ALGOTS
Obfuscator

TECHNICAL DOCUMENTATION

User guide

ALGOTS Obfuscator | English
Windows 10 / 11 (64-bit) • MQL5 / MetaEditor

VERSION 4.2.1 | MQL5 | Windows x64

© ALGOTS

ALGOTS / ENGLISH

Contents

01 Requirements ... 3

02 Installation ... 3

03 Activating your licence ... 4

04 Quick start: your first protected build ... 5

05 The desktop app in detail ... 5

06 Presets: Safe, Harden and Custom ... 6

07 Compile check with MetaEditor ... 7

08 Seeds and per-customer builds ... 7

09 Reversible builds and Restore ... 8

10 What you get in the output folder ... 8

11 Command line (CLI) ... 9

12 Licence management ... 11

13 Languages ... 12

14 Files and folders on your PC ... 12

15 Best practices ... 13

16 Troubleshooting ... 13

17 Uninstalling ... 14

18 Support ... 14

ALGOTS Obfuscator protects MQL5 source code. It takes a `.mq5` project with its `.mqh` include files and produces source code that still compiles in MetaEditor and is designed to preserve the program's behaviour, while making it much harder to read, copy and reuse. Compilation and functional testing remain necessary. Readable names become meaningless ones, comments and formatting go away, and in the stronger modes strings, constants and control flow get extra protection.

Everything runs locally on your PC. Your source code is never uploaded.

1. Requirements

Operating system	Windows 10 or Windows 11, 64-bit
Permissions	A normal user account. No administrator rights are needed
Internet	Required to activate the licence and for periodic licence checks
MetaTrader 5	Optional, but recommended: the 64-bit MetaEditor (`metaeditor64.exe`) is used for the automatic compile check
Disk space	About 100 MB

ALGOTS Obfuscator does not require Python or any other runtime: everything it needs is included.

2. Installation

1. Run `ALGOTS-Obfuscator-4.2.1-Setup.exe`.
2. Choose the setup language. The installer is available in the same 13 languages as the application.
3. On the **Select Additional Tasks** page choose:
   - **Create a desktop shortcut** — optional.
   - **Add the CLI to the user PATH (command: algots-obfuscator)** — lets you run the command-line version from any PowerShell or Command Prompt window by typing `algots-obfuscator`. It changes only your own user PATH and needs no administrator rights.
4. Complete the wizard.

The program is installed for the current user only, by default in `%LOCALAPPDATA%\Programs\ALGOTS Obfuscator`. Shortcuts are created in the Start menu: **ALGOTS Obfuscator** (desktop app), **ALGOTS Obfuscator CLI** (opens a terminal with the command-line help) and **Uninstall ALGOTS Obfuscator**.

Updating: run the newer setup. It replaces the program files and keeps your settings, licence and private build data.

> **Windows SmartScreen.** If Windows shows "Windows protected your PC" when you run the setup, verify the download source and check publisher/signature details and file integrity when available. Obtain the installer only from algots.net or your official purchase link. If you cannot verify it or the warning persists, contact support@algots.net before running it; do not bypass Windows protections.

> **Package integrity.** At every start the application checks that its own files have not been altered. If you see *"Package integrity check failed. Reinstall the complete application."*, a file was modified or removed (often by antivirus quarantine). Reinstall with the setup; if your antivirus removed a file, contact support@algots.net to investigate the quarantine and verify the installer and its files before taking further action. Do not disable protection.

3. Activating your licence

You receive your licence key by email after purchase. It is also available in your order page on Lemon Squeezy.

Desktop app

1. Open **ALGOTS Obfuscator**.
2. Click **License** in the top bar.
3. Paste your key in **License key** and click **Activate License**.

The **OBFUSCATE** and **RESTORE** buttons become available once the licence is active.

Command line

```
algots-obfuscator license activate
```

The key is requested through a hidden prompt. It is never passed as a command-line argument, so it does not end up in your shell history.

What is sent to the licence server. Only your licence key and the device identifier/name are sent, over HTTPS, to the Lemon Squeezy licence service. Your source code, project paths, seeds and build output never leave your PC. The application has no telemetry.

Each licence can be active on a limited number of devices at the same time. To move a licence to another PC, first deactivate it on the old one (see Licence management).

4. Quick start: your first protected build

1. **Main file** — Click **Browse...** next to *Main EA (.mq5)* and select the main `.mq5` file of your project (Expert Advisor, indicator or script). You only select the main file: its `#include` files are found and analysed automatically.
2. **Output folder** — Select an **empty** folder for the protected copy. Never use your source folder.
3. **Preset** — Leave **Safe** for your first build.
4. **Compile check** — Leave *Compile automatically with MetaEditor after obfuscation* enabled. Click **Auto** next to *MetaEditor* and *MQL5 root* if they were not detected automatically.
5. Click **OBFUSCATE**.

When the build finishes, the status bar shows the result of the compile check. Click **Open output** to see the protected files.

Before you deliver anything, test the protected program in MetaTrader 5 (for example in the Strategy Tester) exactly as you would test the original.

5. The desktop app in detail

Field / button	What it does
Main EA (.mq5)	The main `.mq5` source file of the project to protect (EA, indicator or script; this is the current English UI label)
Output folder	Where the protected project is written
MetaEditor	Path to `metaeditor64.exe`, used for the compile check. **Auto** detects it
MQL5 root	The terminal's `MQL5` data folder (the one containing `Include\`). Needed to resolve `#include <...>` files and for the compile check. **Auto** detects it
Preset	Safe, Harden or Custom (see below)
Options	The individual protections. Editable only with the Custom preset
Build / recovery seed	Optional seed for repeatable builds, or the seed of a reversible build. **Generate** creates a secure random seed

Field / button	What it does
Reversible build	Embeds an encrypted copy of the original project in the output so it can be restored later with the seed
Allow absolute `__PATH__`	See Troubleshooting. Off by default
Compile automatically with MetaEditor	Compiles the protected output before it is saved
OBFUSCATE	Starts the protected build
RESTORE	Restores the original project from a reversible build
Open output / Clear log	Opens the output folder / clears the log panel
License	Activation, status, online check and device deactivation
Language / Dark theme	Interface language and light/dark theme

Your choices (paths, preset, options, language, theme) are remembered for the next session. Seeds and licence keys are never saved in the settings.

6. Presets: Safe, Harden and Custom

ALGOTS Obfuscator follows one rule: **if a transformation cannot be proven safe for a given piece of code, that code is left unchanged.** A less aggressively obfuscated build that passes compilation and functional tests is preferable to one that introduces regressions.

Safe (default, recommended)

Renames the project's own symbols — functions, parameters, local and global variables, classes, structs, enums and members — whenever they can be resolved with certainty. MQL5 event handlers (`OnInit`, `OnTick`, `OnCalculate`, ...), `export` functions, built-in names and ambiguous constructs are preserved. Comments and formatting are kept. **Input names are kept**, so the settings dialog of your EA and existing `.set` files keep working.

Use Safe for everyday builds and always as your first build of a new project.

Harden

Everything in Safe, plus:

- **Flatten includes** — merges the project into a single protected source file;
- **Strip comments** and **Minify** — removes comments and formatting;
- **Protect strings** — string literals are not stored in readable form;
- **Encode constants** — integer constants are encoded;
- **Control-flow hardening** — adds opaque conditions that make the logic harder to follow.

Input names are still kept. Use Harden for the copies you deliver, after the same project has built and tested correctly with Safe.

Custom

Lets you switch each protection on or off: flatten includes, minify, strip comments, protect strings, encode constants, control-flow hardening, and rename inputs. String, constant and control-flow protection require *Flatten includes*.

> **Rename inputs** (Custom only) also renames your `input` variables. The program works, but the names shown in the EA properties dialog change and old `.set` files no longer match. Leave it off unless you want exactly that.

The command line also offers finer control in Custom mode, such as excluding specific names from renaming (`--protect-name`) or choosing which symbol categories to rename.

7. Compile check with MetaEditor

With the compile check enabled, ALGOTS Obfuscator compiles the protected project with MetaEditor **before** writing it to the output folder. The build is accepted only if MetaEditor reports zero errors and produces a fresh `.ex5` file. If compilation fails, the previous content of the output folder is left untouched.

- Only the 64-bit MetaEditor is selected automatically. If you have several MetaTrader installations, choose the one you want with **Browse...**.
- The **MQL5 root** must be the data folder of the same terminal (in MetaEditor: *File → Open Data Folder*, then the `MQL5` folder). You must be able to write to it: compilation uses a temporary folder inside it.

The compile check proves that the protected code builds. It does not replace testing the program's behaviour in MetaTrader.

8. Seeds and per-customer builds

Every build uses a seed that decides how names and protections are generated.

- **Empty seed field:** a new random seed is used, so every build is different.
- **Your own seed:** the same project, preset and seed always produce the same output. This is useful to reproduce a build you delivered.
- **Per-customer builds:** use a different seed for each customer. Each one receives a differently protected copy, which helps you tell copies apart.

Keep a record of which seed you used for which customer. Treat seeds as private.

9. Reversible builds and Restore

By default builds are **not reversible**: the output contains only the protected code and the original is not recoverable from it. Keep your original project in your own backup or version control.

If you enable **Reversible build**, the output also contains an encrypted, authenticated copy of the original project (AES-GCM, key derived from the seed). For reversible builds the seed must be a secure random value: click **Generate** (a 256-bit random seed is created). The seed is also saved in your private local vault (see Files and folders).

To restore:

1. Select the protected main `.mq5` file as the main file.
2. Select an empty output folder for the restored project.
3. Enter the seed used for that build.
4. Click **RESTORE**.

The restore is exact, byte for byte, including comments, formatting and file encoding. A wrong seed or a modified file is rejected before anything is written.

> **Without the seed a reversible build cannot be restored — not even by ALGOTS.** Do not lose it, and never send it to customers.

For copies you deliver to customers, a non-reversible build is the cleaner choice.

10. What you get in the output folder

- The protected project, ready to compile or deliver. With *Flatten includes* the main file may be named `YourEA_obfuscated.mq5`.
- `build_report.json` — a **public** report: status, counts of transformed and preserved symbols, warnings and the compile result. It contains no original names, no symbol map, no absolute paths and no seed.

Private data — the symbol map (original ↔ new names), the detailed diagnostic report and, for reversible builds, the seed — is stored in your local vault on your PC, not in the output folder. Use it for your own support work; never distribute it.

Output folder rules. The output folder must be empty or a folder previously created by ALGOTS Obfuscator. Each build replaces it completely and safely: files you removed from the project do not linger from earlier builds, and if anything fails the previous content is restored. Folders containing other data are refused to protect you from accidental overwrites.

11. Command line (CLI)

The CLI uses the same engine and the same licence as the desktop app. It is ideal for scripts, batch builds and CI pipelines.

Starting it

- If you selected *Add the CLI to the user PATH* during setup, open a **new** PowerShell or Command Prompt window and type `algots-obfuscator`.
- Otherwise use the **ALGOTS Obfuscator CLI** Start menu shortcut, or the full path: `"%LOCALAPPDATA%\Programs\ALGOTS Obfuscator\CLI\ALGOTSObfuscatorCLI.exe"`.

Commands

Command	Purpose
`obfuscate MAIN -o OUTPUT`	Builds a protected copy of the project
`restore MAIN -o OUTPUT`	Restores the original from a reversible build
`inspect MAIN`	Lists sources, includes, resources and missing dependencies. Writes nothing
`doctor [MAIN]`	Checks the environment: MetaEditor, MQL5 roots, components
`wizard`	Interactive guided build with a final confirmation
`license ...`	`activate`, `status`, `check`, `deactivate`, `forget-local`
`language [CODE]`	Lists or sets the interface language (shared with the desktop app)
`open-output PATH`	Opens an output folder in Explorer
`version`, `about`	Version and product information

Every command supports `--help`.

Examples

```
# Guided build
algots-obfuscator wizard

# Safe build with compile check
```

```
algots-obfuscator obfuscate "C:\Projects\MyEA\MyEA.mq5" -o "C:\Builds\MyEA" --preset safe --compile

# Harden build with an explicit MetaEditor and MQL5 root
algots-obfuscator obfuscate .\MyEA.mq5 -o .\build-harden --preset harden `
    --metaeditor "C:\Program Files\MetaTrader 5\metaeditor64.exe" `
    --mql5-root "C:\Users\me\AppData\Roaming\MetaQuotes\Terminal\<ID>\MQL5"

# Custom build: keep one name readable
algots-obfuscator obfuscate .\MyEA.mq5 -o .\build-custom --preset custom `
    --flatten-includes --minify --strip-comments --encrypt-strings --protect-name Calculate

# Repeatable per-customer build
algots-obfuscator obfuscate .\MyEA.mq5 -o .\customer-042 --preset harden --seed "customer-042-2026"

# Reversible build, then restore (the seed is asked with a hidden prompt)
algots-obfuscator obfuscate .\MyEA.mq5 -o .\reversible --recovery --show-recovery-seed
algots-obfuscator restore .\reversible\MyEA.mq5 -o .\restored --ask-seed

# Inspect a project and check the environment
algots-obfuscator inspect .\MyEA.mq5
algots-obfuscator doctor
```

Custom options (each also available as `--no-…`): `--strip-comments`, `--flatten-includes`, `--minify`, `--encrypt-strings`, `--encode-constants`, `--opaque-if`, `--rename-parameters`, `--rename-locals`, `--rename-globals`, `--rename-members`, `--rename-inputs`.

For scripts, `restore` also accepts `--seed-file PATH` (a UTF-8 text file containing the seed). Avoid `--seed` with a recovery seed on shared machines: command lines can be visible in shell history and process lists.

Automation

Option	Effect
`--json`	Prints exactly one JSON object on stdout, also for errors
`--quiet` / `-q`	Hides text summaries; warnings and errors still go to stderr
`--verbose` / `-v`	Shows engine logs and diagnostics
`--color auto|always|never`	Terminal colours
`--lang CODE`	Language for this command only, e.g. `--lang en`

JSON results always contain `schema_version`, `ok`, `command`, `exit_code` and either `result` or `error`. Warnings are in `result.warnings`.

Exit codes

Code	Meaning
0	Completed (warnings are possible)
1	Unexpected internal error

Code	Meaning
2	Invalid command-line arguments
3	Invalid input, configuration, file system or output folder
4	Invalid toolchain/environment path, or ambiguous MQL5 root
5	MetaEditor compile check failed or timed out
6	Restore refused (wrong seed, modified file, invalid target)
10	Licence activation required
11	Online licence check required
20	Licence rejected (expired, disabled or not valid for this product)
30	Licence service error — try again later
40	Cannot access the local licence data
50	Product configuration problem — contact support
51	Package integrity check failed — reinstall the application
130	Interrupted (Ctrl+C)

12. Licence management

In the desktop app click **License**; on the command line use `license`.

Action	Desktop app	CLI
Activate	**Activate License**	`license activate`
Show status	Shown in the dialog	`license status`
Force an online check	**Check online**	`license check`
Free this device	**Deactivate This Device**	`license deactivate`
Delete local licence data only	—	`license forget-local --yes`

- **Moving to a new PC:** deactivate on the old PC first, then activate on the new one.
- `forget-local` only deletes the licence data on this PC. It does **not** free the activation on the server; use `deactivate` for that.

- The licence is checked online periodically. If the licence server cannot be reached, you can keep working for a limited time after the last successful check (the dialog shows *Offline use allowed until*). After that, connect to the internet and click **Check online**.
- Changing the system clock backwards does not extend offline use.

For automated environments, `license activate --key-stdin` reads the key from standard input, and the `MQL5OBF_LICENSE_KEY` environment variable is also supported.

13. Languages

The desktop app, the CLI and the installer are available in English, Italian, Spanish, French, German, Portuguese, Turkish, Russian, Chinese (Simplified), Japanese, Korean, Arabic and Persian.

Choose the language in the app with **Language**. The CLI uses the same choice; otherwise it follows the Windows language, falling back to English.

```
algots-obfuscator language        # list languages
algots-obfuscator language it     # set Italian (also for the desktop app)
algots-obfuscator language auto   # follow Windows again
```

Command names, options, exit codes and JSON keys are never translated.

14. Files and folders on your PC

Location	Content
`%LOCALAPPDATA%\Programs\ALGOTS Obfuscator`	The application
`%APPDATA%\MQL5Obfuscator\settings.json`	Your preferences
`%LOCALAPPDATA%\MQL5Obfuscator\Licensing`	Licence state, encrypted with Windows DPAPI for your user account
`%LOCALAPPDATA%\MQL5Obfuscator\Builds`	Private build data: symbol maps and diagnostic reports
`%LOCALAPPDATA%\MQL5Obfuscator\Recovery`	Seeds and history of reversible builds

The `Builds` and `Recovery` folders are private. Back them up if you rely on them for support or restores, and never send them to customers.

15. Best practices

- Keep your original source in version control or a backup. ALGOTS Obfuscator is not a backup tool.
- Build every new project with **Safe** first, then move to **Harden**.
- Keep the compile check on, and test the protected program in MetaTrader before delivering it.
- Deliver non-reversible builds to customers.
- Use a different seed per customer and keep a private record of it.
- Never distribute the private vault, symbol maps or seeds.
- Obfuscation raises the cost of reading and reusing your code. It does not make reverse engineering impossible: combine it with proper licensing and contracts for valuable code.

16. Troubleshooting

The OBFUSCATE and RESTORE buttons are disabled. The licence is not active or needs an online check. Click **License**, then **Activate License** or **Check online**.

"MetaEditor was not found" / "select the terminal MQL5 root". Click **Auto**, or select `metaeditor-64.exe` and the terminal's `MQL5` folder manually. You can also turn off the automatic compile check.

The build fails with compile errors. First check that the original project compiles without errors in the same MetaEditor. Then try the Safe preset. If Safe compiles and Harden does not, contact support with the public `build_report.json` (never the private vault).

`#include <...>` files are not found. Set the MQL5 root of the terminal that contains those files.

The output folder is refused. Choose an empty folder, or a folder previously created by ALGOTS Obfuscator.

The build stops because of `__PATH__`. `__PATH__` would write the absolute path of your source file (which may contain your user name) into the delivered code. Remove it, or enable *Allow absolute `__PATH__`* only if you accept that.

"Package integrity check failed". Reinstall with the setup. If your antivirus quarantined a file, contact support@algots.net for a diagnosis; verify the package provenance and integrity before taking action. Do not disable security protections.

`algots-obfuscator` is not recognised. Open a new terminal after installing. If it still fails, run the setup again and select *Add the CLI to the user PATH*.

Licence: "activation limit reached". Deactivate the licence on a device you no longer use, or contact support.

17. Uninstalling

Use **Settings → Apps → Installed apps → ALGOTS Obfuscator → Uninstall**, or the **Uninstall ALGOTS Obfuscator** shortcut in the Start menu. Uninstalling also removes the CLI folder from your user PATH.

To free your licence for another PC, **deactivate it before uninstalling**. Your settings and private vault are kept; delete the `MQL5Obfuscator` folders listed in Files and folders if you want to remove them as well.

18. Support

- Email: support@algots.net
- Telegram: @ALGOTS_Support
- Web: https://algots.net/support

Please include the version (`algots-obfuscator version`), your Windows version and, for build problems, the public `build_report.json`. Never send seeds, symbol maps or the content of your private vault.

MetaTrader and MQL5 are trademarks of MetaQuotes Ltd. ALGOTS Obfuscator is an independent product and is not affiliated with MetaQuotes.