BCC Maps

BCC Maps

Admin path: **Content Checks > BCC Maps** (`view_bcc_maps.cfm`, `inc/add_bcc_map_action.cfm`, `inc/edit_bcc_map_action.cfm`, `inc/delete_bcc_map_action.cfm`, `inc/get_bcc_map_json.cfm`, `inc/get_mailbox_bcc_count.cfm`).

This page manages **silent message copies** at the SMTP envelope layer. Each entry maps an envelope address (sender **or** recipient, chosen per row) to a BCC target; when mail matching the address flows through Postfix, an additional copy is generated and routed to the target. The original delivery is unaffected; neither the original sender nor the original recipient sees any indication that a copy was made.

BCC Maps is the sibling envelope-level rule table to Global Sender Rules. Where Global Sender Rules decide whether a message is allowed in or blocked, BCC Maps decides whether an additional copy is created — both work on the envelope, before the message body is parsed.

How Postfix BCC works

Postfix has two distinct directives for envelope-level BCC injection:

Directive	Lookup key	Adds BCC when...	Typical use
`sender_bcc_maps`	Envelope **sender** (`MAIL FROM`)	The matched address is the one sending the message	Journaling outbound mail from an executive, monitoring a compromised account
`recipient_bcc_maps`	Envelope **recipient** (`RCPT TO`)	The matched address is the one receiving the message	Compliance journaling of mail to a regulated mailbox, legal-hold copies

The two maps are queried independently on every message — a single delivery can hit both if both a sender BCC and a recipient BCC match. The BCC happens once Postfix has accepted the message; the original envelope is preserved and the additional copy is queued separately.

Hermes wires both directives to MySQL-backed lookup tables in `/etc/postfix/main.cf`:

```
sender_bcc_maps    = mysql:/etc/postfix/mysql-sender-bcc-maps.cf
recipient_bcc_maps = mysql:/etc/postfix/mysql-recipient-bcc-maps.cf
```

Each `.cf` file holds a SQL query that selects `bcc_to` from `bcc_maps` where the address column matches and the row is enabled.

```
-- mysql-sender-bcc-maps.cf
SELECT bcc_to FROM bcc_maps
WHERE address='%s' AND bcc_type='sender' AND enabled=1

-- mysql-recipient-bcc-maps.cf
SELECT bcc_to FROM bcc_maps
WHERE address='%s' AND bcc_type='recipient' AND enabled=1
```

> **No reload required.** Unlike hashed `check_sender_access` lookups (used by Global Sender Rules), MySQL lookups are evaluated live against the database on every message — there is no `postmap` step, no `postfix reload`. Adding, editing, disabling, or deleting a row takes effect on the **next inbound message**. The UI surfaces this implicitly: the success alerts say "entry created/updated/deleted" without the "Postfix reloaded and Amavis restarted" suffix that other envelope pages append.

The page

A single info callout, an Add button that opens a modal, and one DataTable.

Add BCC Map modal

Field	Stored as	Notes
Address	`bcc_maps.address`	The envelope address to watch. Full email (`user@domain.tld`) or `@domain.tld` for domain-wide. Lower-cased on save
Type	`bcc_maps.bcc_type`	`sender` (outbound mail from this address) or `recipient` (inbound mail to this address)

Field	Stored as	Notes
BCC To	`bcc_maps.bcc_to`	The address that receives the silent copy. Single email only; not a pattern. Lower-cased on save
Description	`bcc_maps.description`	Free-text label (e.g. "Legal compliance — exec journaling"); nullable

The handler validates Address against `IsValid("email", ...)` for full addresses and against a `@domain` pattern check for domain-wide rows. BCC To must be a valid email address — domain patterns are not accepted here, only a concrete delivery target. The `(address, bcc_type)` pair is `UNIQUE` in the schema, so attempting to add a second row with the same address and type returns alert `m = 14` and rejects the insert.

BCC Maps (DataTable)

Column	Source
Actions	Edit (modal, AJAX load via `get_bcc_map_json.cfm`), Delete (confirm modal)
Address	`bcc_maps.address`
Type	`bcc_maps.bcc_type` -> Sender badge (primary) or Recipient badge (info)
BCC To	`bcc_maps.bcc_to`
Status	`bcc_maps.enabled` -> Enabled badge (green) or Disabled badge (grey)
Description	`bcc_maps.description` (em-dash if empty)

Edit constraints

The Edit modal makes **Address** and **Type** read-only — they are the natural key of the row (`UNIQUE (address, bcc_type)`) and changing them would semantically be a different rule. To re-target a watched address, delete the row and add a new one. Only **BCC To**, **Status** (enabled / disabled), and **Description** can be changed in place.

The Status toggle is the right tool for pausing surveillance briefly without losing the row — e.g. a compliance journaling rule that should be off during a planned mail-flow test.

The `bcc_maps` table

Column	Purpose
`id`	Auto-increment primary key
`address`	The watched envelope address (full email or `@domain.tld`)
`bcc_to`	The silent-copy target address
`bcc_type`	`sender` or `recipient`
`enabled`	`1` = active, `0` = paused (row preserved, no BCC generated)
`description`	Optional free-text label
`created_at`	Auto-populated timestamp on insert
`UNIQUE KEY`	`(address, bcc_type)` — same address can have one sender BCC AND one recipient BCC, but not two of either

BCC mail still goes through content filtering

Important behavior to understand: the BCC copy that Postfix generates is a real message in its own right, with the BCC target as its recipient. That copy traverses the same pipeline as any other inbound delivery — it goes through Amavis, SpamAssassin, ClamAV, the Sender/Recipient Rules for the BCC target, and any per-recipient quarantine policy.

The consequences:

- **A clean original can produce a quarantined BCC.** If the BCC target's spam threshold is stricter than the original recipient's, or if a recipient rule rejects the BCC sender, the silent copy can be quarantined or dropped while the original delivers normally.
- **A clean original can produce a bounced BCC.** If the BCC target is on an external server, that server's SPF / DMARC / receiver policy will be evaluated against the original sender's domain (which almost certainly does not authorise Hermes's IP). The external server may reject the BCC even though the original sender has nothing to do with the relay.
- **The BCC failure is silent to the original sender.** Postfix generated the BCC after accepting the original message; the original sender's SMTP transaction has already closed successfully. Any bounce of the BCC goes to the BCC target's `MAIL FROM` (typically the original sender, depending on `bounce_size_limit`) or to a double-bounce mailbox, but never causes the original delivery to fail.

The page's info callout flags the SPF case explicitly. For a journaling / compliance use case where loss of a copy is unacceptable, the BCC target should be a **local mailbox** on the same Hermes

instance — the message stays inside the gateway, the external-receiver policy issue does not arise, and any spam-tier issue is visible to the local mailbox owner.

Privacy and compliance

BCC Maps is a surveillance feature. The original sender and the original recipient are never notified that a copy was made; that is the point.

Operationally that means:

- **Auditability.** Each row carries a `created_at` timestamp; the `description` column is intended for the policy reference that justifies the watch (regulatory citation, ticket number, legal-hold matter ID). Filling it in is strongly recommended for any rule that is not strictly self-explanatory.
- **GDPR / employee-monitoring regimes.** In jurisdictions that require explicit employee notification of mail surveillance (EU member states, several US states for employee monitoring of personal communication), the existence of these rules must be disclosed in the employee privacy notice. Hermes does not generate that notice — the operator is responsible for the legal compliance wrapping around any active row.
- **Access control.** The page is only available to authenticated admins under `/admin/2/`. There is no end-user surface for BCC maps; mailbox owners cannot see whether their address is watched.

Cascading delete on mailbox removal

When a mailbox is deleted from Mailboxes, `inc/delete_mailbox_action.cfm` (step 4b) issues:

```
DELETE FROM bcc_maps
 WHERE address = :deleted_mailbox
    OR bcc_to  = :deleted_mailbox
```

That is — every BCC rule referencing the deleted mailbox is removed, whether the mailbox was the watched address or the BCC target. Because the live MySQL lookup re-reads on every message, the change takes effect immediately; no postmap or reload runs.

The same delete handler calls the AJAX endpoint `inc/get_mailbox_bcc_count.cfm` from the confirmation modal **before** the deletion fires, so the admin sees the number of BCC rows that will be cascaded ("This mailbox is watched by 2 BCC rules and is the target of 1 BCC rule") and can

cancel.

Domain-pattern rows (`@domain.tld`) are **not** cascaded by mailbox deletion — they reference a domain, not a specific mailbox, and remain in place until the whole domain is removed or the row is deleted manually.

Failure semantics

Alert	Trigger
`m = 1` / `2` / `3`	Add / Edit / Delete success
`m = 10`	Address field blank on Add
`m = 11`	Address fails email-or-`@domain` syntax check
`m = 12`	BCC To blank on Add or Edit
`m = 13`	BCC To is not a valid email address
`m = 14`	An entry with the same `(address, bcc_type)` already exists
`m = 20`	Missing required form field on Edit / Delete (no `bcc_id`)
`m = 21`	Edit / Delete target row no longer exists

There is no `session.m = 4` "Apply Failed" path because there is nothing to apply — the next message Postfix processes will read the new row from MySQL directly.

Files and containers touched

Path	Owner	Role
`config/hermes/var/www/html/admin/2/view_bcc_maps.cfm`	`hermes_commandbox`	The page
`config/hermes/var/www/html/admin/2/inc/add_bcc_map_action.cfm`	`hermes_commandbox`	Validate + INSERT
`config/hermes/var/www/html/admin/2/inc/edit_bcc_map_action.cfm`	`hermes_commandbox`	Validate + UPDATE (only `bcc_to`, `enabled`, `description`)
`config/hermes/var/www/html/admin/2/inc/delete_bcc_map_action.cfm`	`hermes_commandbox`	DELETE single row
`config/hermes/var/www/html/admin/2/inc/get_bcc_map_json.cfm`	`hermes_commandbox`	AJAX endpoint for the Edit modal
`config/hermes/var/www/html/admin/2/inc/get_mailbox_bcc_count.cfm`	`hermes_commandbox`	AJAX endpoint for the mailbox-delete confirmation modal

Path	Owner	Role
`config/postfix-dkim/etc/postfix/mysql-sender-bcc-maps.cf`	`hermes_postfix_dkim`	MySQL lookup definition for `sender_bcc_maps`
`config/postfix-dkim/etc/postfix/mysql-recipient-bcc-maps.cf`	`hermes_postfix_dkim`	MySQL lookup definition for `recipient_bcc_maps`
`bcc_maps` table	`hermes_db_server` (`hermes` DB)	Source of truth
`hermes_postfix_dkim` container	—	Reads MySQL lookups live on every message

Related

- Global Sender Rules — sibling envelope-level rule table; allow/block decisions rather than copy generation
- Sender/Recipient Rules — the per-pair table that the BCC copy will also pass through on its way to the BCC target
- Mailboxes — deleting a mailbox cascades the cleanup of any BCC rows referencing it; the confirmation modal surfaces the count before the deletion
- Perimeter Checks — sibling Content Checks page; envelope-time rejects that fire before any BCC is generated
- Anti-Spam Settings / Anti-Virus Settings — the content-filter tier that BCC copies traverse alongside the original message
- Message History — both the original and the BCC copy appear as separate entries in the message log
- System Logs — Postfix's `mail.log` records BCC generation as standard delivery lines, one per copy
- Mail Queue — a deferred BCC (external target rejecting on SPF, for example) sits in the queue here for inspection

---

Revision #8
Created 2026-05-31 12:52:21 UTC by Dino Edwards
Updated 2026-05-31 14:01:18 UTC by Dino Edwards