Disclaimers

Disclaimers

Pro Edition feature. Maps to **Email Policies > Disclaimers** (`view_disclaimers.cfm`, `edit_disclaimer.cfm`, `disclaimer_delete.cfm`).

Hermes appends a configurable disclaimer to outbound mail at the gateway, with two scopes:

Scope	Sender match	Use case
Domain	All senders in `@example.com`	Default org-wide compliance/legal language
Relay Recipient	Specific full address (e.g. `vendor@example.com`)	Per-relay-user override for tenants with extra regulatory language

Most-specific match wins: a relay-recipient match is used before the domain default.

Pipeline placement

Disclaimers are applied at SMTP receive time by the **`hermes_body_milter`** container, which Postfix consults as a milter alongside OpenDKIM and OpenDMARC.

```
External MTA / MUA submission
        │
        ▼
Postfix smtpd
   ├─ smtpd_milters chain (in order):
   │   1. OpenDKIM            (signs/verifies)
   │   2. OpenDMARC           (DMARC policy)
   │   3. hermes_body_milter  (THIS — disclaimers, signatures, banners)
   ▼
content_filter → Amavis    (unmodified path; sees the body milter's output)
   ▼
Ciphermail              (server-side S/MIME or PGP, if configured)
```

```
   ▼
Postfix :10026          (OpenDKIM signs the final composed body here)
   ▼
external
```

Body modification happens at smtpd time, **before** content_filter routes to Amavis. By the time Amavis sees the message, the disclaimer is already baked in. Amavis processes a normal-looking message; no internal-state coupling, no temp-file races.

OpenDKIM's outbound signing fires at the `:10026` re-injection — after both the body milter and Ciphermail. Hermes' own DKIM therefore always covers whatever the recipient ultimately receives. Ciphermail's server-side crypto also covers the disclaimer because Ciphermail runs after the milter.

Behavior with S/MIME, PGP, and DKIM-signed mail

The behavior depends on **who** signed/encrypted the message and **when** in the pipeline.

Server-side: signed/encrypted by Ciphermail — disclaimer is applied

Ciphermail runs **after** the body milter. Mail arrives at the milter as plaintext, the disclaimer is appended, then Ciphermail signs or encrypts the modified body. The recipient sees a valid signature **and** the disclaimer. No conflict.

Client-side: signed/encrypted by the user's MUA — disclaimer is skipped

Mail signed in Outlook (S/MIME) or Thunderbird+Enigmail (PGP) arrives at the gateway with the cryptographic envelope already sealed. Modifying the body would either invalidate the signature or mangle the ciphertext.

The body milter detects the following patterns in the headers (or first 32 KB of the body) and exits unchanged when any matches:

Pattern matched	Meaning

`Content-Type: multipart/signed; protocol="application/pkcs7-signature"`	S/MIME detached signature
`Content-Type: application/pkcs7-mime`	S/MIME opaque-signed or enveloped
`Content-Type: multipart/signed; protocol="application/pgp-signature"`	PGP/MIME detached signature
`Content-Type: multipart/encrypted; protocol="application/pgp-encrypted"`	PGP/MIME encrypted
`-----BEGIN PGP SIGNED MESSAGE-----` in body	PGP inline-signed
`-----BEGIN PGP MESSAGE-----` in body	PGP inline-encrypted

When any of those match, the body is left untouched, the signature stays valid, the user's legal-text expectations are preserved (their MUA template is already in the body), and the gateway gets out of the way.

> **Operational consequence.** A site whose users sign client-side will not get gateway disclaimers on those specific signed messages — by design. If org-wide legal text on **all** outbound is mandatory, the only safe pattern is server-side signing in Ciphermail with the disclaimer applied first.

DKIM: Hermes-signed mail is fine; upstream-signed mail is skipped

OpenDKIM signs at the Postfix `:10026` re-injection step — **after** the body milter. So Hermes' own DKIM signature always covers the recipient's view of the message (with disclaimer baked in). No conflict.

The risk is mail that arrives at Hermes **already DKIM-signed by an upstream MTA** — typically a relay user whose own mail server signs before forwarding through us. Modifying that body would invalidate the upstream signature at the recipient.

The body milter treats a pre-existing `DKIM-Signature:` header the same way as a sealed S/MIME or PGP envelope and skips the disclaimer. Since Hermes' own DKIM signs at `:10026` (downstream of this milter), any DKIM-Signature header present at the milter's point in the pipeline came from somewhere upstream of Hermes.

Reply-chain handling — no dedup, by design

The milter does **not** detect or skip messages that already carry a previous disclaimer in their quoted history. Every outbound message gets a fresh disclaimer applied — including replies inside a long thread.

This matches industry norm: commercial server-side disclaimer / signature platforms (Exclaimer, Crossware, CodeTwo, Microsoft 365 transport rules) all stamp every outbound without dedup. The reasoning:

- **Compliance.** Many regulatory regimes (HIPAA email confidentiality, GDPR data-controller notices, financial-services disclosure) treat each transmission as requiring its own disclaimer. Stamping only the first message in a thread arguably leaves later replies non-compliant.
- **Self-contained messages.** If a recipient forwards a reply (with quoted history) to a third party, the disclaimer is preserved per-message in the forwarded text.
- **Predictable behavior.** Operators don't have to explain "sometimes the disclaimer shows, sometimes it doesn't."
- **Cosmetic concern is weak.** Modern MUAs (Gmail, Outlook, Apple Mail) collapse quoted history by default, so stacked disclaimers in long threads are rarely visible to readers.

Earlier iterations of #214 included a sentinel-marker dedup mechanism (`[HD]` / `<!-- HERMES_DISCLAIMER_V1 -->`). That was removed during DEV testing in favor of the industry-norm pattern.

Position: append vs prepend

The schema and UI both expose `position = append | prepend`, but **v1 honors append only.** Prepend is tracked as a v2 enhancement.

Failure semantics

The body milter is **graceful-degradation** by design. Postfix's `milter_default_action = accept` means:

- Milter container down or unreachable → mail flows **unmodified** (missed disclaimer, but no delivery outage)

- Map file unreadable → no entries match → all mail flows unmodified
- Modifier raises an exception → caught and logged → mail flows unmodified
- altermime / parse errors → caught and logged → mail flows unmodified

In every failure case, mail keeps flowing. Worst case is a missed disclaimer, never lost mail. Compare the legacy "modify in amavis hook" approach (#214 Phase 3 v1, retired) which silently dropped messages when the in-place body modification desynced amavis's internal state.

Files generated on save/delete

The CFML include `inc/disclaimer_write_and_reload.cfm` runs after every save or delete and rewrites the entire on-disk state from the `disclaimers` table:

```
/etc/hermes/body_milter/disclaimers/disclaimer_by_sender   sender → option map
/etc/hermes/body_milter/disclaimers/files/<option>/
    body.txt          plain-text disclaimer
    body.html         html disclaimer (may have <img src="cid:..."> refs)
    images/
        1.png         per-disclaimer inline images (#230)
        2.jpg
        ...
```

Where `<option>` is `domain_<safe>` or `relay_<safe>` (non-alphanumeric chars in the source key are replaced with `_`).

Each disclaimer gets its own subdirectory. The files directory is wiped (per-option subdirectories deleted recursively, but the parent `files/` directory and its `.gitkeep` are preserved) and rewritten on every save. There is no incremental update — this guarantees deleted rows and renamed scope keys never leave stale files (or stale image binaries) behind.

No reload step needed. The body milter mtime-watches each map file on every message and reloads when it changes. The CFML `cffile` write to the map file is enough to make the change take effect on the next message processed by the milter.

Inline images (#230)

Admins can paste or upload images directly into the Quill editor when authoring a disclaimer. Supported formats: **PNG, JPEG, GIF**. SVG and WebP are explicitly rejected (security and recipient-compatibility reasons). Limits enforced at save time:

- **5 images max** per disclaimer
- **200 KB per image** (after base64 decode)
- **1 MB total** across all images in a single disclaimer

If any limit is exceeded, the save is rejected with a specific error explaining what failed. Admins can reduce image count or size and re-save.

How it works:

1. Quill embeds pasted/uploaded images as base64 inline `<img src="...">` in the HTML body. The base64 representation is what's stored in the `disclaimers.body_html` column.
2. At save time, the regenerator parses `body_html` for `data:` URLs, decodes each base64 blob, writes the binary as `<option>/images/<N>.<ext>`, and rewrites the HTML in `<option>/body.html` to use `<img src="cid:disclaimer_<option>_img_<N>">` references.
3. At message-send time, the body milter reads `body.html`, walks `<img src="cid:...">` references, and attaches each referenced image as an `image/<format>` MIME part with `Content-ID: <disclaimer_<option>_img_<N>>` and `Content-Disposition: inline`.
4. The milter wraps the message as `multipart/related` so the recipient MUA resolves cid references against the inline parts.

MIME structure transformation (representative example):

```
Original outbound:
  multipart/alternative
    text/plain
    text/html (no images)

After milter (with disclaimer including 1 image):
  multipart/related
    multipart/alternative
      text/plain  (with text disclaimer appended; images omitted from text)
      text/html   (with html disclaimer + <img src="cid:...">)
    image/png
      Content-ID: <disclaimer_..._img_1>
      Content-Disposition: inline
```

This structure renders inline in all major MUAs (Gmail, Outlook, Apple Mail, Thunderbird, mobile clients).

The plain-text version of the disclaimer omits images entirely — base64 inline images don't translate to text, and recipients viewing the message in plain-text mode see the disclaimer text without any image markers.

Hermes' own DKIM signature covers the modified body (including the multipart/related wrap and image parts), because OpenDKIM signs at the postfix `:10026` re-injection step — downstream of the body milter. The signature validates against what the recipient receives.

Auto-derive of plain-text part

The Quill editor on `edit_disclaimer.cfm` drives `body_html`. By default the plain-text part shipped to recipients with a non-HTML MUA is auto-derived from the HTML on save: `<br>`, `</p>`, `</li>` become newlines, all other tags are stripped, runs of 3+ newlines collapse to 2.

Admins who need character-perfect plain text different from the auto-strip (e.g. for regulated industries) can toggle **Edit plain-text version separately** to expose a second editor. When set, `body_text` is shipped verbatim instead of derived.

Disabled rows

Rows with `enabled = 0` are skipped entirely on regen — no files written, no map entry. The milter never matches that scope until the row is re-enabled.

Internal-only mail

v1 does **not** suppress disclaimers for internal-only mail (sender + all recipients in `@local_domains`). Domain disclaimers will be applied to internal mail in the same domain. If this is a problem for your install, file a feature request to add an internal-only bypass.

Why a separate milter and not an amavis hook

Earlier #214 iterations attempted to dispatch the disclaimer from inside an amavisd-new `Custom.pm` `before_send` hook, calling altermime via `system()` on the temp file amavis was managing. amavisd-new 2.13 caused two problems: the legacy `@disclaimer_options_bysender_maps` dispatch path was removed (variables still parse but no code reads them), and the `before_send` hook documentation says "may modify mail" but in practice in-place body modification desynchronizes amavis's internal MIME state and silently loses mail.

The body milter approach moves the body-modification step out of amavis entirely. amavis's role is unchanged from before #214 ever existed; the milter sits in postfix's smtpd_milters chain alongside OpenDKIM and OpenDMARC, the same architectural pattern Hermes already uses for body-touching policy enforcement. amavis is fully decoupled from the disclaimer feature, which means amavis upgrades and the disclaimer feature evolve independently.

This same milter container is intended to host:

- **#226 User Signatures** (per-mailbox personal text from LDAP attributes or user-portal editor)
- **#228 External Sender Banner** (warning banner on inbound external mail)
- **Future Link Guard** (URL rewriting through a click-through endpoint)

Each is a `Modifier` subclass in `/usr/local/bin/hermes-body-milter` registered in the `MODIFIERS` list. The dispatcher is unchanged.

Revision #8
Created 2026-05-31 12:52:41 UTC by Dino Edwards
Updated 2026-06-13 12:29:38 UTC by Dino Edwards