SVF Policies

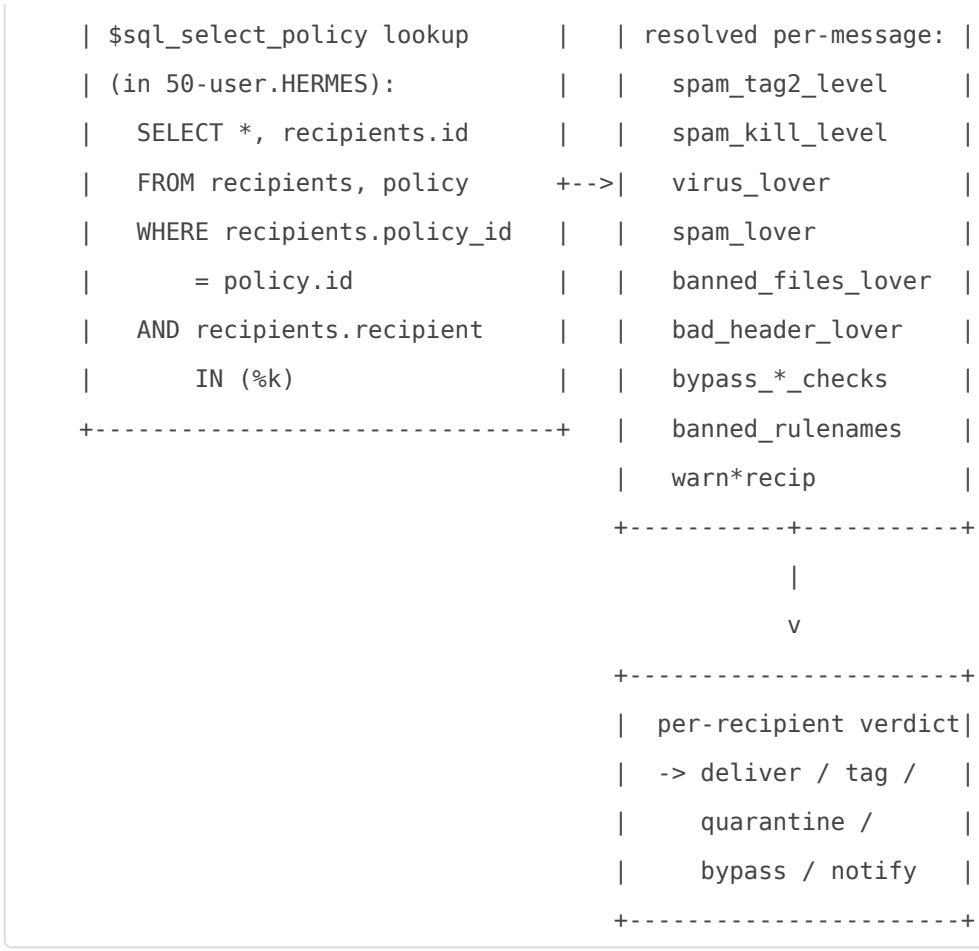
SVF Policies

Admin path: **Content Checks > SVF Policies** (`view_svf_policies.cfm`, `inc/get_svf_policies.cfm`, `inc/update_amavis_config_files.cfm`, `inc/restart_amavis.cfm`).

This page manages the **SVF (Spam / Virus / File) policies** that Amavis applies on a per-recipient basis. Each policy bundles four groups of decisions -- spam scoring thresholds, a banned-file ruleset name, four "accept" toggles (deliver instead of quarantine on virus / spam / banned-file / bad-header), four "bypass" toggles (skip the corresponding scan entirely), and three recipient notification toggles. When a message arrives, Amavis looks up the recipient in the `recipients` table, joins to the `policy` table on `policy_id`, and uses that policy's row to drive every per-message decision -- including which File Rule to enforce for attachments.

SVF policies are how the gateway expresses "marketing tolerates more spam than legal does," "abuse@ has to receive raw spam samples," or "this VIP mailbox skips banned-file checks because they trade `.iso` images legitimately." The global engine settings on Anti-Spam Settings and the per-rule weights on Score Overrides decide **how a message is scored**; the SVF policy assigned to the recipient decides **what happens to that score**.

Where SVF Policies sits



The recipient lookup is the policy resolver. Every recipient in the `recipients` table has a `policy_id` pointing at a row in the `policy` table; the `spam_policies` table is a thin index that adds `system` / `custom` / `default_policy` flags on top. A recipient with no matching row falls back to the **default policy** (`spam_policies.default_policy = '1'`) -- the page enforces that exactly one default exists at all times.

What's actually in a policy

The `policy` table is the Amavis-shaped row; only the columns the UI exposes are documented here (`policy` has additional NULL columns inherited from Amavis's reference schema that this page doesn't touch).

Field	DB column	Effect
Policy Name	`policy.policy_name` + `spam_policies.policy_name`	Display name; visible in the recipient dropdown on Relay Recipients and Mailbox Recipients. Up to 32 chars; letters, numbers, spaces, underscores, hyphens, `@`, and periods only

Field	DB column	Effect
Spam Tag Score	`policy.spam_tag2_level`	The Amavis `$spam_tag2_level` -- the score at which the spam header is added to the message (e.g. `X-Spam-Status: Yes`). Below this the message is delivered without a spam header. Range `-999 .. 999`
Spam Quarantine Score	`policy.spam_kill_level`	The Amavis `$spam_kill_level` -- the score at which the message is quarantined (or bounced, depending on `final_spam_destiny` on Anti-Spam Settings). Below this but above tag, the message is delivered with a spam header. Range `-999 .. 999`
File Rule	`policy.banned_rulenames`	The name of a File Rule (from `file_rule_components.rule_name`) -- Amavis maps this to the `@banned_filename_re` ruleset emitted into `50-user` and applies that ruleset's allow / ban regex to every attachment for this policy's recipients
Accept Viruses	`policy.virus_lover` (`Y` / `N`)	When `Y`, virus-flagged messages are **delivered** (with a notation) instead of quarantined. Almost always `N`; exists for forensic mailboxes
Accept Spam	`policy.spam_lover`	When `Y`, spam-flagged messages are delivered instead of quarantined. Useful for abuse / postmaster mailboxes that need to see the raw spam
Accept Banned Files	`policy.banned_files_lover`	When `Y`, messages with banned attachments are delivered instead of quarantined
Accept Bad Headers	`policy.bad_header_lover`	When `Y`, messages with malformed headers (per RFC) are delivered instead of quarantined
Bypass Virus Checks	`policy.bypass_virus_checks`	When `Y`, **skip ClamAV entirely** for this policy's recipients. No scan happens; no virus score contributes
Bypass Spam Checks	`policy.bypass_spam_checks`	When `Y`, skip SpamAssassin entirely. No score; no rule contributions; no Bayes update
Bypass Banned Checks	`policy.bypass_banned_checks`	When `Y`, skip banned-extension matching. Attachments are not screened against any File Rule
Bypass Header Checks	`policy.bypass_header_checks`	When `Y`, skip bad-header detection. Malformed-header messages pass through

Field	DB column	Effect
Notify on Banned File	`policy.warnbannedrecip`	When `Y`, the recipient receives an Amavis notification when a banned-file message is quarantined for them
Notify on Virus	`policy.warnvirusrecip`	Same, for virus quarantines
Notify on Bad Header	`policy.warnbadhrecip`	Same, for bad-header quarantines

`policy.spam_modifies_subj` is fixed to `Y` on add (the checkbox-equivalent isn't on the UI), which lets the subject tag configured on Anti-Spam Settings prepend to messages between tag and quarantine scores.

> **Operational consequence -- Accept vs Bypass.** "Accept" still runs the check; the message is just delivered when it fires. "Bypass" doesn't run the check at all. Use Bypass when the recipient must not pay the scan cost (e.g. high-volume automated relay) and Accept when the recipient must see the message but also wants the verdict header for downstream filtering (e.g. a SIEM mailbox or a mailbox that runs its own filtering on the spam header).

> **Operational consequence -- Bypass disables the verdict entirely.** Bypass Virus Checks means the message is never scanned by ClamAV; a virus reaching that recipient is not caught downstream by anything else in Hermes. Combine Bypass with a recipient-specific compensating control (e.g. quarantine at the destination mail server) or use Accept instead.

System vs custom vs default policies

Three orthogonal flags on `spam_policies`:

Flag	Stored as	Effect
`system`	`spam_policies.system = '1'`	Ships with the install. Cannot be deleted from the UI. Five system policies are seeded: `No Antispam & No Antivirus`, `Antispam & Antivirus`, `Antispam Only`, `Antivirus Only`, `Default`

Flag	Stored as	Effect
`custom`	`spam_policies.custom = '1'`	Created by an operator on this page (or via Copy of a system policy). Can be renamed, edited, deleted (unless default or assigned -- see below)
`default_policy`	`spam_policies.default_policy = '1'`	The policy applied to any recipient whose `recipients.policy_id` does not resolve. **Exactly one** row in `spam_policies` has this flag; the edit handler toggles it atomically by setting every row to `2` then the target row to `1`

The DataTable badges each row Yes/No for System and Default so the operator sees the flags at a glance. System rows lose their delete checkbox; the default row's "Default Policy" select is read-only in the edit modal with a hint to "set another policy as the default instead."

The page

A Page Guide callout, a collapsible **Add SVF Policy** card, and a DataTable of every existing policy (system + custom merged) with per-row Edit, Copy, and Delete actions.

Add SVF Policy card

A single form covering all four sections (basic + Accept + Bypass + Notifications). On submit:

1. Validates `policy_name` non-blank, character-safe, and not a duplicate
2. Validates `spam_tag2_level` and `spam_kill_level` as floats in `-999 .. 999`
3. Validates `banned_rulenames` (File Rule) non-blank
4. INSERTs into `policy` (with `spam_tag_level` hardcoded to `-999` and `spam_modifies_subj = 'Y'`)
5. INSERTs into `spam_policies` with `custom = '1'`, `system = '2'`, `default_policy = '2'` and `policy_id = <new policy.id>`
6. Runs the Amavis apply chain (see Save and apply flow below)

The Copy action duplicates an existing policy under the name `Copy of <original>` (with a date-time suffix if that name is already taken). Useful for branching a system policy into a custom variant without re-keying every toggle.

SVF Policies DataTable

Column	Source

(checkbox)	Selection for bulk Delete Selected. Disabled with a hover tooltip on system rows
Policy Name	`spam_policies.policy_name`
System	Yes/No badge driven by `spam_policies.system`
Default	Yes/No badge driven by `spam_policies.default_policy`
Spam Tag	`policy.spam_tag2_level`
Spam Quarantine	`policy.spam_kill_level`
File Rule	`policy.banned_rulenames`
Actions	Edit, Copy, Delete (Delete hidden on system rows)

Edit reuses the same validation as Add. Renaming a policy propagates the new name into `spam_policies.policy_name` in the same UPDATE.

Deletion guards

A custom policy can only be deleted when all three guards pass:

Guard	Source	Alert
Not a system policy	`spam_policies.system <> '1'`	`m = 10` -- "System policies cannot be deleted"
Not the default policy	`spam_policies.default_policy <> '1'`	`m = 11` -- "The default policy cannot be deleted. Set another policy as the default first"
Not assigned to any recipient	`recipients.policy_id <> :id`	`m = 12` -- "This policy is assigned to the following recipient(s): . Assign them to a different policy first"

Single delete reports the specific failure; bulk delete silently skips guarded rows and reports a per-batch count via `m = 13` ("No policies were deleted") if zero deletes succeeded. The list of blocking recipients is surfaced in the single-delete failure alert so the operator can see exactly which entries need to be reassigned on Relay Recipients or Mailbox Recipients first.

Save and apply flow

```
1. View page submits action="add_policy" | "edit_policy" |
   "copy_policy" | "delete_policy" | "bulk_delete"
2. Action handler validates input, runs deletion guards,
```

```
   INSERTs / UPDATEs / DELETEs on the policy + spam_policies tables
3. cfinclude update_amavis_config_files.cfm:
     - Read /opt/hermes/conf_files/50-user.HERMES
     - Substitute SERVER-NAME, SERVER-DOMAIN, sa-spam-subject-tag,
       final-{virus,banned,spam,bad-header}-destiny,
       enable-dkim-{verification,signing},
       HERMES-USERNAME, HERMES-PASSWORD,
       FILE-RULES-GO-HERE (from file_rule_components table),
       DKIM-KEYS-GO-HERE (from dkim_sign table)
     - Back up /etc/amavis/conf.d/50-user -> 50-user.HERMES.BACKUP
     - Move rendered file into place
4. cfinclude restart_amavis.cfm:
     docker container restart hermes_mail_filter
5. session.m = 1|2|3|5 -> green alert ("Policy Added" / "Updated"
   / "Deleted" / "Copied")
6. cflocation back to view_svf_policies.cfm
```

A few important things about this chain:

- **The policy table is not substituted into `50-user`.** Amavis reads it live at scan time via the `$sql_select_policy` SQL lookup defined in `50-user.HERMES`. The save-and-apply chain still re-renders `50-user` to refresh the static placeholders (file rules, DKIM keys, destinies) -- but the SVF policy itself is picked up by the next message after the UPDATE commits, no Amavis restart strictly required for the policy change alone. The restart is there to make the operation atomic with any other config that might have drifted, and to surface a clear green alert.
- **Copy does not restart Amavis.** It only INSERTs and sets `m = 5`; the new policy doesn't affect Amavis until it's assigned to a recipient (and `recipients` changes don't go through this page).
- **The whole chain is wrapped in `cftry`/`cfcatch`.** If the update or restart fails, the policy rows are already committed but the operator sees `m = 40` ("Policy was saved but Amavis configuration update or reload failed") instead of the green alert. A subsequent successful save on any page that triggers the same chain re-renders correctly.

Failure semantics

Alert	Trigger
`m = 1`	Add Policy succeeded; Amavis updated and reloaded
`m = 2`	Edit Policy succeeded; Amavis updated and reloaded

Alert	Trigger
`m = 3`	Delete Policy (single or bulk with at least one success) succeeded
`m = 5`	Copy Policy succeeded (no Amavis restart -- new copy is unassigned)
`m = 10`	Single delete refused: system policy
`m = 11`	Single delete refused: default policy
`m = 12`	Single delete refused: policy assigned to recipient(s) -- recipient list surfaced
`m = 13`	Bulk delete completed with zero successes (every row was protected)
`m = 30`	Policy name empty
`m = 31`	Policy name has invalid characters
`m = 32`	Policy name duplicates an existing policy
`m = 33`	Spam Tag Score empty or non-numeric
`m = 34`	Spam Tag Score outside `-999 .. 999`
`m = 35`	Spam Quarantine Score empty or non-numeric
`m = 36`	Spam Quarantine Score outside `-999 .. 999`
`m = 37`	File Rule not selected
`m = 38`	Copy: source policy not found
`m = 40`	Save succeeded but Amavis apply chain threw

Recipient assignment

SVF policies are bound to recipients on the **Email Relay > Recipients** page (`view_internal_recipients.cfm`) and the **Email Server > Mailboxes** page (`view_mailboxes.cfm`). Each page exposes a Policy dropdown populated from `spam_policies` . Assigning a policy writes the matching `policy.id` into `recipients.policy_id` , and Amavis picks it up on the next message to that recipient.

A recipient row with `policy_id` pointing at a row that no longer exists falls through to the default policy at scan time -- this is the same fall-through as a recipient with no row in the `recipients` table at all. The deletion guard on this page (which refuses delete while any recipient still references the policy) is the front-line defence against accidentally creating that fall-through.

Files and containers touched

Path	Owner	Role
`config/hermes/var/www/html/admin/2/view_svf_policies.cfm`	`hermes_commandbox`	The page (validation + Add / Edit / Copy / Delete / Bulk Delete)
`config/hermes/var/www/html/admin/2/inc/get_svf_policies.cfm`	`hermes_commandbox`	Loads system, custom, and combined policy lists plus the file-rule dropdown
`config/hermes/var/www/html/admin/2/inc/update_amavis_config_files.cfm`	`hermes_commandbox`	Renders `50-user` from template + DB (file rules, DKIM keys, destinies)
`config/hermes/var/www/html/admin/2/inc/restart_amavis.cfm`	`hermes_commandbox`	`docker container restart hermes_mail_filter`
`config/hermes/opt/hermes/conf_files/50-user.HERMES`	template (read) -> `hermes_mail_filter` (live `/etc/amavis/conf.d/50-user`)	Holds `$sql_select_policy` which Amavis uses to resolve a recipient to a policy row at scan time
`/etc/amavis/conf.d/50-user.HERMES.BACKUP`	`hermes_mail_filter`	Pre-write backup of the prior live `50-user`, refreshed each save
`policy` table	`hermes_db_server` (`hermes` DB)	Amavis-shape policy row -- the source of truth for every per-recipient verdict
`spam_policies` table	`hermes_db_server`	Thin index over `policy` with `system` / `custom` / `default_policy` flags
`recipients` table	`hermes_db_server`	`recipients.policy_id` is the foreign key Amavis joins on at scan time; the assignment is managed by Relay Recipients and Mailboxes pages
`file_rule_components` table	`hermes_db_server`	Source of the File Rule dropdown -- `policy.banned_rulenames` stores the chosen rule name
`hermes_mail_filter` container	--	Hosts Amavis; restarted on add / edit / delete; reads `policy` directly per-message at scan time

Related

- Anti-Spam Settings -- engine-wide toggles and the `final_*_destiny` quarantine actions. The SVF policy decides whether a message clears the tag/quarantine threshold; Anti-Spam Settings decides what Amavis does with a quarantine verdict (DSN or silent)
- Antivirus Settings -- ClamAV runs in the same Amavis pass that consults the SVF policy. Bypass Virus Checks on a policy turns off ClamAV for that recipient entirely
- Score Overrides -- per-rule SpamAssassin weights; tunes the contributions that add up to the final score the SVF policy thresholds compare against

- Message Rules -- custom SpamAssassin rules whose scores contribute to the same final score
- File Rules -- bundles File Extensions and File Expressions into the named ruleset selected by `policy.banned_rulenames` on this page
- File Extensions -- the catalogue that feeds File Rules; an extension is only enforced when its ruleset is bound to a policy here
- Malware Feeds -- ClamAV signature feeds; a policy with Bypass Virus Checks `Y` bypasses every signature the feeds ship
- Perimeter Checks -- rejection at SMTP-time pre-empts every SVF policy decision; the policy only applies to mail that clears the perimeter
- Message History -- a quarantined message records the recipient and the verdict; cross-referencing the recipient to its policy here explains why that verdict fired
- Email flow -- the full pipeline showing where SVF policy lookup happens

---

Revision #8
Created 2026-05-31 12:52:33 UTC by Dino Edwards
Updated 2026-05-31 14:01:25 UTC by Dino Edwards