Semaphore Software LLC



User's Manual

October 28, 2026

ParseLogsToDB

Table of Contents

1 Introduction 3
2 Requirements 3
3 Platform Support 3
4 Initial Setup 4
4.1 Obtain and Unpack the Software 4
4.2 Choose a Working Folder 4
4.3 Verify Functionality 4
5 Program Overview 5
6 Information Commands 5
6.1 help 5
6.2 version 5
7 Core Commands 6
7.1 check-new 6
7.2 deactivate 6
7.3 machine-code 6
7.4 license 7
7.4.1 Offline (Air-Gapped) License Activation 7
7.5 process 8
8 Log File Requirements and Processing Behavior 8
9 Database Behavior 8
10 JSON File Format and Options 9
10.1.1 Required Keys 9
10.1.2 Optional Keys 9
10.1.3 Optional Email Keys 10
10.1.4 Template Config File (no email) 12
10.1.5 Template Config File (SMTP) 13
10.1.6 Template Config File (Mailtrap) 14
10.1.7 JSON File Guidelines 14
10.1.8 Protecting Credentials 14
11 Log Files and Execution Report Email 15
12 Troubleshooting 15

1 Introduction

ParseLogsToDB is a command-line interface (CLI) application for parsing signal logs and inserting the information into a perpetual database table. The application uses a configurable JSON file to provide the required configuration.

2 Requirements

- Operating System: Windows 10/11.
- Database: An existing SQL Server or PostgreSQL database in which ParseLogsToDB can create the configured table and perform SELECT, INSERT, UPDATE, and DELETE operations.
- License: Valid ParseLogsToDB license key.
- Disk Space: Around 175 MB.
- Installation: No installer. Unzip files and run the executable.
- ODBC Driver (SQL Server only): Install the driver named in the server_query configuration (for example, ODBC Driver 18 for SQL Server).
- PostgreSQL support is built in and needs no separate driver.
- Network: Access to target equipment via IP address. Internet access is needed for online license activation / deactivation and version checking. Air-gapped machines can use offline activation.
- Working Folder: Write access is required because the application stores its license certificate and diagnostic logs in the current working folder.

3 Platform Support

ParseLogsToDB can parse logs from the following platforms and log types.

1. ElectroLogIXS
    a. Crossing Events
    b. Data

4 Initial Setup

4.1 Obtain and Unpack the Software

ParseLogsToDB will be provided in a ZIP file containing a Revision History, End User License Agreement, this User's Manual, and the files needed to run the ParseLogsToDB executable.

Unpack the complete contents of the ZIP file to an empty folder (recommended to be named ParseLogsToDB). Keep the executable and all accompanying files together.

Review the End User License Agreement, the entirety of the User's Manual, and other documentation before proceeding.

4.2 Choose a Working Folder

Always launch ParseLogsToDB from the same writable working folder. The application stores cert.skm and debug.log in the current working folder, not automatically beside the executable. If a shortcut or scheduled task is used, set its Start in location to the ParseLogsToDB installation folder.

4.3 Verify Functionality

Open a command prompt window (Windows key **"cmd"** + Enter) and change directory to the folder that contains the ParseLogsToDB executable (**cd "[YOUR_FILE_PATH]"**).

Enter the command **ParseLogsToDB --version** and confirm the current version number is displayed.

```
C:\[YOUR_FILE_PATH]>ParseLogsToDB --version
ParseLogsToDB.exe, version 1.0.0
```

5 Program Overview

ParseLogsToDB is used to parse previously downloaded signal log files and insert them into a database table based on a configuration provided through a JSON file. All program functionality is controlled through a CLI (Command Line Interface) using commands whose definitions or usage are provided below as well as through the **`--help`** flag that may be placed after any of the commands to provide usage details.

The program requires a valid license key which must be provided through the **`license`** command. Once input, the key will be retained indefinitely. The license key can be removed from a device by using the **`deactivate`** command, which will free up the license for use on a different device.

The user can check if there is an updated version of ParseLogsToDB available by using the **`check-new`** command.

The main program function is invoked by using the **`process`** command in conjunction with the name of a JSON configuration file.

The details for usage of all these commands can be found in their own section of this manual, or as previously mentioned by appending the **`--help`** flag behind any of these commands in the CLI.

6 Information Commands

6.1 help

The **`--help`** flag will display information and list all available commands for whatever it is appended to. In the case of using **`--help`** after invoking the CLI directly, information will be provided about the core commands available.

```
C:\[YOUR_FILE_PATH]>ParseLogsToDB --help
Usage: ParseLogsToDB.exe [OPTIONS] COMMAND [ARGS]...

Options:
  --version  Show the version and exit.
  --help     Show this message and exit.

Commands:
  check-new
  deactivate
  license
  machine-code
  process
```

6.2 version

The **`--version`** flag will display the version of ParseLogsToDB being run. It cannot be called with any other commands.

7 Core Commands

7.1 check-new

Check if a newer version of ParseLogsToDB is available.

```
C:\[YOUR_FILE_PATH]>ParseLogsToDB check-new
New version check initiated.
Application is currently up-to-date.
```

7.2 deactivate

Deactivate the license on the current machine. Online deactivation removes the machine from the license server and deletes the local certificate; --local removes only the local certificate.

Options:

- **`--force`** – Skip confirmation prompt and force deactivation
- **`--local`**: Remove the license file from this machine without contacting the license server (air-gapped machines, or when the server rejects the deactivation)

```
C:\[YOUR_FILE_PATH]>ParseLogsToDB deactivate

==================================================
LICENSE DEACTIVATION WARNING
==================================================
This will permanently remove the license from this machine.
After deactivation:
  • All license files will be deleted
  • You will need to re-enter your license key to use this application
  • The license can be activated on a different machine
==================================================

Do you want to proceed with deactivation? [y/N]: y
Initiating license deactivation...
The deactivation was successful.
```

7.3 machine-code

NOTE: This feature is used for offline (air gapped) license activation, see license section below for more details.

Display the machine code and friendly name to send to your license administrator for offline activation. A valid license is not required to run this command.

```
C:\[YOUR_FILE_PATH]>parselogstodb machine-code

Provide the following values to your license administrator for offline
activation:

Machine code:  0123456789abcdef0123456789abcdef0123456789abcdef

Friendly name: SIGNAL-PC - Operator
```

The identity values above are examples; use the values printed on the target machine.

7.4 license

Activate a license key online or import an offline activation file. Provide exactly one of LICENSE_KEY or --file PATH. Use machine-code to obtain the machine identity needed for offline activation.

```
C:\[YOUR_FILE_PATH]>parselogstodb license ABCDE-01234-FGHIJ-56789
Adding license with key: *****-*****-*****-56789
License ACTIVE. Expiration: Valid forever!
```

Options:

- **`--file PATH`**: Import an activation file downloaded from the web activation form

```
C:\[YOUR_FILE_PATH]>parselogstodb license --file activation.skm
Importing activation file: activation.skm
License ACTIVE. Expiration: Valid forever!
```

The example uses a time-unlimited license; the activation file still needs refreshing every 183 days. If the status reports "not activated for this machine" after a hardware change, re-run license for the new machine identity. This needs a free seat on the key; contact the administrator to remove the stale machine if the activation limit is reached.

7.4.1 Offline (Air-Gapped) License Activation

Use this procedure when the machine running ParseLogsToDB cannot access the internet. The license administrator uses an internet-connected computer to prepare the activation file.

1. Run parselogstodb machine-code on the target machine. Send both the machine code and friendly name to your license administrator.
2. Administrator: From the Activation Form (https://app.cryptolens.io/Form/A/ZdUToafQ/3575), enter the ParseLogsToDB license key and the computer machine code. Download the activation file.
3. Transfer the downloaded file unchanged to the air gapped computer. Run parselogstodb license --file <file>, replacing <file> with the downloaded file path. Quote paths containing spaces.

4. Import the activation file within 183 days of issue. Obtain and import a new file every 183 days, including for a time-unlimited license. Refresh before the current file becomes stale to avoid interruption.
5. To remove the local license or move to another machine, run parselogstodb deactivate --local, then ask the administrator to remove the old machine in the dashboard. Local removal does not free the server seat.

7.5 process

Parses logs using information from the provided configuration JSON file into a database table.

```
C:\[YOUR_FILE_PATH]>ParseLogsToDB process "C:\Path With
Spaces\config.json"
License ACTIVE. Expiration: Valid forever!
Processed '1.1.1.1 Location Name Download Date.log': 111999 records added
```

8 Log File Requirements and Processing Behavior

- Place log files directly in the folder specified by logs_path. Subfolders are not scanned.
- Only files ending in .log or .txt are considered; extension matching is case-insensitive.
- The first whitespace-delimited part of each filename, before the extension, must be a valid IPv4 address.
- The detected log type must match logs_type. Files with an invalid filename, an unsupported type, or a type mismatch are skipped while processing continues with the remaining files.
- After successful database processing, each source file is moved into a processed subfolder beneath logs_path, or into the folder given by processed_path when that key is set.
- If the processed folder already contains a file with the same name, the existing processed copy is kept and the newly processed source file is removed after its records are handled. This also applies when several configurations share one processed_path: a file whose name already exists there is not archived again.

9 Database Behavior

ParseLogsToDB creates the configured table when it does not exist. If the table already exists, its schema must be compatible; the application does not migrate or alter an incompatible schema.

Records already present for the same location and time range are not inserted again. Data-log end-of-log markers are maintained independently for each IP address and program CRC/checksum; only markers for the same location and program may be updated or removed as newer log data is incorporated.

ElectroLogIXS timestamps are normalized to UTC using the log header's time-zone and daylight-saving settings, then stored without a time zone offset.

10 JSON File Format and Options

The configuration file must be of the JSON format (ending with a .json suffix) and follow the JSON format. The file must contain all the required key value pairs and may contain some or all of the optional key value pairs to supplement functionality. The keys and the value types are explained below with an example of the content of a JSON file in the Template Config File section.

10.1.1 Required Keys

The JSON file must contain these keys (case-sensitive) with a matching data value:

- **logs_path** – The path of a folder containing the log files which are to be parsed. The complete path should be provided including escape characters.
  - Example: **"C:\\Users\\name\\sub_folder_name\\test_logs_folder"**
- **logs_type** – One of the following allowable log types. Because different log types are placed in different database table configurations, only one log type may be processed during each run of the program. Separate configuration files may be set up and run at different times for different log types.
  - **"data"**
  - **"crossing_system_event"**
- **database_name** – The name of the database in which the table will be stored.
- **table_name** – The name of the table in which the data will be stored.
- **server_driver** – The server type and driver combination.
  - Example: **"mssql+pyodbc"**
- **server_path** – The path at which to access the SQL database server
  - Example: **"localhost\\SQLEXPRESS"**
- **server_query** – Any additional server connection parameters that may be required to successfully access the database server.
  - Example:

```
{
   "driver": "ODBC Driver 18 for SQL Server",
   "TrustServerCertificate": "Yes",
   "Trusted_Connection": "Yes"
}
```

10.1.2 Optional Keys

- **server_username** – Username for server login, if required. For PostgreSQL, this is the role name; when omitted, the Windows login name is used.
- **server_password** – Password for server login, if required. Omit it to use Windows integrated authentication (SQL Server Trusted_Connection or PostgreSQL SSPI).
- **server_port** – Database server port from 1 through 65535. Defaults to the driver's default (5432 for PostgreSQL).
- **processed_path** – Folder that processed log files are moved to. Created if it does not exist. Defaults to a "processed" subfolder beneath logs_path. Must not be the same folder as logs_path. A relative path is resolved against the folder the command is run from, the same as logs_path. Example: "C:\\Users\\name\\archive\\processed_logs”

10.1.3 Optional Email Keys

These keys are associated with the option to receive script execution reports via email after the execution of any `process` command.

10.1.3.1 Universal Email Keys

These keys are always required to access the email execution report functionality.

- `email_backend` – There are different methods of sending email, and this parameter must be set to determine which should be used. If this parameter is present and configured, an error message will be provided for any missing support parameters for the selected backend. Backend options are:
  - `smtp`
  - `mailtrap`
- `email_from_addr` – The address from which the email will originate.
- `email_to_addrs` – A list of email addresses to which the email will be sent.
- `email_reply_to_addr` – An optional key for a single Reply-To address for reports from any backend. It does not change the recipients.

10.1.3.2 SMTP Email Keys

These keys configure the SMTP backend. smtp_server is required; email_pw is required only when smtp_auth is true. The other keys are optional.

- `email_pw` – The password used to log in as email_from_addr. Required only when smtp_auth is true; omitted or ignored when smtp_auth is false.
- `smtp_server` – The address of the SMTP server that will send the email.
- `smtp_port` – An optional key to configure the SMTP port (defaults to 587 if unused).
- `smtp_starttls` – An optional boolean key that upgrades the connection with STARTTLS before sending (defaults to true). Set false only for a relay that accepts plaintext SMTP.
- `smtp_auth` – An optional boolean key that logs in using email_from_addr and email_pw (defaults to true). Set false for a relay that does not require authentication.
- `smtp_timeout` – An optional connection timeout in seconds, from 1 through 3600. Omitted or null uses 30 seconds. For a 120-second timeout, set smtp_timeout to 120. The timeout applies to each socket operation, not the total send duration.

STARTTLS and authentication are on by default and can be disabled with smtp_starttls / smtp_auth. Port 587 is the default. Disabling STARTTLS sends the message and any authentication credentials in clear text; it does not enable implicit TLS. Implicit TLS/SMTPS endpoints such as port 465 are not supported.

For an internal relay without TLS or authentication, set smtp_starttls and smtp_auth to false, typically use smtp_port: 25, and omit email_pw. Public SMTP providers generally require STARTTLS before accepting authentication.

10.1.3.3 Mailtrap Email Keys

These keys are required when using the Mailtrap backend.

- `mailtrap_token` – A personal Mailtrap token to utilize the service for the `email_from_addr` provided.

10.1.4 Template Config File (no email)

This is a template config file. Copy and fill in your values.

```
{
   "logs_path": "C:\\Path\\To\\Logs",
   "logs_type": "data",
   "database_name": "YourDatabaseName",
   "table_name": "YourTableName",
   "server_driver": "mssql+pyodbc",
   "server_path": "localhost\\SQLEXPRESS",
   "server_query": {
      "driver": "ODBC Driver 18 for SQL Server",
      "TrustServerCertificate": "Yes",
      "Trusted_Connection": "Yes"
   }
}
```

10.1.5 Template Config File (SMTP)

This is a template config file. Copy and fill in your values.

```
{
   "logs_path": "C:\\Path\\To\\Logs",
   "logs_type": "data",
   "database_name": "YourDatabaseName",
   "table_name": "YourTableName",
   "server_driver": "mssql+pyodbc",
   "server_path": "localhost\\SQLEXPRESS",
   "server_query": {
      "driver": "ODBC Driver 18 for SQL Server",
      "TrustServerCertificate": "Yes",
      "Trusted_Connection": "Yes"
   },

   "email_backend": "smtp",
   "email_from_addr": "sender@yourdomain.com",
   "email_to_addrs": ["recipient@example.com"],
   "email_reply_to_addr": "replies@example.com",
   "email_pw": "YOUR_EMAIL_PASSWORD",
   "smtp_server": "smtp.yourdomain.com",
   "smtp_port": 587,
   "smtp_starttls": true,
   "smtp_auth": true,
   "smtp_timeout": 30
}
```

10.1.6 Template Config File (Mailtrap)

This is a template config file. Copy and fill in your values.

```
{
   "logs_path": "C:\\Path\\To\\Logs",
   "logs_type": "data",
   "database_name": "YourDatabaseName",
   "table_name": "YourTableName",
   "server_driver": "mssql+pyodbc",
   "server_path": "localhost\\SQLEXPRESS",
   "server_query": {
      "driver": "ODBC Driver 18 for SQL Server",
      "TrustServerCertificate": "Yes",
      "Trusted_Connection": "Yes"
   },

   "_mailtrap_option": "Option 2: Mailtrap",
   "email_backend": "mailtrap",
   "email_from_addr": "sender@yourdomain.com",
   "email_to_addrs": ["recipient@example.com"],
   "email_reply_to_addr": "replies@example.com",
   "mailtrap_token": "YOUR_MAILTRAP_TOKEN"
}
```

10.1.7 JSON File Guidelines

- Keys are case-sensitive and must match exactly.
- Omit optional keys when they are not used. Empty strings are not valid for email_pw, smtp_server, or mailtrap_token, processed_path, and email_to_addrs cannot be an empty list.
- Extra or unexpected keys will be ignored.
- The logs_path value must identify an existing directory. The processed_path value, when supplied, may point to a folder that does not yet exist; it is created on first run and must differ from logs_path. The smtp_port value, when supplied, must be an integer from 1 through 65535, and all configured email addresses must be valid.
- smtp_timeout must be an integer from 1 through 3600 when supplied, or null to use the default. email_reply_to_addr must be a valid single email address when supplied. Malformed or out-of-range typed values stop config loading before logs are processed or reports sent; missing required email fields and sending failures are CLI warnings.

10.1.8 Protecting Credentials

Configuration files store database passwords, SMTP passwords, and Mailtrap tokens as plain text. Restrict access with Windows file permissions, and do not commit, email, or otherwise share populated configuration files.

Use Windows integrated authentication when available so server_username and server_password can be omitted.

11 Log Files and Execution Report Email

ParseLogsToDB keeps a running log of execution success or failure. This log is saved in the current working folder as debug.log and rotates when it reaches 2 MB. The file contains detailed execution information for each CLI run and includes error traces for troubleshooting. The developer may request this file when additional troubleshooting assistance is needed.

Because the user may be interested in receiving information about the success or failure of each run, the option is given to have the program send an Execution Report via email following the completion of the program execution. The email sent provides only information for the immediately preceding run of the program.

12 Troubleshooting

- Configuration errors: Confirm that the file has a .json extension, contains valid JSON, includes every required key, points logs_path to an existing directory, and does not set processed_path to the same folder as logs_path.
- ODBC or database errors: Confirm that the named ODBC driver is installed and that the server address, authentication settings, database name, table name, and database permissions are correct.
- No files processed: Confirm that supported .log or .txt files are directly inside logs_path and that each filename begins with a valid IPv4 address.
- Online license commands, version-check, or email errors: Confirm that the computer has network access and that required service credentials are valid. For an air-gapped installation, use the offline activation workflow.
- Unexpected failures: Review debug.log in the working folder for detailed error information.