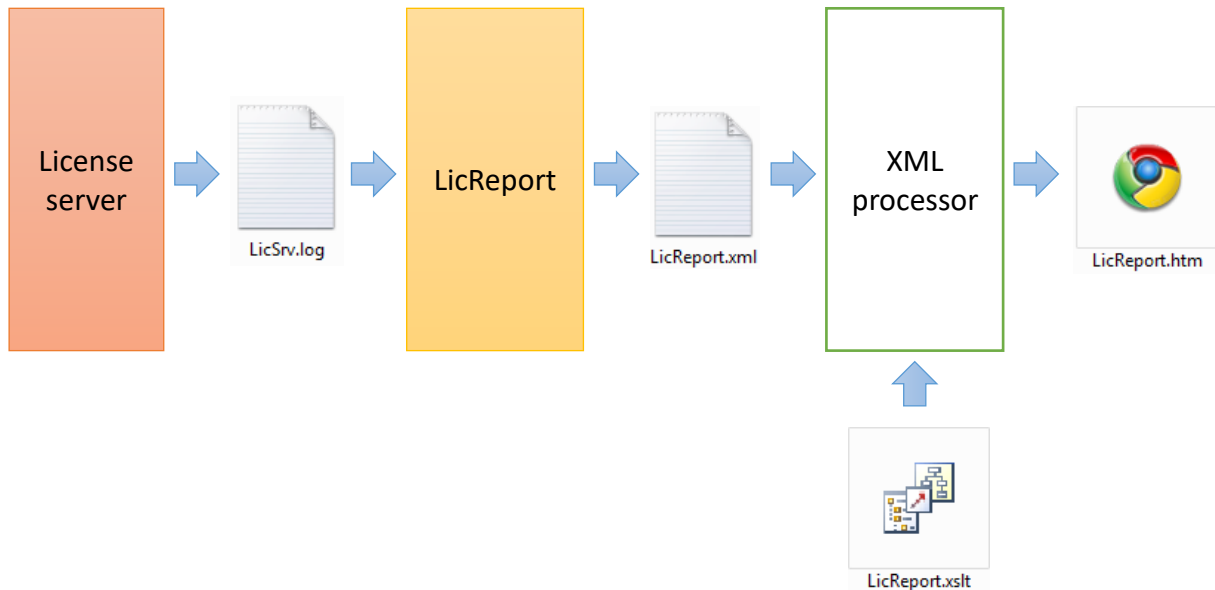
Default cipher list string means using all available ciphers except ciphers containing ADH,LOW,EXP,MD5,RC4 and 3DES algorithms; the ciphers are ordered/prioritized by the encryption algorithm key length, so by their reliability.

After updating this value the services MatFloatingLicenseServer60 and MatLicenseServer7 require a restart.

## 7 Generating license reports with LicReport Tool

### 7.1 Introduction

The License server reporting tool allows generating easy readable reports from about the license server's work. Below the reporting tool's workflow is demonstrated:



The License server generates a log file during its work. The log file is read by the LicReport application, its content is analyzed and the result of the analysis is saved in an XML file that can be used for further screening.

Besides the LicReport saves the hardcoded version of the XSL transformation file that can be used for the LicReport.xml for its transformation to the HTML format.

### 7.2 LicReport usage

LicReport tool comes with the installation of the LicAdmin. Its default location is C:\Program Files\Materialise\Floating License Server\LicAdmin\LicReport.exe. It can also be delivered as a command line tool named LicReport.exe, its work is regulated by the command line parameters.

There are three main usage modes: license module usage report, log entries report and server license list report.





## 7.2.1 License module usage report


This mode is used to generate the output containing the information about the separate license modules usage duration. By default the XML file named LicReport.xml is generated, its schema is demonstrated below:

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSPY v5 rel. 4 U (http://www.xmlspy.com) by Johan Duchateau (Materialise N.V.) -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified" attributeFormDefault="unqualified">
  <xs:element name="Report">
    <xs:annotation>
      <xs:documentation>Root element</xs:documentation>
    </xs:annotation>
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Generated" type="xs:dateTime">
          <xs:annotation>
            <xs:documentation>Date/time when the report is generated</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="Usages">
          <xs:annotation>
            <xs:documentation>License usages information</xs:documentation>
          </xs:annotation>
          <xs:complexType>
            <xs:sequence>
              <xs:element name="Module" minOccurs="0" maxOccurs="unbounded">
                <xs:annotation>
                  <xs:documentation>Specific module usage information</xs:documentation>
                </xs:annotation>
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="ID" type="xs:integer">
                      <xs:annotation>
                        <xs:documentation>Module ID</xs:documentation>
                      </xs:annotation>
                    </xs:element>
                    <xs:element name="Name" type="xs:string">
                      <xs:annotation>
                        <xs:documentation>Module name</xs:documentation>
                      </xs:annotation>
                    </xs:element>
                    <xs:element name="Version" type="xs:string">
                      <xs:annotation>
                        <xs:documentation>Module version</xs:documentation>
                      </xs:annotation>
                    </xs:element>
                    <xs:element name="Type" type="xs:integer">
                      <xs:annotation>
                        <xs:documentation>License type: 0 - license, 1 - evaluation</xs:documentation>
                      </xs:annotation>
                    </xs:element>
                    <xs:element name="Platform" type="xs:integer">
                      <xs:annotation>
                        <xs:documentation>License platform: 0 - win32, 3 - x64</xs:documentation>
                      </xs:annotation>
                    </xs:element>
                    <xs:element name="Blocks">
                      <xs:annotation>
                        <xs:documentation>Specific module usage blocks</xs:documentation>
                      </xs:annotation>
                      <xs:complexType>
                        <xs:sequence>
                          <xs:element name="Usage" minOccurs="0" maxOccurs="unbounded">
                            <xs:annotation>
```

```

<xs:documentation>Module usage block</xs:documentation>
</xs:annotation>
<xs:complexType>
  <xs:sequence>
    <xs:element name="Day" type="xs:date">
      <xs:annotation>
        <xs:documentation>Date part of the license acquiring moment</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="Time" type="xs:time">
      <xs:annotation>
        <xs:documentation>Time part of the license acquiring moment</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="Duration" type="xs:duration">
      <xs:annotation>
        <xs:documentation>License usage duration</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="Seconds" type="xs:integer">
      <xs:annotation>
        <xs:documentation>License usage duration in seconds</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="TypeEx" type="xs:integer">
      <xs:annotation>
        <xs:documentation>0 - real license, 1 - temporary</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="Host" type="xs:string">
      <xs:annotation>
        <xs:documentation>Host the license is obtained by</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="Timeout" type="xs:boolean">
      <xs:annotation>
        <xs:documentation>Specifies if the license is released on timeout</xs:documentation>
      </xs:annotation>
    </xs:element>
  </xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="Duration" type="xs:duration">
  <xs:annotation>
    <xs:documentation>Sum of all the module usage blocks durations</xs:documentation>
  </xs:annotation>
</xs:element>
<xs:element name="Seconds" type="xs:integer">
  <xs:annotation>
    <xs:documentation>Sum of all the module usage blocks durations in seconds</xs:documentation>
  </xs:annotation>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="Duration" type="xs:duration">
  <xs:annotation>
    <xs:documentation>Sum of all the license module usages</xs:documentation>
  </xs:annotation>
</xs:element>
<xs:element name="Seconds" type="xs:integer">
  <xs:annotation>
    <xs:documentation>Sum of all the license module usages in seconds</xs:documentation>
  </xs:annotation>
</xs:element>

```



```
</xs:element>  
</xs:sequence>  
</xs:complexType>  
</xs:element>  
</xs:schema>
```

The LicReport can analyze log files both for the local and the floating version, in the second case it must run on the same PC where the floating server is installed.

The log file to analyze can be specified using one of two command line keys:

```
--find-file [local|floating]
```

this way the LicReport will try to locate the path to the log file using the settings of the respective license server;

```
--input-file <path>
```

this way the exact location of the log file can be specified.

It is necessary to set one of the keys shown above, otherwise the LicReport will return.

It is also possible to specify the time period for the log entries to be analyzed using the following keys:

```
--from-date <yyyy-mm-dd hh:mm:ss>  
--until-date <yyyy-mm-dd hh:mm:ss>
```

If the period is not specified, the entire log file will be used.

The path to the output file can be set using the following command line key:

```
--output-file <path>
```

If it is not set, the file named LicReport.xml will be generated next to the input log file.

It is possible to generate module usages in the CSV format by using the `--csv-output` key. The report will contain the following headers by default: `UsageID, ModuleID, Module, Version, Type, TypeEx, Platform, UserName, Host, Day, Time, Duration, Seconds, Timeout`. It is possible to select only specific headers by providing a comma-separated list of headers to as a command argument.



## 7.2.2 Log entries report

Using this mode an XML file will be generated with the plain list of the log entries of the following types:

- License check-out
- License check-in
- License request errors

The schema of the output file is demonstrated below:

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSPY v5 rel. 4 U (http://www.xmlspy.com) by Johan Duchateau (Materialise N.V.) -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified" attributeFormDefault="unqualified">
  <xs:element name="Report">
    <xs:annotation>
      <xs:documentation>Root element</xs:documentation>
    </xs:annotation>
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Generated" type="xs:dateTime">
          <xs:annotation>
            <xs:documentation>Date/time when the report is generated</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="LogEntry" minOccurs="0" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="Day" type="xs:date">
                <xs:annotation>
                  <xs:documentation>The date of the log entry</xs:documentation>
                </xs:annotation>
              </xs:element>
              <xs:element name="Time" type="xs:time">
                <xs:annotation>
                  <xs:documentation>The time of the log entry</xs:documentation>
                </xs:annotation>
              </xs:element>
              <xs:element name="ModuleID" type="xs:integer">
                <xs:annotation>
                  <xs:documentation>The license module ID operated</xs:documentation>
                </xs:annotation>
              </xs:element>
              <xs:element name="ModuleName" type="xs:string">
                <xs:annotation>
                  <xs:documentation>The license module name operated</xs:documentation>
                </xs:annotation>
              </xs:element>
              <xs:element name="ModuleVersion" type="xs:string">
                <xs:annotation>
                  <xs:documentation>The license module version operated</xs:documentation>
                </xs:annotation>
              </xs:element>
              <xs:element name="Host" type="xs:string">
                <xs:annotation>
                  <xs:documentation>The host name the license request came from</xs:documentation>
                </xs:annotation>
              </xs:element>
              <xs:element name="EntryType" type="LogEntryType"/>
            <xs:choice>
              <xs:element name="EntryGetLicense">
                <xs:annotation>
                  <xs:documentation>Added if EntryType equal to "GetLicense"</xs:documentation>
                </xs:annotation>
            </xs:choice>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

```

<xs:complexType>
  <xs:sequence>
    <xs:element name="Type" type="xs:integer">
      <xs:annotation>
        <xs:documentation>License type obtained: 0 - license, 1 - evaluation</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="TypeEx" type="xs:integer">
      <xs:annotation>
        <xs:documentation>0 - real license obtained, 1 - temporary one</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="Platform" type="xs:integer">
      <xs:annotation>
        <xs:documentation>License platform obtained: 0 - win32, 3 - x64</xs:documentation>
      </xs:annotation>
    </xs:element>
  </xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="EntryReleaseLicense">
  <xs:annotation>
    <xs:documentation>Added if EntryType equal to "ReleaseLicense"</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Type" type="xs:integer">
        <xs:annotation>
          <xs:documentation>License type released: 0 - license, 1 - evaluation</xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="TypeEx" type="xs:integer">
        <xs:annotation>
          <xs:documentation>0 - real license released, 1 - temporary one</xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="Platform" type="xs:integer">
        <xs:annotation>
          <xs:documentation>License platform obtained: 0 - win32, 3 - x64</xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="Timeout" type="xs:boolean">
        <xs:annotation>
          <xs:documentation>Flag specifying if the license was released on timeout</xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="EntryError">
  <xs:annotation>
    <xs:documentation>Added if EntryType equal to "Error"</xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Reason" type="xs:string">
        <xs:annotation>
          <xs:documentation>The error reason</xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
</xs:choice>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>

```

```

<xs:simpleType name="LogEntryType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="GetLicense"/>
    <xs:enumeration value="ReleaseLicense"/>
    <xs:enumeration value="Error"/>
  </xs:restriction>
</xs:simpleType>
</xs:schema>

```

The same command line keys as for the usage report are used; the plain list mode is activated using the following parameter:

```
--plain-output
```

By default an XML file named LicReportPlain.xml will be created next to the input log files unless another output file name is specified using the `-output-file` key.

### 7.2.3 License list report

This mode generates an XML file containing the list of the license modules currently registered on a license server. Both local and floating license servers can be queried, even the floating ones installed on remote PCs.

The schema of the output XML is demonstrated below:

```

<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSPY v5 rel. 4 U (http://www.xmlspy.com) by Johan Duchateau (Materialise N.V.) -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified" attributeFormDefault="unqualified">
  <xs:element name="Report">
    <xs:annotation>
      <xs:documentation>Root element</xs:documentation>
    </xs:annotation>
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Generated" type="xs:dateTime">
          <xs:annotation>
            <xs:documentation>Date/time when the report is generated</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="SystemID">
          <xs:annotation>
            <xs:documentation>The System ID for the license server</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="Licenses">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="LicenseInfo" minOccurs="0" maxOccurs="unbounded">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="ID" type="xs:integer">
                      <xs:annotation>
                        <xs:documentation>License module ID</xs:documentation>
                      </xs:annotation>
                    </xs:element>
                    <xs:element name="LicVersion" type="xs:integer">
                      <xs:annotation>
                        <xs:documentation>Module licensing version</xs:documentation>
                      </xs:annotation>

```

```

</xs:annotation>
</xs:element>
<xs:element name="Name" type="xs:string">
  <xs:annotation>
    <xs:documentation>License module name</xs:documentation>
  </xs:annotation>
</xs:element>
<xs:element name="Platform" type="xs:integer">
  <xs:annotation>
    <xs:documentation>Module platform: 0 - win32, 3 - x64</xs:documentation>
  </xs:annotation>
</xs:element>
<xs:element name="Version" type="xs:string">
  <xs:annotation>
    <xs:documentation>Module version</xs:documentation>
  </xs:annotation>
</xs:element>
<xs:element name="Type" type="xs:integer">
  <xs:annotation>
    <xs:documentation>Module license type: 0 - license, 1 - evaluation</xs:documentation>
  </xs:annotation>
</xs:element>
<xs:element name="TypeEx" type="xs:integer">
  <xs:annotation>
    <xs:documentation>1 for temporary license</xs:documentation>
  </xs:annotation>
</xs:element>
<xs:element name="StartDate" type="xs:date">
  <xs:annotation>
    <xs:documentation>Validity period start date</xs:documentation>
  </xs:annotation>
</xs:element>
<xs:element name="EndDate" type="xs:date">
  <xs:annotation>
    <xs:documentation>Validity period end date</xs:documentation>
  </xs:annotation>
</xs:element>
<xs:element name="DaysLeft" type="xs:integer">
  <xs:annotation>
    <xs:documentation>Number of days till the expiration date</xs:documentation>
  </xs:annotation>
</xs:element>
<xs:element name="NumUsers" type="xs:integer">
  <xs:annotation>
    <xs:documentation>Number of users the liense is generated for (0 for local licenses)</xs:documentation>
  </xs:annotation>
</xs:element>
<xs:element name="UsageCont" type="xs:integer">
  <xs:annotation>
    <xs:documentation>Number of users currently using the license</xs:documentation>
  </xs:annotation>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:schema>

```

The mode is activated using the following command line key:

```
--licence-list [local|floating]
```

If the floating license server is requested, it is recommended to specify its parameters:

```
--fls-address <host>
--fls-ipport <ip-port>
```

If these parameters are not set, the default values will be used: localhost for the floating server host and 7000 for the IP port number.

The output will be saved in the file named LicList.xml next to the path to log file of the license server of requested type, unless another location is specified by the `-output-file` parameter.

It is possible to generate module usages in the CSV format by using the `--csv-output` key. The report will contain the following headers: `ID`, `LicVersion`, `Name`, `Platform`, `Version`, `Type`, `TypeEx`, `StartDate`, `Period`, `DaysLeft`, `NumUsers`, `UsageCount`. It is possible to select only specific headers by providing a comma-separated list of headers to as a command argument.

### 7.3 Command line parameters list

Parameter	Description
<code>--help</code>	Shows help information
<code>--find-file</code> [local floating]	Detects the file location based on the settings of the respective license server (the floating server must be installed on the same PC). If not set, the <code>-input-file</code> key will be used. Ignored if the <code>-license-list</code> key is set.
<code>--input-file</code> <path>	Specifies the path to the log file to analyze. If not set, the <code>-find-file</code> key will be used. Ignored if the <code>--license-list</code> key is set.
<code>--input-dir</code> <path>	Specifies the path to the directory with log files to analyze. If not set, the <code>-find-file</code> key will be used. Ignored if the <code>--license-list</code> or <code>--input-file</code> key is set.
<code>--output-file</code> <path>	Specifies the path to the output XML file. If it is not set, the output file will be generated next to the input log file.
<code>--from-date</code> <date time>	The log entries earlier than this date/time will not be analyzed. If not set, the earliest log entry to be analyzed will be the first one in the log file. Ignored if the <code>--license-list</code> key is set.
<code>--until-date</code> <date time>	The log entries later than this date/time will not be analyzed. If not set, the latest log entry to be analyzed will be the last one in the log file. Ignored if the <code>-license-list</code> key is set.
<code>--plain-output</code>	Specifies if the plain log entries list must be generated. Ignored if the <code>-license-list</code> key is set.
<code>--license-list</code> [local floating]	The license list registered on the license server specified will be generated.
<code>--fls-address</code>	The floating server host to retrieve the license list from. Used only if the <code>-license-list</code> floating is specified.





<code>--fls-ipport</code>	The floating server IP port to retrieve the license list from. Used only if the <code>-license-list floating</code> is specified.
<code>--no-xsl</code>	Turns off the generation of the hardcoded XSL files.
<code>--csv-output</code> [csv headers]	Turns on generation of the report in CSV format.
<code>--skip-orphans</code>	Enables skipping orphaned log events (reserve without release and vice versa).

## Appendix A: Cipher List Format

The cipher list consists of one or more *cipher strings* separated by colons. Commas or spaces are also acceptable separators but colons are normally used, e.g.:

```
ALL:!ADH:!LOW:!EXP:!MD5:!RC4:!3DES:@STRENGTH
```

The actual cipher string can take several different forms.

It can consist of a single cipher suite such as **RC4-SHA**.

It can represent a list of cipher suites containing a certain algorithm, or cipher suites of a certain type. For example **SHA1** represents all ciphers suites using the digest algorithm SHA1 and **SSLv3** represents all SSL v3 algorithms.

Lists of cipher suites can be combined in a single cipher string using the **+** character. This is used as a logical **and** operation. For example **SHA1+DES** represents all cipher suites containing the SHA1 **and** the DES algorithms.

Each cipher string can be optionally preceded by the characters **!**, **-** or **+**.

If **!** is used then the ciphers are permanently deleted from the list. The ciphers deleted can never reappear in the list even if they are explicitly stated.

If **-** is used then the ciphers are deleted from the list, but some or all of the ciphers can be added again by later options.

If **+** is used then the ciphers are moved to the end of the list. This option doesn't add any new ciphers it just moves matching existing ones.

If none of these characters is present then the string is just interpreted as a list of ciphers to be appended to the current preference list. If the list includes any ciphers already present they will be ignored: that is they will not be moved to the end of the list.

The cipher string **@STRENGTH** can be used at any point to sort the current cipher list in order of encryption algorithm key length.

The cipher string **@SECLEVEL=*n*** can be used at any point to set the security level to *n*, which should be a number between zero and five, inclusive. See [SSL\\_CTX\\_set\\_security\\_level](#) for a description of what each level means.

The cipher list can be prefixed with the **DEFAULT** keyword, which enables the default cipher list as defined below. Unlike cipher strings, this prefix may not be combined with other strings using **+** character. For example, **DEFAULT+DES** is not valid.

The content of the default list is determined at compile time and normally corresponds to **ALL:!COMPLEMENTOFDEFAULT:!eNULL**.