

# Malware Feeds

# Malware Feeds

Admin path: **Content Checks > Malware Feeds** (`view_malware_feeds.cfm`,  
`inc/get_malware_feeds_settings.cfm`, `inc/malware_feeds_save_global.cfm`,  
`inc/malware_feeds_add_feed.cfm`, `inc/malware_feeds_edit_feed.cfm`,  
`inc/malware_feeds_delete_feed.cfm`, `inc/malware_feeds_toggle_feed.cfm`,  
`inc/malware_feeds_save_urls.cfm`, `inc/generate_malware_feeds_configuration.cfm`).

This page manages the third-party ClamAV signature feeds that supplement the stock `freshclam` definitions on [Antivirus Settings](#). The feed manager is [Fangfrisch](#), a small Python tool that handles per-feed authentication, cadence control, integrity verification, and post-download deployment. Hermes ships ten built-in feed definitions (free and commercial), exposes a custom-feed form for additional sources, and a per-feed URL editor for signature file selection. Refresh runs as an [Ofelia](#) job inside `hermes_mail_filter`.

This page replaced an earlier `view_antivirus_signature_feeds.cfm` page (orphan cleanup tracked as issue #257); any sidebar bookmark or external link pointing at the old page should be updated.

## How feeds reach ClamAV

```
+-----+
| hermes-fangfrisch-refresh (Ofelia job) |
|   inside hermes_mail_filter           |
|   schedule: @every <refresh_interval> |
+-----+
|
| v
+-----+
| /usr/bin/fangfrisch refresh           |
|   reads /etc/fangfrisch/fangfrisch.conf |
|   iterates enabled feeds               |
|   skips feeds whose own interval has not |
```

```

| elapsed |
+-----+
|
| v
+-----+
| Per-feed download |
| auth via API key / customer_id / |
| serial_key when required |
| integrity check (sha256, md5, off) |
| -> /var/lib/fangfrisch/signatures/ |
+-----+
|
| v
+-----+
| on_update_exec=/usr/local/bin/setup- |
| clamav-sigs (post-update hook) |
| validates each file with `clamscan` |
| copies valid files to /var/lib/clamav/ |
| signals clamd to reload |
+-----+

```

The page emits `/etc/fangfrisch/fangfrisch.conf` (an INI file) on every save. Fangfrisch itself is invoked on a fixed Ofelia schedule; the schedule is regenerated from `ofelia_jobs.schedule` and reflects the Global Settings > Refresh Interval picker.

## Container and tool placement

Component	Detail
Container	<code>hermes_mail_filter</code> (IPv4 <code>.105</code> , same container as ClamAV, Amavis, SpamAssassin)
Feed manager	<code>fangfrisch</code> (Python, third-party ClamAV signature aggregator)
INI config	<code>/etc/fangfrisch/fangfrisch.conf</code> (bind-mounted, owned <code>root:clamav</code> , mode <code>640</code> )
State DB	<code>sqlite:///var/lib/fangfrisch/db.sqlite</code> (per-feed last-refresh, integrity hashes)
Download dir	<code>/var/lib/fangfrisch/signatures/</code> (raw downloaded files)
Deploy dir	<code>/var/lib/clamav/</code> (validated files, ClamAV signature store)

Component	Detail
Post-update hook	<code>/usr/local/bin/setup-clamav-sigs</code> (validates with <code>clamscan</code> , copies to deploy dir, signals reload)
Scheduler	Ofelia job <code>hermes-fangfrisch-refresh</code> row in <code>ofelia_jobs</code>
Default cadence	<code>@every 10m</code> (Fangfrisch then honors per-feed <code>interval =</code> to decide what to actually re-fetch)

## Global Settings card

Four controls write to `parameters2 WHERE module = 'malware_feeds'`. The first three substitute into the `[DEFAULT]` section of `fangfrisch.conf` on every save; the fourth updates the Ofelia row that schedules the refresh job.

Field	Storage	INI / scheduler effect	Notes
Log Level	<code>parameters2.value2 (log_level)</code>	<code>[DEFAULT] log_level = ...</code>	<code>debug,info,warning,error,fatal</code> ; logs go to <code>docker logs hermes_mail_filter</code>
Default Max Size	<code>parameters2.value2 (max_size)</code>	<code>[DEFAULT] max_size = ...</code>	Per-file cap. Regex anchors a number followed by <code>KB</code> , <code>MB</code> , <code>M</code> , or <code>B</code> (e.g. <code>5MB</code> , <code>10M</code> , <code>250KB</code> ). Inherited by feeds that don't set their own
Update Timeout (sec)	<code>parameters2.value2 (on_update_timeout)</code>	<code>[DEFAULT] on_update_timeout = ...</code>	Bounded 1-300. Caps how long <code>setup-clamav-sigs</code> is allowed to run
Refresh Interval	<code>parameters2.value2 (refresh_interval)</code> AND <code>ofelia_jobs.schedule</code>	Ofelia <code>@every &lt;interval&gt;</code>	Allowed values: <code>5m, 10m, 15m, 30m, 1h, 2h, 4h</code> . Fangfrisch's own per-feed <code>interval =</code> still gates whether each feed actually re-downloads on a given run

The post-update hook path is hard-coded to `/usr/local/bin/setup-clamav-sigs` and shown read-only beneath the form as `[DEFAULT] on_update_exec`. The hook lives inside the `hermes_mail_filter` image and validates each downloaded file with `clamscan` before copying it to `/var/lib/clamav/`; a file that fails validation is left in the Fangfrisch download dir and not deployed.

## Malware Feeds card

Rows from `malware_feeds_config` populate a DataTable; per-row form posts toggle, edit, manage URLs, and (custom feeds only) delete. The schema:

Column	Role
<code>id</code>	Surrogate key
<code>section_name</code>	INI section header, <code>[&lt;section_name&gt;]</code> . Lowercase alphanumeric + underscore ( <code>^[a-z0-9_]+\$</code> ). Cannot change after creation. Unique.
<code>display_name</code>	Card label, free text
<code>enabled</code>	<code>tinyint(3)</code> , 0/1. Sliders here flip this. <code>enabled = yes/no</code> line in INI
<code>is_builtin</code>	<code>tinyint(3)</code> , 0/1. Built-in rows cannot be deleted (the Delete action button is suppressed in the UI and the delete handler refuses)
<code>prefix</code>	<code>\${prefix}</code> interpolation source for URL entries. Optional
<code>interval_value</code>	Per-feed cadence (e.g. <code>1h</code> , <code>4h</code> , <code>1d</code> ); blank = inherit <code>@every &lt;refresh_interval&gt;</code>
<code>max_size</code>	Per-feed cap; blank = inherit Global Default Max Size
<code>integrity_check</code>	<code>sha256</code> , <code>md5</code> , <code>disabled</code> , or NULL (default <code>sha256</code> )
<code>api_key_1_name</code> / <code>api_key_1_value</code>	Optional auth key (e.g. <code>customer_id</code> , <code>receipt</code> ). Value stored AES-encrypted with key <code>/opt/hermes/keys/hermes.key</code>
<code>api_key_2_name</code> / <code>api_key_2_value</code>	Second auth key (e.g. MalwarePatrol's <code>product</code> ). Same encryption
<code>description</code>	Free text
<code>sort_order</code>	Display order; custom-add inserts at 100

## Built-in feed catalog (factory rows)

Feed	Type	Default state	Auth	Notes
SaneSecurity	Free	Enabled	None	Broad zero-day coverage; mirror <code>https://ftp.swin.edu.au/sanesecurity/</code>
URLhaus	Free	Enabled	None	Malicious URL signatures from abuse.ch
MalwarePatrol	Commercial	Enabled	<code>receipt</code> , <code>product</code> IDs	Configure both keys via Edit; subscription IDs are documented in the in-card help

Feed	Type	Default state	Auth	Notes
MalwareExpert	Commercial	Enabled	<code>serial_key</code>	URL template embeds the serial in the path
SecuriteInfo	Commercial	Enabled	<code>customer_id</code>	Free tier available; paid tier unlocks extra URLs
TwinWave	Free	Enabled	None	Public GitHub-hosted signatures
ClamPunch	Free	Enabled	None	Heuristic family signatures
RFXN	Free	Enabled	None	R-fx Networks Linux Malware Detect signatures
InterServer	Free	Enabled	None	Hash + URL signatures
Ditekshen	Free	Enabled	None	YARA/ClamAV detection rules

A commercial feed is "enabled" only in the sense that its row is marked `enabled = 1`; without API keys the feed is configured but will not actually fetch (the in-card help describes the per-vendor key requirements and the table icon shows a yellow warning triangle on commercial rows missing keys).

## Add Custom Feed modal

Free-form add for any feed source not in the built-in catalog. Validation:

Field	Rule
Section Name	<code>^[a-z0-9_]+\$</code> , required, must not already exist
Display Name	Required
URL Prefix	Optional, becomes the <code>prefix =</code> line and the substitution source for <code>\${prefix}</code> in URL entries
Update Interval	Optional, number followed by <code>m</code> (minutes), <code>h</code> (hours), or <code>d</code> (days). Examples: <code>10m</code> , <code>1h</code> , <code>1d</code>
Max Size	Optional, number followed by <code>KB</code> , <code>MB</code> , <code>M</code> , or <code>B</code> . Examples: <code>5MB</code> , <code>250KB</code>
Integrity Check	Dropdown: default (sha256), sha256, md5, disabled
Description	Optional free text

A new custom feed is inserted with `enabled = 0` and `is_builtin = 0`; the admin then opens the URL manager to register at least one URL before turning the row on.

# Manage URLs modal (per-feed)

Rows from `malware_feed_urls` keyed by `feed_id`. Each URL becomes a line in the corresponding `[<section_name>]` block of `fangfrisch.conf`:

```
url_<url_key> = <url_value>
filename_<url_key> = <filename_override>  ## only when filename_override set
```

When a URL is toggled off, the `url_` prefix is replaced with `!url_` to inactivate the line without losing the configuration. `${prefix}` in the URL value is expanded against the feed's `prefix =` at fetch time.

Field	Rule
Name ( <code>url_key</code> )	<code>^[a-z0-9_]+\$</code> , must be unique within the feed ( <code>UNIQUE KEY uq_feed_url(feed_id, url_key)</code> )
Download URL ( <code>url_value</code> )	Full URL, or <code>\${prefix}&lt;path&gt;</code> shorthand when the feed has a prefix
Save As ( <code>filename_override</code> )	Optional. Renames the downloaded file locally; useful when the source filename is too generic
Toggle	Per-URL on/off. Disabled URLs are skipped without being deleted

Built-in feeds may have URLs that Fangfrisch maintains internally — the in-modal note explains that an empty URL table for a built-in feed means it is using its packaged defaults, not that it is broken.

## Save flow

- View page submits `action= save_global | add_feed | edit_feed | delete_feed | toggle_feed | url_action`
- `malware_feeds_*.cfm` validates and UPDATES/INSERTS/DELETES the row(s)
- `generate_malware_feeds_configuration.cfm` runs on EVERY action:
  - `SELECT module='malware_feeds' rows from parameters2 -> globalSettings`
  - `SELECT malware_feeds_config -> all feed rows`
  - `SELECT malware_feed_urls -> all URLs grouped by feed_id`

- d. Build [DEFAULT] section + one [<section\_name>] block per feed
  - e. Decrypt api\_key\_\*\_value with AES + /opt/hermes/keys/hermes.key  
(key emitted as `<api\_key\_\*\_name> = <plain>`)
  - f. Write temp file -> /opt/hermes/tmp/<trans>\_fangfrisch.conf
  - g. dos2unix (tolerated if missing)
  - h. cffile write -> /etc/fangfrisch/fangfrisch.conf
  - i. docker exec hermes\_mail\_filter chown root:clamav + chmod 640  
(tolerated if container is down)
  - j. cfinclude ofelia\_generate\_config.cfm  
(rewrites /etc/ofelia/config.ini if any schedule changed)
4. cflocation back to view\_malware\_feeds.cfm
  5. session.m + session.alerttype + session.alertmsg drives the alert banner

Every UI action -- including a single-row enable/disable toggle -- runs the full INI regen, ownership fix, and Ofelia config regen. There is no incremental write path; the INI is always rendered from the current database state. This means manual edits to `/etc/fangfrisch/fangfrisch.conf` are lost on the next save -- store all configuration in the database.

## API key encryption

The `api_key_1_value` and `api_key_2_value` columns store AES-Base64 ciphertext using the key in `/opt/hermes/keys/hermes.key`. The edit modal shows a masked preview (20 asterisks + last 4 chars of the plaintext) for visual confirmation without exposing the full key. Decryption happens only in `generate_malware_feeds_configuration.cfm` at the moment the INI is rendered; a decryption failure replaces the key line with a commented `## <name> = [decryption error]` marker rather than aborting the save.

The encryption key file is mounted into `hermes_commandbox` only; neither `hermes_mail_filter` nor any other service reads it. This keeps the plaintext key out of the running config on disk for as short a window as possible (write -> chmod 640 root:clamav -> next Fangfrisch run reads -> file remains until next save replaces it).

## Manual refresh

The Ofelia job runs on schedule, but the same command can be invoked manually from a host shell:

```
docker exec hermes_mail_filter fangfrisch --conf /etc/fangfrisch/fangfrisch.conf refresh
```

Fangfrisch is conservative — it will still skip feeds whose own per-feed `interval =` window has not elapsed. To force a re-download of a single feed regardless of cadence, the Fangfrisch state DB can be cleared for that feed:

```
docker exec hermes_mail_filter sqlite3 /var/lib/fangfrisch/db.sqlite \
  "DELETE FROM refreshlog WHERE source = '<section_name>';"
```

Then re-run the refresh. The post-update hook re-validates with `clamscan` and deploys to `/var/lib/clamav/`. To inspect downloaded files:

```
docker exec hermes_mail_filter ls -la /var/lib/fangfrisch/signatures/
```

## Failure semantics

Failure	Behavior
Global save with non-allowlisted <code>log_level / max_size / timeout / interval</code>	<code>session.m=malware_feeds_error</code> , <code>alerttype=danger</code> , <code>alertmsg</code> explains; no DB write
Add Custom Feed with duplicate section name	<code>session.m=error</code> , <code>alertmsg</code> names the conflict; INSERT not attempted
Toggle/edit on non-existent <code>feed_id</code>	<code>session.m=error "Feed not found"</code> ; no UPDATE
Delete attempted on a built-in feed	UI suppresses the button; handler refuses the row
API key decryption error at INI regen	INI line replaced with <code>## &lt;name&gt; = [decryption error]</code> ; save still completes; Fangfrisch will treat the auth as missing on the next run
Container down during <code>chown / chmod</code>	<code>cftry</code> swallows the exec failure; INI is still written to the bind mount and the <code>chown</code> is applied next save when the container is back up
dos2unix binary missing	Tolerated via <code>cftry</code> ; INI is written without the line-ending normalization step

## Files and containers touched

Path	Owner	Role
<code>config/hermes/var/www/html/admin/2/view_malware_feeds.cfm</code>	<code>hermes_commandbox</code>	The page
<code>config/hermes/var/www/html/admin/2/inc/malware_feeds_*.cfm</code>	<code>hermes_commandbox</code>	Validate / save / regen per action

Path	Owner	Role
<code>config/hermes/var/www/html/admin/2/inc/generate_malware_feeds_configuration.cfm</code>	<code>hermes_commandbox</code>	Renders the INI from the DB; runs on every action
<code>/etc/fangfrisch/fangfrisch.conf</code>	<code>hermes_mail_filter</code> (bind-mounted, root:clamav, 640)	Live Fangfrisch config
<code>/var/lib/fangfrisch/db.sqlite</code>	<code>hermes_mail_filter</code>	Per-feed last-refresh state
<code>/var/lib/fangfrisch/signatures/</code>	<code>hermes_mail_filter</code>	Raw downloads (pre-validation)
<code>/var/lib/clamav/</code>	<code>hermes_mail_filter</code> (Docker named volume <code>mail_filter_data_clamav</code> )	Validated signature store; ClamAV reads from here
<code>/usr/local/bin/setup-clamav-sigs</code>	<code>hermes_mail_filter</code> (image-baked)	Post-update validation + deploy hook
<code>/opt/hermes/keys/hermes.key</code>	<code>hermes_commandbox</code> only	AES key for <code>api_key_*_value</code> columns
<code>malware_feeds_config</code> table	<code>hermes_db_server</code> ( <code>hermes</code> DB)	Per-feed row state
<code>malware_feed_urls</code> table	<code>hermes_db_server</code>	Per-feed URL list (FK cascade delete on feed delete)
<code>parameters2</code> rows <code>module='malware_feeds'</code>	<code>hermes_db_server</code>	Global Settings card
<code>ofelia_jobs</code> row <code>hermes-fangfrisch-refresh</code>	<code>hermes_db_server</code>	Schedule (auto-updated when Refresh Interval changes)

## Related

- [Antivirus Settings](#) -- the ClamAV engine that consumes the signatures Fangfrisch downloads; engine toggles (ScanMail, ScanArchive, etc.) and the per-engine signature whitelist live on that page
- [Scheduled Tasks](#) -- the Ofelia admin page; the `hermes-fangfrisch-refresh` job row is editable there (manual Run Now, enable/disable)
- [Score Overrides](#) -- per-rule SpamAssassin weight changes; not related to ClamAV but the closest neighbor for the "tune a built-in rule" pattern
- [Antispam Settings](#) -- SpamAssassin runs in the same Amavis pass; a ClamAV virus verdict from a Fangfrisch-supplied signature always pre-empts the spam score
- [DNS Resolver](#) -- every Fangfrisch HTTP download resolves through `hermes_unbound`; outbound HTTPS to the feed providers must be reachable
- [Email flow](#) -- full pipeline diagram showing where ClamAV (and therefore feed-derived signatures) fits

Revision #48

Created 2026-05-31 12:52:27 UTC by Dino Edwards

Updated 2026-06-20 13:33:15 UTC by Dino Edwards