

ISPPROTECT

Malware Scanner End-User Manual

Your Server Detective.

Version 1.0.16
Projektfarm GmbH · ispprotect.com

Contents

1. Introduction 03

2. Installation 04

3. Licensing 06

4. Quick start 07

5. Command-line reference 08

6. How a scan runs 13

7. CMS, plugin and version detection 14

8. Signature feeds 15

9. Database scanning 16

10. Email reporting 17

11. Configuration wizard 18

12. Performance and tuning 19

13. Results, quarantine and whitelisting 20

14. Submitting samples 21

15. ISPProtect Console (malware-agent) 22

16. Data, state and environment 26

17. The ClamAV verdict service 28

18. Troubleshooting 29

19. Support and legal 30

1. Introduction

ISPProtect is a malware detection system for Linux web hosting environments. It scans web roots, applications and databases for malware, webshells, spam tools and vulnerable or outdated software — and reports what it finds before your users or providers do.

A scan combines three independent detection layers:

Level	Engine	What it catches
1	Binary signature engine (ClamAV verdict service)	Known malware, backdoors, trojans by binary signature
2	Pattern & heuristic engine	184 bundled pattern rules, obfuscation chains, suspicious code constructs
3	Bayesian classifier (optional)	Previously unseen malware, learned from verified malware and clean corpora

Every run writes a complete JSON report plus text artifacts for the phases that ran, and can deliver results by email. Confirmed hits can be quarantined and restored.

1.1 Architecture in brief

The scanner itself is a single self-contained Go binary (`malware-scan`). Binary signature scanning is provided by a separate small service, `clamav-ipc-service`, which links `libclamav`. The scanner starts it as a child process, asks it for verdicts over a local socket, and stops it when the scan ends. Where the service is absent — portable installs, or a deliberately removed package — the scanner reports the binary engine as unavailable and continues with pattern, Bayesian and database scanning.

2. Installation

2.1 From the APT repository (recommended)

Packages are signed and published per distribution. Pick the suite matching your system — installing another distribution's packages leaves you with a `clamav-ipc-service` binary built against a different glibc.

```
# prerequisites
sudo apt update
sudo apt install -y ca-certificates curl gnupg

# add and trust the repository key
sudo install -m 0755 -d /etc/apt/keyrings
curl -fsSL https://repo.ispprotect.com/public.key | \
    sudo gpg --batch --yes --dearmor -o /etc/apt/keyrings/ispprotect.gpg

# add the repository for your distribution
. /etc/os-release
echo "deb [signed-by=/etc/apt/keyrings/ispprotect.gpg] https://repo.ispprotect.com/ $VERSION_CODENAME main" |
\
    sudo tee /etc/apt/sources.list.d/malware-scan.list

# install
sudo apt update
sudo apt install malware-scan
```

Debian 11 (Bullseye) left LTS in August 2026 and its packages moved to archive.debian.org. If `apt` reports 404 errors while installing the dependencies, point the system's sources at the archive first.

On AlmaLinux, Rocky Linux, RHEL and other EL 8/9/10 (x86_64), with EPEL providing `clamav-lib`:

```
# EPEL provides clamav-lib
sudo dnf install -y epel-release

# Repository definition for this EL release (el8, el9 or el10)
sudo curl -fsSL -o /etc/yum.repos.d/ispprotect.repo \
    https://repo.ispprotect.com/rpm/ispprotect-el$(rpm -E %rhel).repo

# Import the signing key
sudo rpm --import https://repo.ispprotect.com/public.key

sudo dnf install malware-scan
```

Signing key. Packages and repository metadata are signed with the key

```
2FCF BFB8 26F4 962B A04A  EB1B 6022 3BD3 F8E6 4310
ISPProtect Repository Signing Key 2026 <info@ispprotect.com>, valid until 2029-10-10
```

Check it with `curl -fsSL https://repo.ispprotect.com/public.key | gpg --show-keys --with-fingerprint`.

Supported distributions: Debian 10 (legacy), 11, 12, 13, Ubuntu 22.04, 24.04 and EL 8, 9, 10. `malware-scan` pulls in `clamav-ipc-service` as a dependency; nothing has to be configured for it.

Updates arrive through the same channel — `apt upgrade` or `dnf upgrade` keeps the scanner current. Scanner installation and updates are intentionally handled **only** by signed system packages.

2.2 Portable package

A portable tarball runs without installation. Where `clamav-ipc-service` is not present, the binary engine is reported as unavailable and the scan continues with the remaining engines.

3. Licensing

License	Price	Notes
Trial	free	License key: `TRIAL`. No registration. Availability, expiry and scan limits are checked by the license server.
Annual	€82.68 / year (€6.89 / month)	One server, unlimited scans, BanDaemon IP-blocking add-on included.
Per-scan	€5 / €8 / €20 / €35 / €60	5 / 10 / 25 / 50 / 100 scans. Not server-bound, no time limit.

All prices exclude VAT; VAT applies to EU customers without a valid VAT ID. Current prices and offers: ispprotect.com.

Pass your license either directly or as a path to a key file:

```
malware-scan --license-key TRIAL --path /var/www          # trial
malware-scan --license-key /etc/malware-scan/license.key  # key file
```

To keep the key out of the process list, pass it through the environment instead. The scanner uses the first of `--license-key`, `MALWARE_SCAN_LICENSE_KEY`, the file named by `MALWARE_SCAN_LICENSE_FILE`, and the stored key, and removes both variables from its environment once read:

```
MALWARE_SCAN_LICENSE_FILE=/etc/malware-scan/license.key malware-scan --path /var/www
```

`malware-scan --license-status` shows the license; with `--json-out` it prints it as JSON (`schema_version`, `valid`, `status`, `message`, `trial`, `plan`, `expires_at`, `expires_date`, `days_remaining`, `scans_remaining`, `unlimited_scans`, `max_servers`, `key_masked`). `status` is `valid`, `expired`, `locked`, `invalid`, `no_scans` or `error` (no license server reachable); fields the server does not supply are `null`. The command exits 0 for a valid license and 3 otherwise.

Linking a server to the ISPProtect Console (section 15) needs a paid license: annual or per-scan packs. Trial licenses cannot be linked.

4. Quick start

```
# basic scan of one site
malware-scan --path /path/to/website

# deep scan including the database check
malware-scan --deep-check --db-scan --path /var/www

# scan and email the results
malware-scan --email-results admin@example.com --path /var/www
```

Without `--license-key`, the scanner prompts for one. A plain `--path` scan runs all malware engines, the CMS and plugin version checks and the ownership check (section 6).

5. Command-line reference

```
Usage: malware-scan --path <path> [options]
```

Option	Description
`--license-key <key>`	License key or path to a key file
`--path <path>`	Scan root; quoted globs are accepted when they resolve to one root
`--no-malware-scan`	Skip malware engines and all file-content work
`--no-version-scan`	Skip CMS and WordPress plugin version checks
`--no-plugin-version-scan`	Skip WordPress plugin versions only
`--version-scan <software>`	Check one supported CMS or product only
`--bayes-scan`	Enable the Bayesian (ML) classifier
`--no-image-scan`	Disable the additional binary image pass (it is on by default)
`--deep-check`	Deep inspection of suspicious files (slower, more thorough)
`--no-deep-check-network`	Deep check local includes only: no DNS lookups, no downloads
`--deep-check-timeout <s>`	Seconds per remote download in the deep check (default 5, maximum 300)
`--allow-single`	Permit scanning a single file
`--scan-level <1|2>`	1 = binary signatures and patterns (default), 2 = patterns only
`--regex-scan-all-files`	Apply the pattern rules to all file types
`--check-fps[=false]`	Suppress known false-positive matches (default on)
`--show-hits`	Print detections immediately
`--show-progress`	Show a progress bar for the file scan
`--incremental`	Reuse persistent clean/malware hash state between runs
`--concurrency <n>`	Concurrent file scans (default 5, maximum 1024)
`--exclude <glob>` / `--include <glob>`	Path filters; repeatable; `*` and `**` supported
`--exclude-from <file>` / `--include-from <file>`	Load filter patterns from a file; repeatable
`--max-age <days>`	Consider only files newer than this many days
`--ignore-chmod0`	Ignore files with mode 000
`--ignore <name>`	Suppress a signature by ID or name; repeatable (section 13)
`--fail-on <severity>`	Lowest severity that counts for exit code 1: `info`, `low`, `medium`, `high` or `critical` (default `medium`)
`--no-owner-check`	Skip the ownership and empty-script-file checks (on by default)

Option	Description
`--verify-checksums`	Compare WordPress core and plugin files with the official checksums (section 6)
`--time-check`	Check for suspicious file timestamps
`--blacklist-check`	Check the domains of the web server configuration against the DNS blacklists of `blacklist.conf` (section 10.1)
`--blacklist-config <file>`	DNS blacklist configuration (default: `blacklist.conf` in the configuration directory)
`--db-scan`	Scan MySQL/MariaDB databases
`--db-scan-config <file>`	Database configuration file
`--db-socket <path|auto>`	Connect through a Unix socket; no password needed
`--db-user <name>`	Database user (default: configured user, or the OS user)
`--db-scan-maxrows <n>`	Maximum rows read per table (default 1000000, 0 = all)
`--db-exclude <pattern>` / `--db-only <pattern>`	Exclude databases / scan only these databases; repeatable
`--db-exclude-table <pattern>` / `--db-only-table <pattern>`	Exclude tables / scan only these tables; repeatable
`--db-no-context`	Omit matched database values from reports
`--email-results <address>`	Send the report to this address; repeatable
`--email-config <file>`	SMTP configuration for report delivery
`--email-sender <address>`	Sender address of the report e-mail
`--email-hostname <name>`	Host name used in the report e-mail
`--email-empty-results`	Send the e-mail even when nothing was found
`--quarantine <dir>`	Move detected malware into this directory (turns quarantine mode on)
`--quarantine-mode`	Turn quarantine mode on (implied by `--quarantine`)
`--restore`	Restore files from the quarantine directory
`--results-dir <dir>`	Report directory (default `/var/log/malware-scan`)
`--json-file <file>`	Additionally write the complete JSON run report here
`--json-out`	Write the complete JSON run report to stdout (needs `--non-interactive`)
`--non-interactive`	Never prompt; required by `--json-out`
`--scan-id <id>`	Use this scan ID (1–64 of `A-Z a-z 0-9 . _ -`)
`--status-file <file>`	One-line status `PHASE;percent;hits;eta;scan_id`
`--progress-fd <n>`	JSON-lines progress events to inherited descriptor *n*
`--progress-file <file>`	JSON-lines progress events to a file
`--license-status`	Show the license and exit (JSON with `--json-out`)

Option	Description
`--list-signatures`	List the signatures of the scan engine, one per line, and exit; needs no license and no network
`--json`	With `--list-signatures`: print the full records as one JSON array; alone it is a configuration error
`--signature-dir <dir>`	Where downloaded signature feeds are stored
`--no-sig-update`	Never download third-party signature feeds
`--update-signatures`	Refresh the feeds even when they are current
`--sig-update-interval <h>`	Hours a downloaded feed stays current (default 24)
`--extended-signatures`	Also load the mail and spam sets (more false positives)
`--use-tmpfs`	Use `/dev/shm` for temporary files when available
`--use-tmp <dir>`	Use a writable temporary directory
`--whitelist <file>`	Add a file's SHA-256 to the whitelist and exit
`--whitelist-path <file>`	Whitelist file to use (default `~/.ispp_scan.whitelist`)
`--configure [db|email|all]`	Interactive configuration wizard, then exit
`--configure-dir <dir>`	Where the wizard writes its files
`--system-report`	Write a local `system_report.txt` and exit
`--cleanup [days]`	Remove reports older than this many days (7 without a value)
`--check-permissions [strict|lax]`	Run the permission check also without a file malware scan (`strict` without a value)
`--perm-check` / `--perm-check-strict`	Permission check in lax / strict mode, also without a file malware scan
`--no-check-permissions`	Switch the permission check off; it cannot be combined with the three options above
`--learn-ham <file>` / `--learn-spam <file>`	Train the Bayesian model with a benign / malicious file
`--reset-ip-address`	Reset the IP binding of the license
`--debug` / `--debug-regex-matches`	Diagnostic output / print exact regular-expression matches (may expose file content)
`--version`	Show the scanner version
`--timing`	Print total scan runtime
`--quiet`	Suppress terminal text only — reports are still written
`--report` / `--false-positive`	Submit a missed-detection / false-positive sample
`--confirm-file-upload`	Explicit consent required for any sample upload
`--email`	Optional contact address when submitting samples
`--help`	Show the complete option list

WordPress plugin vulnerability checking always runs. The ownership check runs together with the file walk unless `--no-owner-check` is given; it does not run for a single file. The permission check runs with every file malware scan in lax mode unless `--no-check-permissions` is given. Bayesian classification is off unless `--bayes-scan` is supplied; the enhanced image scan is on unless `--no-image-scan` is supplied.

`--no-malware-scan` still permits the independent CMS/plugin metadata phases, but does not initialize malware scanners, hash the file inventory, run deep checks, or populate the malware cache.

5.1 Exit codes

Code	Meaning
0	Scan complete, no finding at or above `--fail-on`
1	Scan complete, at least one finding at or above `--fail-on` (default `medium`)
2	Configuration or command-line error (including a reused `--scan-id`)
3	License invalid, expired, locked, out of scans, or not verifiable
4	I/O or runtime error: a check failed, the results directory could not be used, or the report could not be written
130	Interrupted by SIGINT or SIGTERM; a partial report is written

A finding is any detection of a file, remote resource, directory, database or blacklisted domain; each has a severity of `info`, `low`, `medium`, `high` or `critical`. Only findings at or above `--fail-on` (default `medium`) lead to exit code 1; weaker findings stay in the report. An interruption outranks findings, findings outrank failed checks, and failed checks give exit code 4; they are listed in the report's `errors`. Outdated software, vulnerable plugins and permission issues are reported but are not findings. A second SIGINT/SIGTERM ends the scanner at once, without a report.

5.2 Automation

For a control panel or script, combine `--non-interactive --json-out` with a fixed `--scan-id` and a progress stream:

```
malware-scan --path /var/www --non-interactive --scan-id panel-42 \
  --progress-fd 3 --json-out 3>progress.jsonl >report.json
```

With `--json-out` (or `--progress-fd 1`) stdout carries only JSON; every message for a person goes to stderr.

Report. `scan_report_<scan-id>.json` (always written to the results directory, and to stdout with `--json-out`) has `schema_version` 3. It starts with `schema_version`, `exit_code`, `complete`, `interrupted` and `scan_id`, followed by run data (`scanner_version`, `signature_set`, `scan_path`, `results_dir`, `started_at`, `completed_at`, `duration_seconds`, `modes` and `counts`). `finding_counts` holds `total`, `failing` (findings at or above `--fail-on`), `by_severity` and `by_type`. All findings are in one list, `findings`; every entry has an `id`, a `type` (`file`, `remote`, `directory`, `database` or `blacklist`), a `severity`, a `confidence` and a list of `detections`. A detection names the `signature_id` (`ISPP-…`), the `name`, the `engine` (`pattern`, `clamav`, `clamav-image`, `bayes`, `check` or `hash`) and, where known, the matched regions. Detections that were dropped (false-positive lists, `--ignore`, an official checksum match) are counted in `suppressed`, and `notes` explains limits of the run, such as a verification that was unavailable or a list that was shortened. The results of the maintenance checks follow in `software`, `wordpress_plugins`, `plugin_vulnerabilities` and `permission_issues`; `artifacts` lists the files of the run and `errors` the failed checks. Hits from the deep check's downloads have the type `remote` with `url` and `referenced_by`. All keys are snake_case. A report is at most 64 MiB: `errors` keeps at most 1,000 entries, and when the report is still too large the maintenance lists and then the findings are shortened, with a `sections_truncated` or `findings_truncated` note.

Scan ID. A reused `--scan-id` whose report already exists in the results directory is refused with exit 2.

Status file. `--status-file` holds a single line, replaced atomically: `PHASE;percent;hits;eta;scan_id`. Phases are `START`, `SCANNING`, `COMPLETE`, `INTERRUPTED` and `FAILED`; `COMPLETE` and `INTERRUPTED` appear only once the JSON report exists. `FAILED` means the scan ended without a report (the exit code says why); a scan that stops early, even through a second signal, still leaves `FAILED` or `INTERRUPTED` behind. The file has mode `0644`.

Progress stream. `--progress-fd` and `--progress-file` write one JSON object per line with `event`, `scan_id` and `time`. Events are `start` (with `scanner_version`, `scan_path` and the planned `stages`), `stage_start`, `progress` (with `stage` and `percent`, plus `files_scanned`, `files_total`, `hits` and `eta_seconds` for files, or `database`, `database_index`, `database_total`, `table`, `table_index`, `table_total`, `rows` and `hits` for databases), `stage_end` (with `status` `ok`, `failed` or `interrupted`, and `error`) and `end` (with `status`, `exit_code`, `findings` and `report`). Stages are `files`, `permissions`, `timestamps`, `blacklist`, `software`, `plugins`, `plugin_vulnerabilities`, `database` and `report`. The stages `permissions`, `timestamps` and `blacklist` are planned only when the matching check is switched on. Progress events come at most every 250 ms per stage. Every scan that emitted `start` also emits `end`, including early failures. A consumer that falls behind loses `progress` events, never the others, and never slows the scan down. The descriptor must be open for writing and is not passed on to programs the scanner runs. `--progress-file` refuses a link, pipe, directory or other non-regular entry, replaces an existing regular file with a new one of mode `0600` and refuses a name that ends in `/`.

6. How a scan runs

1. **File discovery** — recursive traversal of the scan root with include/exclude filtering and a persistent hash cache for incremental runs.
2. **Binary signature scan** — verdicts from the ClamAV IPC service.
3. **Pattern matching** — 184 bundled detection rules; regular expressions are compiled before scanning.
4. **CMS detection** — platform and version identification.
5. **Plugin check** — WordPress plugin versions against a vulnerability database.
6. **Deep inspection** — advanced content analysis (with `--deep-check`).
7. **Bayesian classification** — ML detection (with `--bayes-scan`).
8. **Trust lists** — SHA-256-based false-positive reduction: a file that is on the official list of known release files or in your whitelist is not reported.
9. **Reporting** — results compiled, delivered and archived.

Ownership check. Together with the file walk the scanner looks at who owns the files. It reports files owned by an unexpected account, that is neither the owner of the site, root nor the web server user (`ISPP-OWN-1`; `high` for script files, otherwise `medium`), files owned by the web server user (`www-data`, `apache`, `httpd`, `nginx` or `wwwrun`) below a differently owned directory (`ISPP-OWN-2`), setuid or setgid bits on regular files (`ISPP-OWN-4`), files whose owner has no account in a place where no site owner is known, for example directly below a directory owned by root (`ISPP-OWN-5`; below a site owner such a file is reported as `ISPP-OWN-1`) and directories with 50 or more empty script files (`ISPP-FS-1`). Results for files that are not scripts are combined per directory. `--no-owner-check` switches the check off; `--ignore` does not cover it.

Permission check. With every file malware scan the scanner also checks permissions, in lax mode; `--check-permissions`, `--perm-check` and `--perm-check-strict` select the mode and run the check even without a file malware scan, `--no-check-permissions` switches it off. It reports configuration files such as `wp-config.php` that others can access, and regular files and directories below the scan root, the root included, that anyone may write; symbolic links are skipped. The world-writable entries have severity `medium` and a suggested mode without the world-write bit, for example `1777` becomes `1775`. Each of the three rules lists at most 200 entries, the first ones in path order; when there are more, the report carries a `permission_issues_truncated` note with the number left out. `permissions_<scan-id>.txt` gives the total of each rule and the entries. The entries are `permission_issues`, not findings: they do not change the exit code and nothing is quarantined because of them.

Official checksums. With `--verify-checksums` the scanner compares WordPress core files and plugin files with the checksums published by wordpress.org. It requests `https://api.wordpress.org/core/checksums/1.0/` for the core and `https://downloads.wordpress.org/plugin-checksums/<slug>/<version>.json` for plugins; the requests carry the WordPress version and locale, or the plugin slug and version, and the user agent `ispprotect-malware-scan`. Lists are cached, and a scan makes at most 5,000 requests, four at a time. A file that matches its official checksum is not reported as malware; the dropped detections are listed in `suppressed` with the reason `official_checksum_match`. Differences are findings: a modified core file (`ISPP-VFY-1`, `high`), a file that does not belong to the core (`ISPP-VFY-2`, `high` for scripts, otherwise `low`), a modified plugin file (`ISPP-VFY-3`, `medium`) and a WordPress version that cannot be verified (`ISPP-VFY-4`, `low`). When a list cannot be fetched, the notes `verification_unavailable` or `verification_unit_ignored` say so. The option is off by default and is ignored with a warning for a single file.

Identical copies. A file with exactly the same content as a high-confidence content detection is reported with the detection `ISPP-HASH-1` and handled like the original.

7. CMS, plugin and version detection

The version scan identifies the following platforms and reports outdated installations together with their exact location — including forgotten copies left in old folders:

WordPress · Joomla! · Drupal · Magento · MediaWiki · Contao · PrestaShop · Shopware · Roundcube · phpMyAdmin · WoltLab Board (wbb) · MySQLDumper · TYPO3 · CMS Made Simple

WordPress plugin versions are always checked against a vulnerability database using concurrent lookups. Plugin reports contain both the plugin directory and the main-file path, so findings can be acted on immediately.

Disable individual phases with `--no-version-scan` (all version checks) or `--no-plugin-version-scan` (plugin checks only).

8. Signature feeds

Before every binary scan the scanner checks the third-party signature feeds it keeps on the machine and downloads anything missing or stale. Each feed lives in its own directory under `/var/lib/malware-scan/sigs/` with the attribution notice of the publishing project:

Feed	Content
Sanesecurity	Malware hashes, macro, URL and rogue databases
PhishTank	Phishing databases derived from PhishTank reports

Downloads are verified against the checksum published next to each database, written atomically, and skipped entirely while a feed is younger than `--sig-update-interval` (24 hours by default). Mirrors are tried in order; a failed feed never blocks a scan; a directory that is not writable (e.g. running as an unprivileged user) is skipped.

- `--no-sig-update` — switch feed downloads off completely
- `--update-signatures` — force a refresh
- `--extended-signatures` — additionally load the mail and spam databases (detects more, more false positives on hosted content)
- `--signature-dir <dir>` — alternative storage location

Signature sets that ship with the product instead of being downloaded are installed under `/var/lib/malware-scan/db/bundled/`, one directory per project, each with its licence notice. Two third-party sets are bundled: the Yara-Rules project (GPL-2.0) and the ReversingLabs YARA rules (MIT). The rules are modified for the scan engine (files flattened and renamed, rules the engine cannot evaluate removed, comments and licence notices kept). The proprietary ISPProtect signatures ship in the package as well but are first-party.

9. Database scanning

With `--db-scan`, the scanner examines MySQL/MariaDB content for injected malware and spam links.

Configuration in `/etc/malware-scan/db-config.conf` (or any file passed via `--db-scan-config`):

```
host = localhost
port = 3306
user = scanner
password = secret
databases = wordpress,joomla
```

The database user needs read access to the listed schemas. The shorthand `--db-scan=<file>` enables scanning with that specific configuration file.

The file uses `my.cnf` syntax, including `[client]` sections, quoted values and the `socket` and `protocol` keys. Like the `mysql` client, the scanner reaches a server on `localhost` through its Unix socket when it finds one (`/run/mysqld/mysqld.sock`, `/var/run/mysqld/mysqld.sock`, `/var/lib/mysql/mysql.sock`, `/run/mysql/mysql.sock`, `/tmp/mysql.sock`); `protocol = tcp` or `host = 127.0.0.1` forces TCP, and so does a `port` other than 3306 unless a `socket` is configured (a second instance on its own port).

No password is needed when MariaDB's `unix_socket` authentication is enabled, as it is for `root@localhost` on Debian and Ubuntu:

```
malware-scan --path /var/www --db-scan --db-socket auto
```

With `--db-socket <path|auto>` the configuration file becomes optional; the user is then the configured one, `--db-user`, or the operating-system user running the scan. `--db-user` naming a different user than the configuration file does not send that file's password.

What is reported. There is one finding for each database, table, column and signature, with the number of matching rows and at most three sample rows. A sample row holds the key of the row (`pk`), a SHA-256 of the row (`row_hash`), a piece of the surrounding text (`context`) and the matched text (`matched`). A finding has one of two classes: `stored_payload` for content in ordinary tables, and `logged_attack` for matches in log and statistics tables, which record requests instead of content the site uses; these are reported with severity `info` and without sample rows. Working tables of security plug-ins that store their own signatures and scan results are not read, and columns with credential-like names are neither matched nor quoted. With `--db-no-context` the `context` and `matched` values are left out; the key and `row_hash` remain.

Scope. `--db-scan-maxrows <n>` limits the rows read per table (default 1000000, `0` reads all rows). `--db-only`, `--db-exclude`, `--db-only-table` and `--db-exclude-table` take patterns and can be repeated.

Output. The terminal summary shows how many databases, tables and rows were examined, which databases and tables were skipped, and the number of detections. The details are written to `database_<scan-id>.txt` and appear in the report as findings of type `database`. The file is tab-separated with the columns `Database`, `Table`, `Column`, `Signature`, `Name`, `Severity`, `Class`, `Rows` (all matching rows), `Key` (`column=value,...`), `RowHash` and `Context`, one line per sample row. Database findings at or above `--fail-on` count for exit code 1 and are part of the e-mail report.

10. Email reporting

Deliver scan reports by email with `--email-results <address>` (repeatable for multiple recipients). SMTP settings live in `/etc/malware-scan/email-config.conf` or any file passed via `--email-config`:

```
smtp_host = smtp.example.com
smtp_port = 587
smtp_user = alerts@example.com
smtp_password = secret
from_address = scanner@example.com
```

The scanner has no built-in scheduler: it emails results at the end of a run, so combine it with cron for daily, weekly or monthly automated reports.

10.1 DNS blacklists

`--blacklist-check` takes the domain names from the Apache and Nginx configuration and looks them up in DNS blacklists (finding `ISPP-BL-1`, severity `medium`). The scanner ships no blacklist zones: the free public mirrors of Spamhaus and SURBL may not be used for commercial purposes or on shared hosting servers. You need your own licensed access, for example a Spamhaus Data Query Service (DQS) key or a SURBL data feed served by your own name server, and you enter its zones in `/etc/malware-scan/blacklist.conf` or in any file passed via `--blacklist-config`. The package installs `blacklist.conf.example` as a starting point:

```
[blacklist]
name = spamhaus-dbl
zone = <your-dqs-key>.dbl.dq.spamhaus.net
format = spamhaus-dbl

[blacklist]
name = surbl
zone = <your-surbl-zone>
format = surbl
```

Each `[blacklist]` entry has three keys:

- `name`: the label shown in reports, result files and e-mails; 1 to 64 letters, digits, `.`, `_` or `-`, unique within the file.
- `zone`: the DNS zone the domains are looked up in. A DQS zone contains your access key.
- `format`: how the answers are read, `spamhaus-dbl` for the Spamhaus DBL (also through DQS) or `surbl` for the SURBL multi list.

The file lists at most 16 entries; the zones are queried in the order given. Because a zone can contain an access key, the file must be readable by its owner only (`chmod 600`) and belong to root or to the account running the scan; the scanner refuses it otherwise. The zone is used for the DNS queries only: reports, result files, e-mails and messages show the `name` of the entry, and the finding names the configuration file in `config_file`. The search order is the same as for the database and email settings (section 11). Without a usable configuration the check does not run: the scan reports the error and ends with exit code 4 unless there are findings.

11. Configuration wizard

Database and email settings can be written interactively instead of by hand:

```
malware-scan --configure          # asks which part to configure
malware-scan --configure db       # database credentials only
malware-scan --configure email    # report delivery only
```

The wizard asks for each value (offering the currently configured value as default), then **checks the settings against the real server**: it connects to the database and lists the schemas, and opens the SMTP session and authenticates without sending anything. A failed check is reported and can be overridden, so a host can be prepared before the server it talks to exists.

Files are written with mode `0600` — they hold passwords — to `/etc/malware-scan/` when that is writable, otherwise to the user's configuration directory (`~/.config/malware-scan/`). Use `--configure-dir` to put them somewhere else.

Once written, scans pick both files up automatically; the scan prints which file it used. The search order is:

1. the path given on the command line (`--db-scan-config`, `--email-config`, `--blacklist-config`)
2. this account's configuration directory
3. `/etc/malware-scan/`
4. under `sudo`, the configuration directory of the invoking account
5. for databases only, the system's MySQL locations (`/etc/mysql/debian.cnf`, `/root/.my.cnf`, `~/.my.cnf`)

A file that exists but belongs to another account is skipped rather than failing the scan. When nothing is found, the error lists every path that was tried.

12. Performance and tuning

- **Concurrency** — `--concurrency <n>` controls parallel file scans (default 5, max 1024). Higher values speed up large trees at the cost of I/O load.
- **Incremental scans** — `--incremental` reuses persistent clean/malware hash state, so unchanged files are not re-examined.
- **Temporary files** — `--use-tmpfs` prefers `/dev/shm`; `--use-tmp <dir>` names an alternative directory.
- **Filtering** — `--exclude` / `--include` accept `*` and `**` globs and can be repeated; `--exclude-from` / `--include-from` read patterns from files.
- **Runtime** — `--timing` prints the total scan runtime for benchmarking.

13. Results, quarantine and whitelisting

Reports. Every scan writes `scan_report_<scan-id>.json` plus per-phase artifacts (including empty reports) under `/var/log/malware-scan` by default. Override the location with `--results-dir` or `MALWARE_SCAN_RESULTS_DIR`; `--json-file` additionally copies the full report anywhere you like. `--quiet` suppresses terminal text only, never reports. `--cleanup [days]` removes reports older than that many days (7 without a value); it never creates the results directory.

Result files. Each file name carries the scan ID. Depending on the checks that ran, the results directory holds `found_malware_<scan-id>.txt`, `database_<scan-id>.txt`, `permissions_<scan-id>.txt`, `timestamps_<scan-id>.txt`, `blacklists_<scan-id>.txt`, `software-versions_<scan-id>.txt`, `wordpress-plugins_<scan-id>.txt`, `plugin-vulnerabilities_<scan-id>.json` and, when files were quarantined, `quarantine-manifest-<scan-id>.json`. The directory has mode `0700` and the files mode `0600`. Text files are tab-separated, start with a header line and write control characters in escaped form. `found_malware_<scan-id>.txt` has one line for each file with a detection at or above `--fail-on` and the columns `FilePath`, `Signature`, `Name`, `Severity`, `DetectedAt` (RFC 3339) and `Scanner` (`binary`, `binary-image`, `pattern`, `bayes`, `check` or `hash`). For the columns of `database_<scan-id>.txt` see section 9.

Quarantine. `--quarantine <dir>` (or `--quarantine-mode`) moves detected local files out of the webroot; `--restore`, together with `--quarantine`, puts them back. Only files with a detection at or above `--fail-on` are moved, and only for a pattern, binary-signature or Bayesian detection, for the content checks `ISPP-CHK-3` and `ISPP-CHK-4` and for an identical copy (`ISPP-HASH-1`). Everything else is reported only and never moved: ownership and file-system findings (`ISPP-OWN-*`, `ISPP-FS-*`), checksum results (`ISPP-VFY-*`), the file-name and location checks `ISPP-CHK-1` and `ISPP-CHK-2`, timestamp and blacklist findings, database findings, and the version, plugin and permission results. A file that changed after the scan, or that is larger than 512 MiB, is left in place. A file counts as quarantined only when the original is gone; if it has further hard links, a warning is printed. A quarantine directory that contains the scan path is a configuration error (exit 2). The quarantine and results directories are left out of the scan, and an interrupted scan quarantines nothing. The e-mail report lists the files with detections at or above `--fail-on`.

Ignoring signatures. `--ignore <name>` takes a signature ID (for example `ISPP-…`) or a signature name and can be repeated. It suppresses pattern, binary-signature and Bayesian detections, the checks `ISPP-CHK-1` to `ISPP-CHK-4` and database findings of that signature. It does not cover ownership (`ISPP-OWN-*`, `ISPP-FS-*`) and checksum (`ISPP-VFY-*`) findings, timestamp and blacklist findings, or the permission, version and plugin results. Suppressed detections are counted in `suppressed`.

Whitelisting. Verified-clean files can be excluded from future verdicts: `--whitelist <file>` adds a file's SHA-256 to `~/.ispp_scan.whitelist` (or the file named by `--whitelist-path`) and exits. Whitelisting keeps legitimate files quiet across runs.

14. Submitting samples

Missed-malware and false-positive samples help improve detection:

```
malware-scan --report <file> --confirm-file-upload
malware-scan --false-positive <file> --confirm-file-upload --email you@example.com
```

Both send the complete selected file to ISPProtect and therefore require the explicit `--confirm-file-upload` flag. Files larger than 10,048,000 bytes and symlinks are rejected before any request is made.

15. ISPProtect Console (malware-agent)

15.1 What it is

`malware-agent` is an optional, separate package that connects a server to the **ISPProtect Console** at `https://license.ispprotect.com`. The Console shows scan reports of linked servers in one place and can start scans, review findings and carry out repairs on request. The scanner works fully without the agent; installing the scanner does not install or enable it.

The agent only makes outbound HTTPS connections. It opens no TCP or UDP listener and polls the Console for work.

Requirements:

- Linux with systemd, and root privileges (the production agent refuses to run as another user).
- A local filesystem that supports Linux file leases. For repairs and restores, `fs.lease-break-time` must be 5 seconds or more; the agent refuses safe repairs otherwise.
- An installed `malware-scan` package, `ca-certificates`, `systemd` and PHP (`php-cli`, or `/usr/bin/php` on EL). Node.js is recommended. The package manager installs these as dependencies; PHP and Node.js are used to syntax-check repair candidates, and repairs of a file type are rejected without its interpreter.
- An active paid license: annual or per-scan packs. Trial licenses cannot be linked. Expired or locked licenses keep report access and local restores; new scans and AI requests need an active paid license.

15.2 Installation

Install the package from the same repository as the scanner (section 2):

```
apt-get install malware-agent       # Debian / Ubuntu
dnf install malware-agent           # EL8 / EL9 / EL10
```

The package does not enroll the server or start the service; both are explicit steps for the administrator.

15.3 Linking a server

The first enrolment supplies the Console address, your scanner license file and the website directories the Console may work in:

```
sudo malware-agent --link \
  --portal https://license.ispprotect.com \
  --license-file /etc/malware-scan/license.key \
  --root /var/www
sudo systemctl enable --now malware-agent
```

Use the actual license file of the scanner; the key itself never appears on the command line. Repeat `--root` for every website directory (between 1 and 32). Roots must be existing, absolute directories without symlinks; `/` and the agent state directory are refused. The scanner executable defaults to `/usr/bin/malware-scan` and can be changed with `--scanner`.

The command prints a URL and a one-time code. Open the URL, or enter the code in the Console, and confirm that the displayed hostname is your server.

For every later login, or to add another account, run:

```
sudo malware-agent --link
```

Codes expire after ten minutes and work once. **A code grants full control of that server, so keep it private.** An account can collect several servers, but each server needs its own code, issued locally on that server.

A fresh code can move a server to another account. The previous account loses access, and its pending work and schedules are cancelled. The new owner inherits the full server history, including case messages and retained samples. File-upload consent and automatic policies are reset on transfer.

15.4 What the Console can do on the server

The agent accepts only typed jobs; the protocol has no way to send a shell command.

Job	Effect
Scan	Runs the installed scanner on an enrolled root, without interaction. Database scans use the scanner's own local database configuration; the Console cannot supply credentials or SQL.
File capture	Reads a selected file and sends its contents to the Console. Needs consent (section 15.5).
Repair	Replaces a reviewed file after the Console has shown the full before/after text. A root-private backup is made first.
Restore	Puts a backed-up file back. It never overwrites a newer version of the file.
Ignore	Adds a hash-specific (`sha256:`) whitelist entry that only affects scans started by the agent.

Restrictions: the agent only touches files below the enrolled roots and reads them confined to the root. It does not follow symlinks that lead out of a root, and refuses special files. A repair needs an unchanged file (same SHA-256), a regular file, at most 512 KiB, with no hard links, and refuses binary and empty files and files with setuid, setgid or capabilities. During the short commit, the containing directory is briefly restricted to root, so access to files in it can be interrupted for a moment. Busy files, unsupported filesystems and changed files fail safe: nothing is changed.

15.5 Privacy: what is sent

- **At enrolment:** the license key (read from the license file), the hostname, the configured roots, the agent version and a device token that the agent generated locally and later uses to authenticate. The agent version is sent on every poll.
- **Reports and findings:** results of scans started from the Console, and completed `scan_report_<scan-id>.json` reports of the scanner (section 15.6), contain metadata and findings such as file paths, hashes, signature names and software versions. They do not contain file contents. Database snippets and local artifact paths are removed from imported reports, and portal scans omit database match contents. A report carries at most the 10,000 most important findings; a note says how many were left out. While a scan runs, progress events are sent.
- **File contents:** only after an authorized Console user has granted upload consent. Until then, uploads are disabled. Selected files can contain private code or credentials. The Console stores samples encrypted and keeps them for its own retention period. Withdrawing consent cancels pending upload, AI and repair jobs.
- **AI features:** only for files you have opted in. Opted-in AI work is sent to OpenRouter and the model providers configured in the Console. AI judgements and repair proposals use points shared by the license. A proposal is not evidence that a site is safe.
- **Who controls the server:** anyone who holds a valid link code, or has access to the linked Console account, controls the server within the limits of section 15.4. Handle codes and accounts accordingly. Upload consent can be granted by a Console user signed in with a link code of this server.

15.6 Settings and files

Item	Default
Configuration	`/etc/malware-agent/config.json`, mode `0600`, owned by root
State, backups, recovery journals	`/var/lib/malware-agent` (mode `0700`)
Imported report directories	`/var/log/malware-scan`
Local backup retention	30 days

The file contains a secret device token; never share it or paste its content into a ticket. Keys a user may change:

Key	Meaning
`report_dirs`	Trusted directories from which scanner reports (`scan_report_<scan-id>.json`) are imported automatically. Set at first enrolment with repeated `--report-dir`; `"report_dirs": []` disables report import. Directories must be outside the website roots. Reports of another schema version are not imported; run a new scan with the current scanner.
`backup_days`	Days to keep backups, 1 to 365. Backups needed for recovery are kept longer.
`roots`	The enrolled website roots. The configuration also records the identity (device and inode) of each root, so do not edit this key by hand. To change roots, stop the service and create a new configuration with the enrolment command; keep the old configuration and backups until recovery is no longer needed.

`--state-dir` and `--config` at enrolment select other locations. Restart the service after you edit the configuration.

15.7 Disconnecting a server

1. In the Console, use **Disconnect server**. This revokes the server's device credential.
2. On the server, run `sudo systemctl disable --now malware-agent`.

There is no local unlink command. To connect again, create a new private configuration with the enrolment command in section 15.3. Removing the package stops the service but keeps the configuration, backups and recovery journals.

15.8 Upgrading from malware-daemon

`malware-agent` replaces the earlier `malware-daemon` package; the two cannot be installed together. Stop and disable the old service first:

```
sudo systemctl disable --now malware-daemon.service
# install the malware-agent package with your package manager
sudo systemctl enable --now malware-agent.service
```

Without `--config`, the agent uses `/etc/malware-agent/config.json` and otherwise reuses `/etc/malware-daemon/config.json`. An invalid configuration is an error and does not trigger a fallback. Existing enrolment, state directory, backups and recovery journals stay in place; do not move or delete them. New enrolments use `/etc/malware-agent/config.json` and `/var/lib/malware-agent`.

15.9 Service operations

```
systemctl status malware-agent
journalctl -u malware-agent --since today
```

`malware-agent --version` prints the agent version. If recovery reports a changed inode, stop the service and keep `directory-guard.json`, `file-metadata.json` and `backups/` in the state directory for inspection; do not delete a journal to silence the error.

16. Data, state and environment

Purpose	Default	Override
Scan reports	`/var/log/malware-scan`	`--results-dir`, `MALWARE_SCAN_RESULTS_DIR`
Signature feeds	`/var/lib/malware-scan/sigs/`	`--signature-dir`
Bundled signature sets	`/var/lib/malware-scan/db/bundled/`	—
Data files (signature DB, whitelist, classifier)	`/var/lib/malware-scan`, `/usr/share/malware-scan`, `/etc/malware-scan`, then relative to the binary and the working directory	`MALWARE_SCAN_DATA_DIR` (colon-separated)
Writable state (e.g. `classifier.model.user`)	`MALWARE_SCAN_STATE_DIR`, then `/var/lib/malware-scan`, XDG locations, user config directory, working directory	`MALWARE_SCAN_STATE_DIR`
Incremental-scan state (`--incremental`)	`cache/<sha1 of the absolute scan root>.state` in the first writable state directory above	`MALWARE_SCAN_STATE_DIR`
License key	stored key	`MALWARE_SCAN_LICENSE_KEY`, `MALWARE_SCAN_LICENSE_FILE`
Verdict service discovery	next to the scanner binary, `/usr/libexec/malware-scan`, `/usr/lib/malware-scan`, `PATH`	`MALWARE_SCAN_CLAMAV_IPC` (explicit path)
Agent configuration (`malware-agent`)	`/etc/malware-agent/config.json` (mode `0600`)	`--config` at enrolment
Agent state, backups, recovery journals	`/var/lib/malware-agent`	`--state-dir` at enrolment
Agent report import	`/var/log/malware-scan`	`report_dirs` in the agent configuration (section 15.6)
External verdict service	child process is started per scan	`MALWARE_SCAN_CLAMAV_IPC_SOCKET` (use an already running service)

Packaged installs ship signatures in `/var/lib/malware-scan/`; portable installs keep using the local `db/` and `data/` folders. Environment variables override all of it — useful when running multiple scanner instances with different signature sets.

The incremental state is keyed by the absolute, cleaned scan root, so `--path .`, `--path /var/www/site/` and `--path /var/www/site` share one state file.

16.1 Running in a sandbox

Under systemd hardening, in a container, or with a seccomp/AppArmor profile, the scanner needs:

- **Read access** to the scan root and to `/var/lib/malware-scan`, `/usr/share/malware-scan` and `/etc/malware-scan`.

- **Write access** to the results directory (`/var/log/malware-scan`), the state directory (with `--incremental`, training or feed updates; point `MALWARE_SCAN_STATE_DIR` at a writable path), the quarantine directory when used, and a temporary directory (`TMPDIR`, `--use-tmp` or `--use-tmpfs`).
- **The verdict service.** The scanner starts `clamav-ipc-service` as a child process, so `fork` / `exec` must be allowed and the binary in `/usr/libexec/malware-scan` or `/usr/lib/malware-scan` must be executable. The service reads the signature directories (`/var/lib/malware-scan/db`, `/var/lib/malware-scan/sigs`) and every file it is asked about, and listens on a Unix socket in a private directory under `TMPDIR`, `XDG_RUNTIME_DIR`, `/tmp` or `/run` (the first that works; the path must stay under 100 bytes). Scanner and service must see the same temporary directory. The service is killed when the scanner exits. Alternatively, run one service outside the sandbox and set `MALWARE_SCAN_CLAMAV_IPC_SOCKET` to its socket.
- **Network (HTTPS):** `license.ispprotect.com` and `license2.ispprotect.com` always; `api.wordpress.org` for plugin versions and, with `--verify-checksums`, the core checksums, and `downloads.wordpress.org` for the plugin checksums; the signature feed mirrors unless `--no-sig-update`; and, with `--deep-check`, the URLs found in scanned files unless `--no-deep-check-network`. `--blacklist-check` needs DNS only: it looks the domains up in the zones of `blacklist.conf` through the system resolver. Database scans need the database socket or TCP port.
- **Network for `malware-agent` (section 15):** outbound HTTPS (port 443) to the ISPProtect Console at `license.ispprotect.com`; the agent opens no listener.

17. The ClamAV verdict service

`clamav-ipc-service` is a separate GPL-licensed program that links `libclamav`. The Debian packages install it as a dependency of `malware-scan`. It can also be installed and used on its own:

```
apt-get install clamav-ipc-service
```

Where it is absent, the scanner reports that the binary engine is unavailable and continues with pattern, Bayesian and database scanning. The service is started per scan and stopped afterwards; point `MALWARE_SCAN_CLAMAV_IPC_SOCKET` at a running instance to share one service between scans.

18. Troubleshooting

"Binary engine unavailable" — `clamav-ipc-service` is not installed or not findable. Install the package or set `MALWARE_SCAN_CLAMAV_IPC`. Scans continue with the remaining engines.

Configuration file not found — run `malware-scan --configure`, or check the search order in section 11; the error lists every location tried.

Too many false positives on hosted mail/spam content — avoid `--extended-signatures`, and whitelist verified-clean files (section 13).

Scan of a single file fails — single-file scans need `--allow-single`.

Unexplained load between scans — feeds refresh at most every `--sig-update-interval` hours; use `--no-sig-update` on bandwidth-limited hosts. Consider `--incremental` and `--use-tmpfs` for large trees.

Permission problems as an unprivileged user — feed directories that are not writable are skipped automatically; run reports land in a user-writable `--results-dir`.

Agent: "enrollment is incomplete; run malware-agent --link --config ..." — the server was never linked. Run `sudo malware-agent --link` as described in section 15.3, then start the service.

Agent: "agent configuration must be a regular file readable only by its owner (0600)" — fix the permissions: `chmod 600 /etc/malware-agent/config.json`, owned by root.

Agent: "portal URL must use HTTPS" — pass `--portal https://license.ispprotect.com`; only HTTPS addresses are accepted, without path or query.

Agent: "production agent must run as root" — run the command with `sudo`, and run the service as root.

Agent: "managed root identity changed: <path>; update the local configuration after verifying the directory" — the directory at that path is not the one that was enrolled (for example after a storage migration). Verify it, then stop the service and create a new configuration with the enrolment command (section 15.6).

Agent: "portal returned <status>: <message>" — the Console rejected the request, for example because the license is a trial or inactive, or the credential was revoked. A link code that has expired (ten minutes) or was already used is also rejected: run `sudo malware-agent --link` again for a fresh code.

Agent: "safe repairs require fs.lease-break-time of at least 5 seconds" — raise the `fs.lease-break-time` kernel setting to 5 or more.

19. Support and legal

- Documentation installs to `/usr/share/malware-scan/`
- Support: support@ispprotect.com · ispprotect.com
- Provider details: Imprint / Impressum
- Malware and false-positive samples: `--report` / `--false-positive` (section 14)

ISPProtect is proprietary software of Projektfarm GmbH. Third-party signature sets remain under their own licences and attributions (Sanesecurity, PhishTank, Yara-Rules, ReversingLabs); `clamav-ipc-service` is distributed under the GPL in its own package and repository.