Message History

Message History

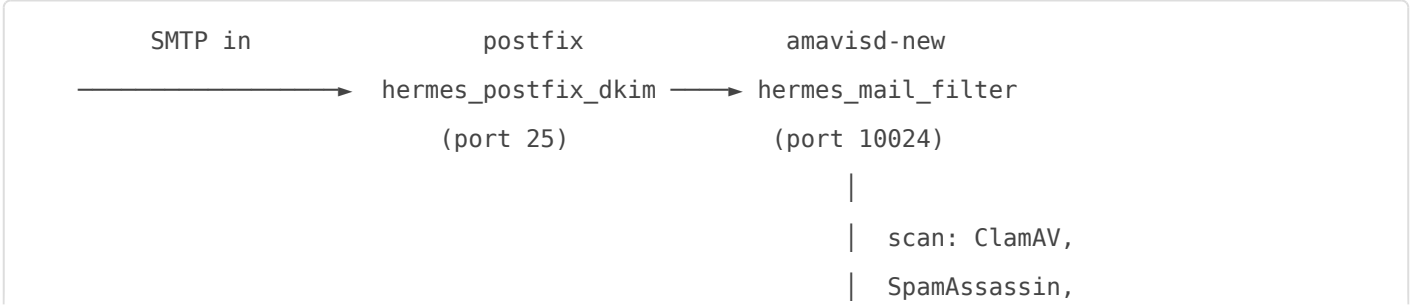
Admin path: **Content Checks > Message History** (`view_message_history.cfm`, `view_message.cfm`, `inc/messages_release_message.cfm`, `inc/messages_block_sender.cfm`, `inc/messages_allow_sender.cfm`, `inc/messages_train_ham.cfm`, `inc/messages_train_spam.cfm`, `inc/messages_forget_bayes.cfm`, `inc/messages_sa_learn_sync.cfm`).

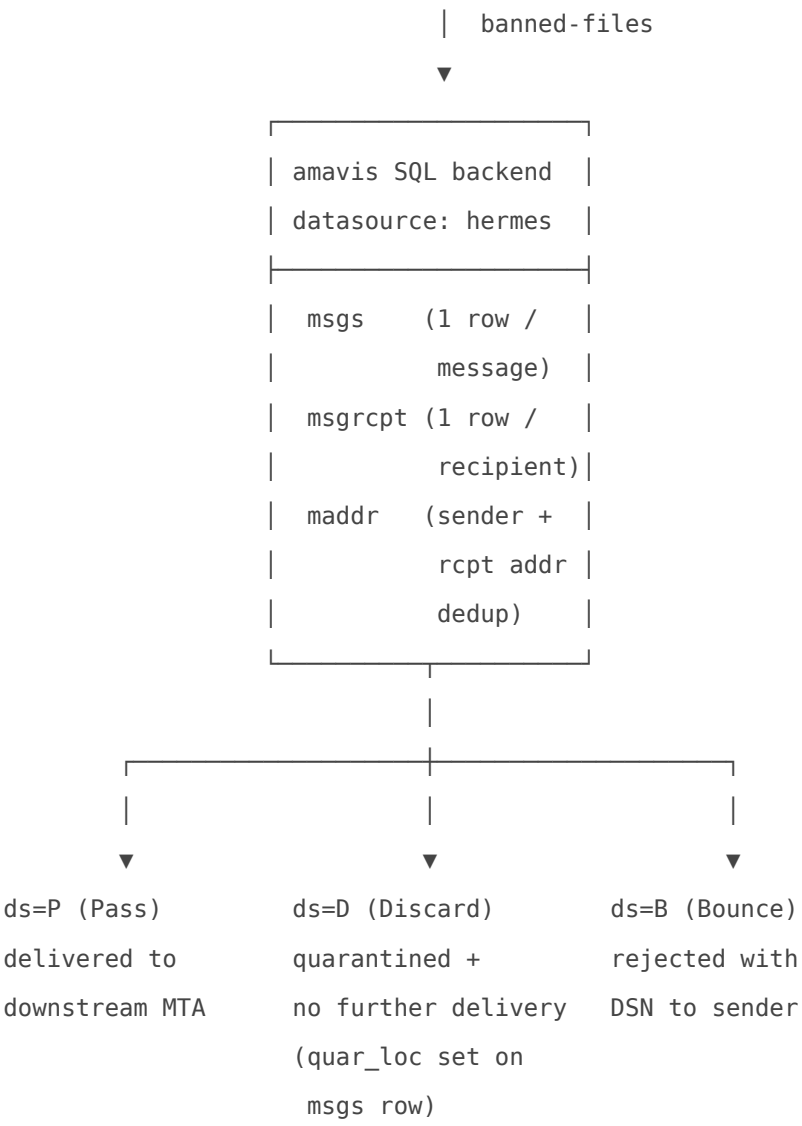
This is the **operator inspection surface** for everything that has flowed through the content filter. Every message Amavis processes lands as one row in `msgs` (per-message metadata) plus one row per recipient in `msgrcpt` (per-recipient disposition). This page is the joined view over those two tables, with a date range filter, a content-type filter, a delivery-status filter, and per-row actions to release from quarantine, train Bayes, or block/allow the sender.

Pairs with System Logs and Mail Queue. System Logs shows the raw syslog stream (connection negotiation, milter results, queue lifecycle). Mail Queue shows what Postfix is currently holding. **Message History shows what the content filter saw, what verdict it produced, and what landed where** -- and lets the admin act on those rows.

The same `msgs` table feeds the **Messages Processed** donut on System Status; the per-user self-service version of this view lives at `/users/2/view_message_history.cfm` and is scoped to the logged-in recipient only.

How a message gets into `msgs` and `msgrcpt`





The `ds` ("disposition") column on `msgrcpt` is the per-recipient verdict. The `content` column on `msgs` is the per-message **why** -- virus, spam, banned attachment, bad header, oversized, clean, etc. Together they answer "did this message get through, and if not, what blocked it?"

What's in the search form

The Search Messages card at the top of the page is the filter set; all fields are submitted as URL params so any search is bookmarkable and back-button safe.

Field	URL param	Effect
Start Date/Time	`startdate`	Lower bound on `msgs.time_iso`. Defaults to **24 hours ago**. Validated as a date by `isValid("date", ...)`; invalid values short-circuit to the error template

Field	URL param	Effect
End Date/Time	`enddate`	Upper bound on `msgs.time_iso`. Defaults to **now**. Same validation as startdate
Search Results Limit	`limit`	`LIMIT` clause on the join query. One of `1000`, `1500`, `2500`, `5000`, `10000`, `15000` -- the dropdown is the allowlist, anything else aborts. **Defaults to 1000.** The form text warns: setting limit to 10000+ significantly increases page load time
Type	`content_filter`	Multi-select against `msgs.content` -- the per-message content type (see table below). Empty = all types. Tom Select widget with remove and clear buttons
Action	`action_filter`	Multi-select against `msgrcpt.ds`. Empty = all actions. Three options: `P` Delivered, `D` Blocked (Discarded), `B` Blocked (Bounced)

The date pickers are Tempus Dominus widgets bound to the start/end inputs at page load; they emit `yyyy-MM-dd HH:mm:ss` into the form fields so the validation regex matches whether the admin types the date or picks it.

The `msgs.content` codes -- "what was this?"

These are the values rendered by the Type column and the values used by the Type multi-select. They come from the `msg_content_type` table (seeded at install time):

Code	Description	Meaning
`V`	Virus	ClamAV (or another configured scanner) hit a signature
`B`	Banned	A File Rule regex matched an attachment name, MIME type, or archive member
`U`	Unchecked	Amavis received the message but didn't scan (bypass policy, scanner failure, oversized, etc.)

Code	Description	Meaning
`S`	Spam Quarantined	SpamAssassin score reached `spam_kill_level` per the recipient's SVF Policy
`M`	Bad-Mime	MIME structure invalid in a way that broke the parser
`H`	Bad-Header	Header malformed per RFC; subject to per-policy `bad_header_lover`
`O`	Oversized	Message exceeded the configured size limit
`T`	Mta Error	Downstream MTA rejected the release / delivery attempt
`C`	Clean	Scanned, no findings, delivered
`Y`	Spam Tagged	Score reached `spam_tag2_level` (tagged with header) but stayed below `spam_kill_level` (delivered)
`s`	Spam Tagged (OLD)	Legacy lowercase variant; preserved for back-compat with older `msgs` rows

The score column shown on the table is `msgs.spam_level` -- the raw SpamAssassin score from the scan, **not** the per-policy threshold. A row tagged `S` with score `7.2` means the recipient's SVF policy has a `spam_kill_level` of `7.2` or lower.

The `msgrcpt.ds` codes -- "where did it go?"

`ds` is one character per recipient row:

`ds`	Column header	Meaning
`P`	Delivered	Pass -- handed to the downstream MTA (Postfix re-injection on port 10025 for relay topology, LMTP to Dovecot for mailbox topology)
`D`	Blocked	Discard -- not delivered, quarantined on disk under `/mnt/data/amavis/<quar_loc>`
`B`	Blocked	Bounce -- rejected at SMTP time with DSN to sender

ds	Column header	Meaning
anything else	N/A	Unexpected disposition; usually means amavis was killed mid-handoff or the row is partial

Per-recipient is the key: a single message with three recipients can have one `P`, one `D`, and one `B` row in `msgrcpt`. The table renders each `msgrcpt` row separately even though they share a `mail_id`.

The results table

The DataTable below the search card is sortable, paginated (50 / 75 / 100 / All rows per page), and exportable (Copy, CSV, Excel, PDF, Print buttons rendered by the DataTables Buttons extension). Default sort is Date/Time descending.

Column	Source	Notes
Checkbox	`msgs.mail_id`	Selects the row for the Message Actions modal. **Select All** in the header checks every checkbox on the current page
View	--	Magnifier button; opens `view_message.cfm?mid=<mail_id>` with the same `startdate` / `enddate` / `limit` so the back link round-trips correctly
Archived	`msgs.archive`	`Y` if the quarantine file has been moved to the long-term archive mount, `N` if it's still in the live amavis quarantine. Drives where `view_message.cfm` reads the EML from
Date/Time	`msgs.time_iso`	Indexed (`idx_msgs_time_iso`); this is the column the date range filters on. Rendered `yyyy-mm-dd HH:mm:ss`
Sender IP	`msgs.client_addr`	The client IP that handed the message to Postfix. For inbound that's the upstream MTA; for outbound it's the relay submitter
Return-Path	`maddr.email` via `msgs.sid`	The envelope sender (`MAIL FROM`); resolved via the `maddr` address-dedup table
From	`msgs.from_addr`	The header `From:` -- which is what users see and what DMARC aligns to
To	`maddr.email` via `msgrcpt.rid`	The envelope recipient. **Per-recipient** -- one table row per `msgrcpt` row

Column	Source	Notes
Subject	`msgs.subject`	Decoded subject header
Score	`msgs.spam_level`	Numeric score from SpamAssassin; formatted with 2 decimal places
Type	`msg_content_type.description`	Translated from `msgs.content` -- see the code table above
Action	derived from `msgrcpt.ds`	`Delivered` / `Blocked` / `Blocked` / `N/A`

If the date range returns zero rows, the table is replaced by an info alert ("No messages were found for the selected date range").

The View action -- `view_message.cfm`

Clicking the magnifier opens the per-message detail page. What that page can show is gated by two install-time toggles in `/opt/hermes/config/security.conf`:

Toggle	Default	Effect
`ALLOW_MESSAGE_CONTENT=yes`	off	Show the decoded message body (HTML + text). When off, only headers are rendered
`ALLOW_ATTACHMENT_DOWNLOAD=yes`	off	Render the attachment list with a download button per attachment. When off, attachments are silently not listed

Both default off because viewing a quarantined message body is a privileged operation -- it's the difference between "the admin can see a message was rejected" and "the admin can read a user's mail." Sites that need release-decision support enable `ALLOW_MESSAGE_CONTENT`; sites that need forensic attachment extraction enable `ALLOW_ATTACHMENT_DOWNLOAD`. The fast path reads only the raw MIME headers via a buffered Java reader so the headers page loads cheaply even on huge quarantine files; full-body parsing only happens when the toggle is on.

The EML is read from one of two paths depending on `msgs.archive`:

- `archive='N'` -> `/mnt/data/amavis/<quar_loc>` (live amavis quarantine)
- `archive='Y'` -> `/mnt/hermesemail_archive/mnt/data/amavis/<quar_loc>` (long-term archive)

If the file no longer exists on disk, the page aborts to the error template instead of returning a partial render.

Message Actions -- the bulk-action modal

Above the results table, the **Message Actions** button opens a modal that applies one of six actions to every row whose checkbox is ticked. The action runs in a CFML loop over the comma-delimited `mail_id` list; each iteration includes the matching action template per-message.

Action	Include	What it does
Block Sender	`inc/messages_block_sender.cfm`	Adds the envelope sender to the Amavis WB-list as `B` for the recipient of that message. Honors virtual-recipient validation -- bulk attempts against unknown recipients land in `failureinvalidrecipient_email`
Allow Sender	`inc/messages_allow_sender.cfm`	Same as Block Sender but writes `W` (whitelist). The recipient's future mail from that sender bypasses spam scoring
Release Message(s) to Recipient	`inc/messages_release_message.cfm`	Calls `docker exec hermes_mail_filter /usr/sbin/amavisd-release <quar_loc> <secret_id> <recipient>`. Re-injects the message from the quarantine file into Postfix for delivery. Success detected by parsing `250 2.0.0` out of the amavisd-release stdout
Train Message(s) as Spam	`inc/messages_train_spam.cfm`	Runs `sa-learn --spam` against the quarantine EML so Bayes learns that pattern as spam
Train Message(s) as Ham (NOT Spam)	`inc/messages_train_ham.cfm`	Runs `sa-learn --ham` so Bayes learns that pattern as legitimate. Use this on the false positives released from quarantine
Remove Message(s) Previous Training	`inc/messages_forget_bayes.cfm`	Runs `sa-learn --forget` to undo a prior `--spam` or `--ham` call against the same message

After any of the three Bayes actions, the page calls `inc/messages_sa_learn_sync.cfm` (which `docker exec`s `sa-learn --sync` to flush the in-memory token store to the Bayes database) and then runs `/opt/hermes/scripts/bayes_chown_amavis.sh` so the freshly written Bayes files stay owned by the amavis UID inside the content-filter container. **Don't skip the sync** -- without it, scoring decisions based on the new training only land after amavis's next periodic auto-sync, which is up to an hour out.

The release-message path is the most operationally important: it requires the quarantine file still exists on disk (the message hasn't been pruned by the cleanup job), `amavisd-release` exits with a `250`, and the downstream MTA accepts the re-injection. Any of those failing puts the row in `failurereleasemessage_email` and surfaces a red alert.

> **By design.** Releasing a message does **not** automatically train it as ham. If a quarantined spam is actually legitimate, run **Release Message** *and* **Train as Ham** as separate bulk actions so Bayes learns the false positive.

Status alerts -- the `m` flow

The page uses a `session.m` integer to pipe action-outcome alerts between the action-handler block (top of file) and the alert renderers (also top of file, after parameter setup). The handler sets `session.m = <code>` and `cflocation`s back to the same URL with the filter params preserved; the alert renderer reads `session.m`, emits the matching alert, and clears the variable.

`m`	Triggering action	Alert
`1`	Submit clicked with no rows ticked	"You must first select message(s) before clicking the Message Actions button"
`3`	Block Sender	success / warning
`4`	Allow Sender	success / warning
`5`	Release Message(s)	success / warning
`6`	Train Ham	success / warning
`7`	Train Spam	success / warning
`8`	Forget (remove training)	success / warning

The "warning" path fires when *some* rows in the bulk action failed -- the page lists both the successful and the failed subjects so the admin can re-target the failures.

Retention -- the message lifecycle

This page is **not the retention surface**; it is the read/action surface against rows that the retention pipeline maintains. Two scheduled jobs (registered as Ofelia jobs against `hermes_commandbox`) own the message lifecycle:

Schedule	Endpoint	Job
`0 30 01 * * *` (01:30 daily)	`schedule/message_cleanup.cfm`	Prunes `msgs` + `msgrcpt` rows past the configured retention window and deletes the matching quarantine files from `/mnt/data/amavis/`
`@every 60s`	`schedule/quarantine_notify.cfm`	Reads the `idx_msgrcpt_notify` index, sends recipient-facing quarantine notifications for new `ds=D` rows that haven't been notified yet, and flips `notification_sent=1`

Both are managed from Scheduled Tasks; retention thresholds and per-content-type quarantine targets are configured on Anti-Spam Settings. The cleanup job is the reason a `Release Message` action can fail with "quarantine file does not exist" -- if you wait past the retention window, the EML is gone and only the `msgs` row remains as a record.

Performance notes

The base join (`msgs INNER JOIN msgrcpt ON msgs.mail_id = msgrcpt.mail_id`) is hit on every page load with a `WHERE msgs.time_iso BETWEEN ?` range. `idx_msgs_time_iso` is the index that makes the date range cheap; without it the query degrades to a full table scan and pages with `limit=15000` would time out on a busy gateway. The per-row sub-queries (`getfromaddr`, `gettoaddr`, `gettype`) fire **once per result row** because they were originally written with N+1 semantics; on `limit=15000` that's 60K+ extra queries plus 15K DataTable rows being rendered into the DOM. The "10000+ significantly increases page load time" warning on the form is calibrated against that reality.

Don't widen the date range and crank the limit at the same time when debugging a specific incident. Narrow the window first, then widen the limit only if you have to.

Related pages

- Mail Queue -- live queue (what Postfix is currently holding) vs. this page's historical record
- System Logs -- raw `mail.*` syslog stream; use when this page shows a row but you need the connection / milter / delivery trace behind it
- System Status -- the dashboard donut that aggregates the same `msgs` rows
- Scheduled Tasks -- the cleanup + notify jobs that maintain the data this page reads
- Anti-Spam Settings -- spam thresholds, Bayes configuration, and quarantine retention windows
- Anti-Virus Settings -- ClamAV configuration that drives `content='V'` verdicts

- SVF Policies -- per-recipient `spam_kill_level` that decides whether a scored message lands here as `S` (quarantined) or `Y` (delivered with header)
- File Rules -- attachment regexes that drive `content='B'` verdicts
- ARC Settings and DMARC Settings -- upstream authentication signals that contribute to spam scoring and so influence which messages land here as `S` vs `Y`

Revision #8
Created 2026-05-31 12:52:27 UTC by Dino Edwards
Updated 2026-05-31 14:01:22 UTC by Dino Edwards