External Banner

External Banner

Maps to **Email Policies > External Banner** (`view_external_banners.cfm`, `edit_external_banner.cfm`, `external_banner_delete.cfm`). Available on both **Community** and **Pro** editions — phishing protection is a baseline security feature, not a Pro upsell.

Hermes prepends (or optionally appends) a warning banner to **inbound** mail from external senders destined for a local recipient. The banner is injected into the message body itself, so every MUA — webmail, Outlook, Apple Mail, mobile clients — renders it without relying on transport rules or recipient-side configuration. Tracked as #228.

Scope

Scope	Recipient match	Use case
System default	All recipient domains (no override)	Single banner used everywhere; recommended starting point
Per-recipient-domain	Specific local mailbox domain (e.g. `legal.example.com`)	Different copy or compliance language for one domain

Resolution at message time, in the body milter's `ExternalBannerModifier`:

1. Look up the first **local recipient**'s domain in `/etc/hermes/body_milter/banners/banner_by_recipient_domain`.
2. If a matching row exists, use it.
3. Otherwise fall back to the `_default` system-wide entry.
4. Otherwise no banner is applied.

Only the first local recipient is consulted — mixed-domain envelopes get the banner of the first local recipient encountered. This keeps the modification deterministic regardless of envelope ordering.

The `recipient_domain` field is **locked after creation**. Delete and re-create the row to change scope.

What counts as "external"

The body milter uses Postfix's `/etc/postfix/relay_domains` file as the source of truth for "local". A message is considered **inbound from an external sender** when:

- The `MAIL FROM` sender domain is **not** in `relay_domains`, AND
- At least one `RCPT TO` recipient domain **is** in `relay_domains`.

Internal-to-internal mail (sender + all recipients local) is classified as `direction = internal` and the banner is **not** applied. There is no separate allowlist of "trusted partner" external senders today — every external sender to a local recipient triggers the banner if one is configured for that recipient's domain.

Pipeline placement

The banner is injected at SMTP receive time by the **`hermes_body_milter`** container, the same container that emits outbound disclaimers (disclaimers.md) and organizational signatures (organizational-signatures.md). The milter listens on `inet:hermes_body_milter:8893` and Postfix consults it as part of `smtpd_milters`.

```
Inbound external MTA
        |
        v
Postfix smtpd
   +- smtpd_milters chain (in order):
   |    1. OpenDKIM            (verifies upstream DKIM signature)
   |    2. OpenDMARC           (DMARC policy + ARC verification)
   |    3. hermes_body_milter  (THIS -- banner prepended here)
   |       --> Authentication-Results header has already been written
   |           by OpenDKIM/OpenDMARC BEFORE the banner touches the body
   v
content_filter --> Amavis    (sees the banner-prepended body)
   v
Ciphermail              (server-side S/MIME or PGP, if configured)
   v
Postfix :10026          (multi-instance OpenDKIM re-signs the final body)
   v
Local delivery (Dovecot LMTP)
```

Key ordering points:

- **OpenDKIM verifies first.** The upstream sender's DKIM verdict is captured in `Authentication-Results:` headers **before** the banner is injected. The header is preserved on the message; the banner does not retroactively change what OpenDKIM saw at smtpd time.
- **Amavis sees the modified body.** Spam scoring runs against the banner-prepended message. This is intentional — the banner content is short and stable and does not skew SpamAssassin scores in practice.
- **Hermes' downstream re-sign covers the modified body.** The multi-instance OpenDKIM at `:10026` (#232) signs after Ciphermail rebuild, so the final outgoing-to-Dovecot body is covered by Hermes' own signature.

Behavior with signed and encrypted mail

The modifier inherits the same skip rules as Disclaimers for sealed envelopes:

Pattern matched	Meaning	Banner action
`Content-Type: multipart/signed; protocol="application/pkcs7-signature"`	S/MIME detached	**Skip**
`Content-Type: application/pkcs7-mime`	S/MIME opaque/enveloped	**Skip**
`Content-Type: multipart/signed; protocol="application/pgp-signature"`	PGP/MIME detached	**Skip**
`Content-Type: multipart/encrypted; protocol="application/pgp-encrypted"`	PGP/MIME encrypted	**Skip**
`-----BEGIN PGP SIGNED MESSAGE-----` in body	PGP inline-signed	**Skip**
`-----BEGIN PGP MESSAGE-----` in body	PGP inline-encrypted	**Skip**
Pre-existing `DKIM-Signature:` header on inbound mail	Upstream DKIM signed	**Modify anyway** (see below)

The corresponding flags on `ExternalBannerModifier` are `skip_on_signed = True`, `skip_on_pgp_inline = True`, `skip_on_dkim = False`.

Why the banner does NOT skip on upstream DKIM

About 95% of inbound mail today carries a `DKIM-Signature:` header. If the banner skipped on DKIM, the feature would be effectively inert — the warning would only land on the unsigned minority that needs it least.

Hermes already records the upstream DKIM verdict in `Authentication-Results:` before modifying the body. Recipients overwhelmingly read mail through Dovecot/IMAP and the recipient MUA does not re-verify upstream DKIM. The banner is therefore safe in the common case.

The narrower edge case — a recipient who forwards Hermes-banner'd mail to a downstream MX that **does** re-verify upstream DKIM — is addressed by ARC sealing (#229). Hermes' ARC seal at `:10026` records `cv=fail` for the upstream chain (because we modified the body), but the seal itself is mathematically valid and the downstream MX can trust Hermes' ARC verdict if Hermes is on its allowlist. See ARC Settings for the full discussion of the `cv=fail`-by-design pattern.

> **Operational consequence.** Banner injection breaks the original sender's DKIM body hash and any upstream ARC body hash. **This is by design.** Hermes is the authoritative auth boundary for the domains it relays; customer downstream MX servers must allowlist Hermes and accept its delivered mail without re-running DKIM/SPF/DMARC/ARC. A downstream MX that re-verifies upstream auth on mail Hermes forwards is misconfigured — cross-ref ARC Settings, DKIM Settings, and DMARC Settings.

Position: prepend vs append

Position	Behavior	Recommendation
Top (`prepend`)	Banner becomes the first child of the message body (above any quoted history)	**Industry standard** — users see the warning before reading any content
Bottom (`append`)	Banner is appended after the user-visible body	Available for sites that prefer it; rarely used

Both positions are implemented end to end (unlike Disclaimers, where only `append` is honored in v1). HTML prepend is done with BeautifulSoup: the banner fragment is inserted as the first child of `<body>` when present, otherwise prepended to the root.

Templates

Banners use a **server-side template gallery**, not a free-form WYSIWYG editor. Quill 2.x's HTML normalization strips inline styles that Gmail and Outlook need (the same problem hit on Organizational Signatures #226 Phase 2 and on this feature), so admins pick a template and fill in form fields; the server renders pixel-perfect HTML at save time.

Bundled templates (each `inc/external_banner_templates/<key>.cfm`):

Template key	Display name	When to pick it
`warning_yellow`	**Warning Yellow**	Default. Yellow background with orange accent. Matches Microsoft 365 / Mimecast banner style most users recognize
`critical_red`	**Critical Red**	Red background, white text. Phishing-prone industries or post-incident periods where alert level needs to be raised
`subtle_info`	**Subtle Info**	Light gray with blue accent. Less alarming for high-volume inbound (support/sales) where alert fatigue is a concern
`plain_text`	**Plain Text**	Bold prefix + text, no background or border. Maximum cross-MUA compatibility, including text-only clients

All four templates expose the same field set:

Field	Type	Default	Notes
`prefix`	text	`[EXTERNAL]`	Short tag rendered bold at the start. Plain ASCII recommended for Outlook
`headline`	text	"This message originated from outside your organization."	First line, regular weight
`body`	text	"Do not click links or open attachments unless you recognize the sender..."	Second line, smaller text
`show_learn_more`	checkbox	`false`	Reveals the next two fields
`learn_more_url`	url	*empty*	Optional link to internal phishing-awareness training or wiki
`learn_more_label`	text	"Learn more about phishing"	Visible label for the learn-more link

All templates emit **table-based HTML with `bgcolor=` attributes** so Outlook (which strips inline CSS but honors deprecated HTML attributes) renders the banner correctly. Inline styles are belt-

and-suspenders for Gmail, Apple Mail, and mobile clients.

The edit page renders a live preview in an iframe via `inc/render_external_banner_preview.cfm` so the admin sees exactly what `save_external_banner_action.cfm` will store.

Files generated on save/delete

`inc/external_banner_write_and_reload.cfm` runs after every save or delete and rewrites the entire on-disk state from the `external_banners` table:

```
/etc/hermes/body_milter/banners/banner_by_recipient_domain
    <recipient_domain>\t<option>
    _default\t<option>          special key, system-wide fallback

/etc/hermes/body_milter/banners/files/<option>/
    body.txt          plain-text banner (auto-derived at save)
    body.html         pre-rendered html banner
    position          "prepend" or "append" sidecar file
    images/           per-banner inline images (#230 cid pattern)
        1.png
        2.jpg
        ...
```

Where `<option>` is:

- `banner_default` for the system-wide row (NULL `recipient_domain`)
- `banner_<sanitized_recipient_domain>` for per-domain overrides (non-alphanumeric characters replaced with `_`)

The `files/` subdirectory is wiped on every regen (per-banner subdirs deleted recursively; the `.gitkeep` is preserved). This guarantees deleted rows and renamed scopes never leave stale files behind.

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

Plain-text part

The HTML body stored in `external_banners.body_html` is rendered server-side from the chosen template. The plain-text counterpart in `body_text` is auto-derived at save time:

- `<br>` becomes a newline
- `</p>`, `</li>`, `</tr>`, `</td>`, `</div>` become newlines
- All remaining tags are stripped
- Runs of 3+ newlines collapse to 2

The plain-text version is shipped to recipients viewing the message as `text/plain`. Inline images are omitted from the plain-text part — data URLs don't translate to text and recipients in text mode see the banner copy without image markers.

Inline images (#230)

The banner modifier inherits the `#230` cid inline-image pattern from Disclaimers. If a template's HTML contains `<img src="cid:banner_<option>_img_<N>">` references, the body milter:

1. Loads matching `images/<N>.<ext>` files from the option directory.
2. Attaches each as an `image/<format>` MIME part with `Content-ID: <banner_..._img_N>` and `Content-Disposition: inline`.
3. Wraps the message as `multipart/related` so MUAs resolve cid references against the inline parts.

The cid prefix is `banner_` so banner images cannot collide with `disclaimer_` or `signature_` cids inside the same composed message (the three modifiers can all add images to the same outbound; namespacing keeps them separate).

The bundled templates do not currently use inline images — banners are pure text. The infrastructure is present for future template additions (logo, warning icon, etc.).

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 banner, no delivery outage)
- Map file unreadable -> no entries match -> all mail flows unmodified
- Per-option files missing -> log + skip the modify -> mail flows unmodified
- MIME parse exception -> caught and logged -> mail flows unmodified
- Modifier raises any other exception -> caught and logged -> mail flows unmodified

In every failure case, mail keeps flowing. Worst case is a missed banner, 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.

Disabled rows

Rows with `enabled = 0` are skipped entirely during regen — no files written, no map entry. The milter never matches that scope until the row is re-enabled. Useful for staging copy changes before going live (build the new row disabled, preview it on `edit_external_banner.cfm`, flip the switch when ready).

Schema

```
CREATE TABLE IF NOT EXISTS external_banners (
  id               int(11)         NOT NULL AUTO_INCREMENT,
  recipient_domain varchar(255)    DEFAULT NULL,                -- NULL = system default
  template_key     varchar(64)     NOT NULL DEFAULT 'warning_yellow',
  fields_json      longtext        DEFAULT NULL,                 -- form values for re-edit
  body_text        longtext        DEFAULT NULL,                 -- auto-derived plain text
  body_html        longtext        NOT NULL,                     -- pre-rendered html
  position         enum('prepend','append') NOT NULL DEFAULT 'prepend',
  enabled          tinyint(3)      NOT NULL DEFAULT 1,
  updated_at       timestamp       NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp(),
  PRIMARY KEY (id),
  UNIQUE KEY uk_recipient_domain (recipient_domain)
);
```

The `UNIQUE KEY` on `recipient_domain` ensures only one row per recipient domain (and at most one system-default row where `recipient_domain IS NULL`). The `fields_json` blob stores the original form values so reopening the editor restores exactly what the admin typed; `body_html` is the rendered output the milter actually ships.

Verifying it works

The banner appears in the message body, so the easiest verification is to send an inbound message from an external account to a local mailbox and view the result in any MUA (webmail,

Outlook, Apple Mail). Beyond that:

- **Body milter logs** — the modifier logs `external_banner applied: option=<name> position=<prepend|append> plain=<n> html=<n>` per modified message. Surface with `docker logs hermes_body_milter` or via System Logs.
- **`Authentication-Results:` header** is preserved from upstream and visible in the recipient's "view source"; this confirms OpenDKIM ran **before** the banner.
- **`ARC-Seal: ... cv=fail`** in the outgoing message confirms the body was modified after the upstream chain — expected behavior, cross-ref ARC Settings.

Related

- Disclaimers — the outbound counterpart; same `hermes_body_milter` container, parallel design (sender-keyed instead of recipient-keyed)
- Organizational Signatures — second outbound modifier in the same container, with per-recipient resolution
- ARC Settings — full explanation of `cv=fail` after body modification and the Hermes-as-auth-boundary model
- DKIM Settings, DMARC Settings — upstream-verdict context preserved in `Authentication-Results`
- Domains — local mailbox-hosting domains drive the per-domain dropdown on `edit_external_banner.cfm`
- System Logs — surface the body-milter log stream for troubleshooting

---

Revision #48
Created 2026-05-31 12:52:41 UTC by Dino Edwards
Updated 2026-06-20 13:33:22 UTC by Dino Edwards