Mail Queue

Mail Queue

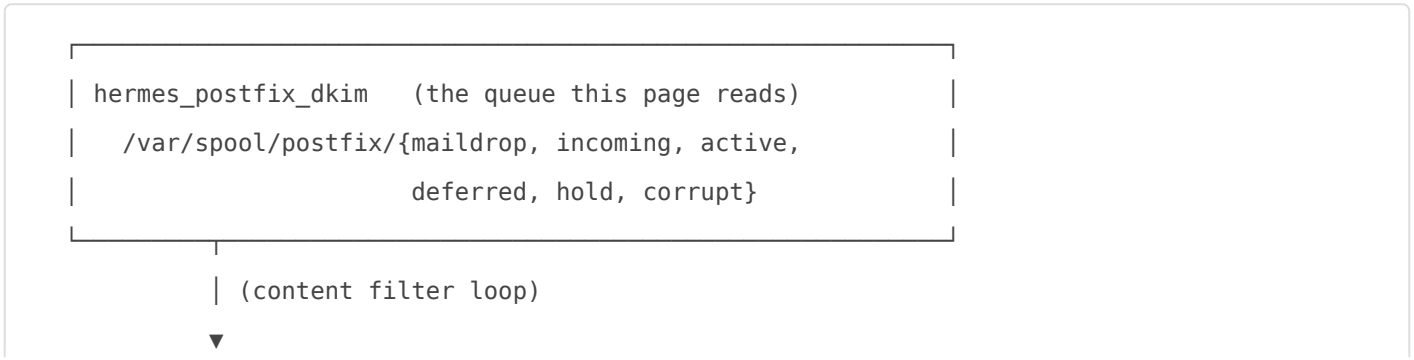
Admin path: **System > Mail Queue** (`view_mail_queue.cfm`, `inc/get_mail_queue_settings.cfm`, `inc/mail_queue_get_queue.cfm`, `inc/mail_queue_action.cfm`, `inc/mail_queue_flush_mailqueue.cfm`, `inc/mail_queue_set_queue_settings.cfm`, `view_mail_queue_message.cfm`, `inc/mail_queue_view_message.cfm`).

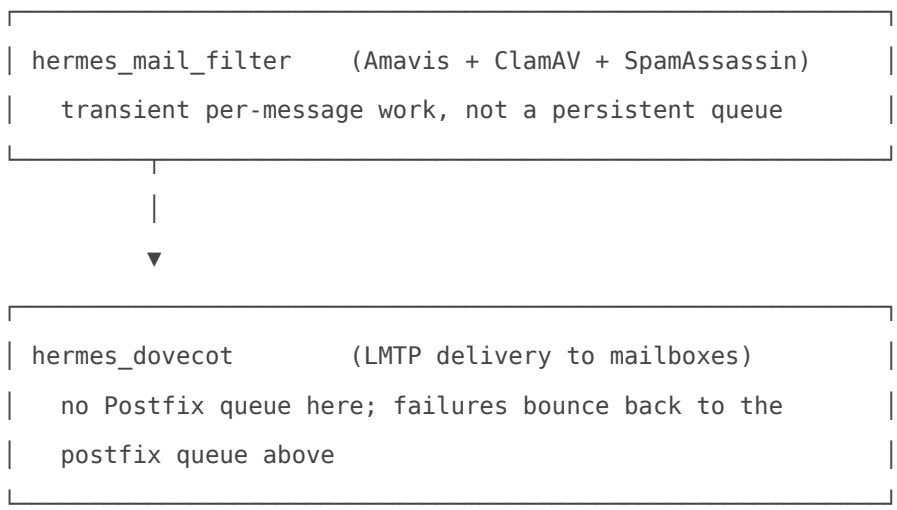
This page is the operator's window into **Postfix's on-disk queue inside `hermes_postfix_dkim`** — the messages Postfix has accepted but not yet finally delivered or bounced. It does two unrelated jobs that share one page:

1. **Queue Settings** — two Postfix tunables (`bounce_queue_lifetime` and `maximal_queue_lifetime`) stored in the `parameters` table and pushed into `main.cf` via the generic Postfix config regen path.
2. **Queue Viewer / Actions** — a live read of `mailq` plus per-message Hold / Unhold / Re-queue / Delete operations and a queue-wide Flush.

The viewer is read-only against `mailq`; everything that mutates the queue goes through `postqueue` or `postsuper` inside the container. Hermes never edits `/var/spool/postfix/*` directly, so admin actions respect Postfix's own queue locking and are safe to run while mail is flowing.

The queue this page shows — and the ones it doesn't





Postfix is the only component that maintains a persistent on-disk spool. A message you see in this viewer is a message Postfix is still holding — it has not been handed off to the next hop (LMTP to Dovecot, remote MX, satellite Amavis), or it was handed off and bounced back into `deferred`, or an admin moved it into `hold`. Amavis's transient work is not a "queue" in the Postfix sense and is not visible here; if the content filter is stuck, messages pile up in `active` on the gateway side, which this page does surface.

Queue Settings

Two values, both saved into rows of the `parameters` table keyed by `parameter = 'bounce_queue_lifetime'` / `'maximal_queue_lifetime'` (`child = 2` parent rows, with the user-selected value stored in the `child = 1` row). The dropdowns range 0–90 days.

Setting	`main.cf` directive	Meaning
Bounce Queue Lifetime	`bounce_queue_lifetime`	How long Postfix retries a bounce message that cannot be delivered to its envelope sender before giving up. `0` means single-delivery attempt only — failing bounces are double-bounced to the postmaster immediately.
Max Queue Lifetime	`maximal_queue_lifetime`	How long Postfix retries a normal message before generating a permanent failure (bounce) to the sender. `0` means single-delivery attempt only.

Both values are stored as integers in the dropdown but written into the DB with the `d` suffix (e.g. `5d`) so they go straight into `main.cf` unmodified. Hermes regenerates `main.cf` from the `parameters` table on save and reloads Postfix; there is no incremental edit path. See the Server Setup doc for the broader Postfix regen pipeline.

> **Why `0` is a real choice.** `bounce_queue_lifetime = 0` is the upstream-recommended default for relays — a bounce that cannot be delivered is more likely a forged sender than a real recipient mailbox, and keeping it in the queue for days wastes attempts on joe-job traffic. Leave the seed value unless you have a specific reason to change it.

Queue Viewer — how the table is built

`inc/mail_queue_get_queue.cfm` does the live read in three phases:

1. **Summary probe.** Runs `docker exec hermes_postfix_dkim /bin/bash -c '/usr/bin/mailq | /usr/bin/tail -1'` to read just the trailing `-- N Kbytes in M Requests.` line and parse `M` out as the total queue count. This is cheap — no full parse, no full transfer of the queue contents.
2. **Overload gate.** If the total exceeds **500** (`maxQueueLoad`), the viewer refuses to load the queue at all. The page renders a red callout with the count and shell hints (`postsuper -d ALL`, `postsuper -H ALL`) for the admin to recover from the command line. This is a self-protection step — parsing tens of thousands of `mailq` lines in CFML would hang the page and lock a CommandBox worker thread.
3. **Full parse.** If under 500, runs `docker exec hermes_postfix_dkim /usr/bin/mailq` and parses the multi-line output in CFML into a query object with `QueueID`, `Sender`, `Recipient`, `ConnectionStatus`, and `MsgStatus`. The display table is capped at **100 rows** (`maxQueueDisplay`); a yellow callout appears if the queue has between 101 and 500 entries.

The parser reads the per-entry queue-ID suffix to derive the status column. Postfix's `mailq` marks active messages with `*` and held messages with `!` after the queue ID; everything else is treated as `deferred` (rendered as `N/A` in the badge). This is by design — the viewer is a snapshot, not a queue-state diff.

Suffix	`mailq` meaning	Rendered as
`*`	currently being delivered (in `active`)	green `ACTIVE` badge
`!`	admin-held (in `hold`)	yellow `ON-HOLD` badge
(none)	waiting for retry (in `deferred`)	grey `N/A` badge

The `ConnectionStatus` column is whatever Postfix put in parentheses on the line after the message header (typically the SMTP error from the last delivery attempt — `Connection refused`, `Greylisted, please try again`, etc.). For messages that have never been attempted it is blank.

View Message (`view_mail_queue_message.cfm`)

Clicking the magnifying glass on a row opens a full dump of the queued message — headers and body — via `docker exec hermes_postfix_dkim /usr/sbin/postcat -q <queueid>`. The output is rendered into a plain textarea with a print button. No edit, no resend; if you need the message to go out, use Re-queue from the main viewer.

Per-message actions

All four mutation actions converge on `inc/mail_queue_action.cfm`, which validates the queue ID against `^[A-Fa-f0-9]+$` (defence against shell injection) and shells out to `postsuper` with the right flag:

Action	Postsuper flag	What it does	Typical use
Hold	`-h`	Moves the message into `hold/`. Postfix will not touch it again until unheld.	Pause a stuck loop, freeze a message for forensic copy, hold while debugging upstream issues
Unhold	`-H`	Moves the message back into `deferred/` so retries resume	Recover a held message after the underlying issue is fixed
Re-queue	`-r`	Re-injects the message through the cleanup daemon, re-applying milter chain (OpenDKIM, OpenDMARC, body milter), header_checks, etc.	Force a fresh content-filter pass — useful after fixing a milter, updating a header_check rule, or changing a relay map
Delete	`-d`	Removes the message from the queue **permanently**. No undo.	Drop spam, drop a stuck message you don't want re-delivered, drop a confirmed mail loop

The action handler loops the selected queue IDs and invokes `postsuper` once per ID via a generated temp script under `/opt/hermes/tmp/` — `postsuper` writes its result to stderr, and the temp-script pattern (with `2>&1`) is the only reliable way to capture it from `cfexecute`. Per-ID success or failure is counted independently; the result alert shows both the count and the queue IDs in each bucket.

> **Re-queue is not the same as Flush.** Re-queue re-injects through the milter / content-filter chain (so a fresh OpenDKIM signature is generated, the disclaimer milter runs again, etc.). Flush just nudges Postfix to retry delivery on what is already in `deferred`. If a message is broken because of a milter failure during the original intake, Re-queue can fix it; Flush will not.

Flush Queue

The Flush button runs `docker exec hermes_postfix_dkim /usr/sbin/postqueue -f`. This is a queue-wide "retry now" — it scans the `deferred` queue and moves eligible messages into `active` for an immediate delivery attempt. Held messages are not touched.

A success result means `postqueue` exited cleanly, not that delivery succeeded. If a deferred message's destination is still unreachable, it goes right back into `deferred` after the attempt. Use the System Logs page (or `/remotelogs/postfix/mail.log` for live tail) to see the actual delivery outcomes.

Overload mode — the bulk-recovery path

When the queue exceeds 500 messages the page deliberately refuses to render the table. Both shell-hint commands in the callout are full queue-wide operations that bypass the per-message UI:

```
# Delete everything in the queue (no exceptions, no confirmation)
docker exec hermes_postfix_dkim postsuper -d ALL

# Move every held message back to deferred
docker exec hermes_postfix_dkim postsuper -H ALL
```

These are the standard Postfix mass-action commands. There is no selective `-d` for "delete only spam-bounce" or similar; if you need granular cleanup of a large queue, filter first with `mailq` and a custom shell pipeline, then run `postsuper -d` on the resulting list.

> **Why a hard cap and not pagination.** Pagination would require parsing the full `mailq` output to know the row count anyway, which is the expensive operation we are trying to avoid. The hard cap forces the admin into the command line where the right tools live for bulk queue work.

Concurrent safety

Every action goes through `postqueue` or `postsuper`, which acquire Postfix's own queue locks before touching files. Multiple admins hitting the page in parallel cannot corrupt the queue — at worst, two Delete clicks on the same queue ID will have one succeed and the other return "no such queue file", which is rendered as a failure row in the result alert. The viewer itself is read-only and the `mailq` snapshot can race with mutations (a message you tick may have already been delivered by the time you click the action), which is also fine — the mutation just no-ops with the same "no such queue file" message.

Related pages

- Server Setup — Postfix `myhostname`, `myorigin`, and the `parameters` → `main.cf` regen path that this page's Queue Settings hooks into.
- System Logs — where delivery outcomes for queued messages actually surface (Postfix logs to mail.* → rsyslog → `SystemEvents` → this viewer).
- Intrusion Prevention — IP-level bans for brute-force SMTP-AUTH that show up in Postfix's connection logs.

Revision #64
Created 2026-05-31 12:51:59 UTC by Dino Edwards
Updated 2026-08-22 11:48:28 UTC by Dino Edwards