SAN Management

SAN Management

Admin path: **Email Server > SAN Management** (`view_mailbox_sans.cfm`, `inc/san_actions.cfm`, `inc/sync_mailbox_sans.cfm`, `inc/acme_request_san_certificate.cfm`, `inc/smtp_sni_generate_config.cfm`, `inc/generate_nginx_configuration.cfm`, `schedule/acme_validate_ip.cfm`).

This page maintains the **global list of SAN (Subject Alternative Name) prefixes** that Hermes cross-joins with every mailbox-hosting domain to produce the actual SANs on each domain's TLS certificate. The prefix `mail` plus the domain `example.com` produces the SAN `mail.example.com`; doing it once here lets Hermes mint one certificate per mailbox domain that covers IMAP/POP/Submission, autoconfig/autodiscover, ManageSieve, CalDAV/CardDAV, and any additional client-facing hostnames in a single cert.

Pairs tightly with System Certificates (the certificate store these SANs are stamped into) and Domains (the mailbox-domain rows the prefixes are multiplied against). This page is the **only** input UI for the mailbox-cert SAN list — both the CSR generator on System Certificates and the ACME SAN request path read from `additional_sans` to build the `-d` flag list.

What the page edits

```
additional_sans                          domains (type='mailbox')
+----+--------------+--------+           +----+----------------+
| id | san          | system |           | id | domain         |
+----+--------------+--------+           +----+----------------+
|  1 | autoconfig   |   1    |           |  9 | example.com    |
|  2 | autodiscover |   1    |           | 10 | acme.org       |
|  3 | mail         |   2    |           +----+----------------+
|  4 | imap         |   2    |
+----+--------------+--------+
             |                                    |
             +--- sync_mailbox_sans.cfm cross-joins ---+
                                  |
```

```
                    v
mailbox_sans  (one row per prefix x domain)
+----+-------------+-------------------------+------+------+------+
| id | certificate | subdomain               | ip   | dns  | acme |
+----+-------------+-------------------------+------+------+------+
| 50 |     12      | autoconfig.example.com  | YES  | YES  |  1   |
| 51 |     12      | autodiscover.example.com| YES  | YES  |  1   |
| 52 |     12      | mail.example.com        | YES  | YES  |  1   |
| 53 |     12      | imap.example.com        | NO   | NO   |  1   |
| 54 |     12      | autoconfig.acme.org     | YES  | YES  |  1   |
| ...
```

Two storage rows per change:

Table	Role
`additional_sans`	One row per global prefix. `san` is the subdomain label; `system` is `1` for installer-seeded prefixes (`autoconfig`, `autodiscover`) that cannot be deleted, `2` for admin-added prefixes. There is no `enabled` flag — the row's mere presence means active.
`mailbox_sans`	One row per `additional_sans.san` x `domains` (`type='mailbox'`) combination. Carries the cert FK (`certificate`), the full FQDN (`subdomain`), and the per-SAN validation state (`ip` / `dns` = `YES` / `NO`, plus `*_result_datetime`, `*_result_msg`). `acme = 1` for ACME-managed certs, `2` for imported certs.

The page itself only writes to `additional_sans`. The cross-join into `mailbox_sans` is performed by `sync_mailbox_sans.cfm`, which is also called from the Domains page on add/edit (so adding a new mailbox domain populates its SAN rows immediately).

How a prefix becomes a live SAN

```
form submit (Add SAN Prefix)  ──► san_actions.cfm
                                      |
                                      |  validate:
                                      |    - prefix not blank
                                      |    - matches ^[a-z][a-z0-9-]{0,62}$
                                      |      (DNS label rules: lowercase, starts
                                      |       with letter, <= 63 chars)
                                      |    - not already in additional_sans
```

```
                        |
                        |  INSERT additional_sans (san, system=2)
                        |
                        v
              sync_mailbox_sans.cfm
                  |
                  |  for each (prefix x mailbox-domain):
                  |     if FQDN missing in mailbox_sans:
                  |        INSERT (cert from mailbox_domains,
                  |                subdomain=fqdn, ip='NO', dns='NO',
                  |                acme=1|2 per cert type)
                  |     if FQDN exists with wrong cert binding:
                  |        UPDATE certificate + acme
                  |        (PRESERVE ip/dns validation state —
                  |         resetting would break nginx vhost
                  |         generation until the next validator
                  |         pass)
                  |  for each existing mailbox_sans row whose
                  |     subdomain is no longer in the cross-join:
                  |        DELETE
                  |
                  v
          Validator picks up the new rows on its next pass
          (schedule/acme_validate_ip.cfm @every 1h)
                  |
                  |  POST encrypted subdomain to
                  |    https://verify.hermesseg.io
                  |    -> returns expected IP for the host
                  |  Compare against the SAN's resolved A record
                  |    -> ip = YES/NO with timestamped result_msg
                  |  Resolve DNS for the SAN's CNAME/A chain
                  |    -> dns = YES/NO with timestamped result_msg
                  |
                  v
      All SANs on a cert at dns=YES + ip=YES?
                  |
                  v
acme_request_san_certificate.cfm (Pro)
docker run --rm certbot/certbot:latest \
  certonly --webroot --cert-name <domain> --expand \
```

```
                -d example.com -d autoconfig.example.com \
                -d autodiscover.example.com -d mail.example.com ...
                            |
                            v
            smtp_sni_generate_config.cfm   (Postfix SNI map)
            generate_nginx_configuration.cfm (per-SAN nginx vhosts)
```

Delete reverses the same path: removing a prefix from `additional_sans` calls `sync_mailbox_sans.cfm`, which deletes the corresponding `mailbox_sans` rows for every mailbox domain. The certificate itself is **not** re-issued automatically on delete — the next renewal cycle picks up the smaller SAN set when it runs.

The two seed prefixes

A fresh install seeds two `system = 1` rows:

Prefix	Required for
`autoconfig`	Thunderbird and K-9 Mail auto-configuration. Clients fetch `https://autoconfig.<domain>/mail/config-v1.1.xml`.
`autodiscover`	Outlook and iOS Mail auto-configuration. Clients POST to `https://autodiscover.<domain>/autodiscover/autodiscover.xml`.

Both rows have **Delete** suppressed and the System badge displayed. The action handler re-checks `system = 1` server-side and refuses with error 13 if a crafted POST tries to bypass the missing button. Removing either prefix would break client auto-discovery globally across every mailbox domain — they are non-optional.

Prefix validation rules

The Add form enforces DNS-label syntax both client-side (`pattern="[a-z][a-z0-9-]*"` + `maxlength="63"`) and server-side (`REFind("^[a-z][a-z0-9-]{0,62}$", ...)`):

- **Lowercase letters, numbers, and hyphens only.** No uppercase, no underscores, no dots. Each prefix is a **single** DNS label; multi-label SANs (`internal.mail.example.com`) are not supported here.
- **Must start with a letter.** Leading digits and leading hyphens are rejected per the DNS label spec.
- **Max 63 characters.** Each DNS label is capped at 63 octets.
- **Lowercased on save.** Submitting `Mail` stores as `mail`.

Suggested prefixes from the placeholder text: `mail`, `imap`, `smtp`, `pop`, `webmail`. Pick whichever match the client-facing hostnames you've published in DNS; the prefix only does work if a matching DNS A/CNAME record exists pointing at this server.

The Let's Encrypt budget callout

The page surfaces a live calculation of the cert budget per domain:

```
Let's Encrypt SAN limit: Each domain certificate supports a maximum
of 100 SANs. With <N> prefixes configured, each domain's certificate
uses <N + 1> SANs (1 for the domain + N prefixes), leaving room for
up to <99 - N> additional prefixes.
```

The +1 accounts for the bare domain itself, which is always included on the cert regardless of prefix list (this is hardcoded in the ACME request path).

Other Let's Encrypt rate limits that don't show on this page but still apply:

Limit	Value
SANs per certificate	100
Certificates per registered domain per week	50
Duplicate certificates per week	5
Failed validation attempts per account, per hostname, per hour	5

A misconfigured DNS record (SAN row stuck at `dns = NO`) does **not** burn the duplicate-cert budget because the certbot run is gated on the validator marking every SAN ready first. The validator's failed DNS probes are free and run on Hermes-side resolvers, not Let's Encrypt's.

Validation challenge mechanics

ACME issuance uses **HTTP-01** by default. The certbot container mounts `<repo>/config/hermes/var/www/html` at `/var/www/certbot` so the challenge file lands where the live nginx vhost for the domain already serves `/.well-known/acme-challenge/`. The domain's nginx vhost (generated by `generate_nginx_configuration.cfm`) is therefore required to be up and serving HTTP on port 80 of the public IP that the SAN resolves to.

DNS-01 (TXT-record validation) is **not** wired into this UI. The underlying certbot container supports it but the request path here hardcodes `--webroot`. Internal-only / DNS-only SANs (subdomains that

resolve to an internal IP but should still be on the public cert) need either a manual certbot invocation or a public split-DNS record pointing at the gateway's WAN address — there is no DNS-challenge bypass on this page.

The validator's `ip = YES` check is **separate from** the ACME challenge — it confirms that the SAN's DNS A record points at this gateway's expected IP (which is what `https://verify.hermesseg.io` returns when probed). It exists to catch broken DNS before burning a Let's Encrypt rate-limit slot, not to perform the ACME challenge itself.

How SAN status surfaces elsewhere

This page edits the prefix list; the per-SAN validation state and the per-cert SAN sub-table show up on other pages:

Where	What it shows
Domains **Cert Status** column	Per-domain aggregate: `Verified` (all SANs ip+dns=YES), `Partial`, `Awaiting Cert`, `Pending`, `DNS Failed`, `No SANs`, `No Cert`. Imported certs always render `Imported` regardless of probe state because probes are informational only for those.
System Certificates expanded row **Mailbox SAN Validation** sub-table	Per-cert listing: every SAN bound to the cert, with its `ip_result_msg` / `dns_result_msg` / timestamps. Read-only here.
System Certificates § Generate CSR — Mailbox certificate purpose	The CSR generator pre-fills the SAN list from `additional_sans` x the chosen mailbox domain. Refuses to generate a mailbox CSR if `additional_sans` is empty (impossible in practice because the two system prefixes can't be deleted).
`smtp_sni_generate_config.cfm` (run from Email Server > Settings)	Reads `mailbox_sans WHERE dns = 'YES'`, builds Postfix's `sni_maps`, runs `postmap -F`. Postfix then serves the per-domain cert on `:25` / `:587` via SNI based on the client's TLS SNI extension.
`generate_nginx_configuration.cfm` (run from Domains)	Reads validated `mailbox_sans` rows to write per-SAN nginx `server` blocks (autoconfig, autodiscover, DAV).

Failure semantics

What breaks	What happens

Prefix blank	`session.m = 10`, redirect, no DB write
Prefix fails DNS-label regex	`session.m = 11`, redirect, no DB write
Prefix already in `additional_sans`	`session.m = 12`, redirect, no DB write
Delete attempted on a `system = 1` prefix	`session.m = 13`, redirect, no DB write
Delete with non-numeric `delete_san_id`	`session.m = 20`, redirect
`sync_mailbox_sans.cfm` fails mid-cross-join	Partial `mailbox_sans` state possible; re-saving any mailbox domain or re-adding the same prefix triggers another sync that converges
Validator can't reach `verify.hermesseg.io`	`mailbox_sans.ip` stays at the previous value; cert request gated until next successful probe. Validator runs hourly.
`acme_request_san_certificate.cfm` fails (DNS, port 80, rate limit)	Postmaster email sent with certbot stderr; SAN rows retain validation state; admin can re-trigger by toggling the cert binding on Domains
`smtp_sni_generate_config.cfm` finds zero validated SANs	Deletes `/etc/postfix/sni_maps` and `.db` — Postfix falls back to its default cert on every connection. Non-fatal but clients lose per-domain SNI.

Files and containers touched

Path	Owner	Role
`config/hermes/var/www/html/admin/2/view_mailbox_sans.cfm`	`hermes_commandbox`	Page + Add card + Delete modal + LE budget callout
`config/hermes/var/www/html/admin/2/inc/san_actions.cfm`	`hermes_commandbox`	Add / Delete handler — validates, writes `additional_sans`, calls sync
`config/hermes/var/www/html/admin/2/inc/sync_mailbox_sans.cfm`	`hermes_commandbox`	Cross-joins prefixes x mailbox domains into `mailbox_sans`; idempotent
`config/hermes/var/www/html/admin/2/inc/acme_request_san_certificate.cfm`	`hermes_commandbox`	Pro — runs ephemeral certbot container for SAN-bearing certs
`config/hermes/var/www/html/admin/2/inc/smtp_sni_generate_config.cfm`	`hermes_commandbox`	Pro — builds Postfix `sni_maps` from validated SANs
`config/hermes/var/www/html/admin/2/inc/generate_nginx_configuration.cfm`	`hermes_commandbox`	Per-domain nginx vhost generator (called from Domains; consumes validated SANs)
`config/hermes/var/www/html/schedule/acme_validate_ip.cfm`	`hermes_commandbox` (Ofelia)	Pro — hourly validator; probes each SAN's IP via `verify.hermesseg.io` and updates `mailbox_sans.ip` / `dns`
`additional_sans` table	`hermes_db_server` (`hermes` DB)	The prefix list this page edits

Path	Owner	Role
`mailbox_sans` table	`hermes_db_server` (`hermes` DB)	Per-SAN rows with validation state and cert binding
`system_certificates` table	`hermes_db_server` (`hermes` DB)	Per-cert metadata referenced via `mailbox_sans.certificate`
`/etc/letsencrypt/live/<domain>/`	`hermes_commandbox` (bind-mounted from `config/certbot/conf/`)	Issued SAN certs
`/etc/postfix/sni_maps` + `.db`	`hermes_postfix_dkim` (mounted)	Live SNI map — Postfix serves per-domain cert based on this
`/etc/postfix/sni/*.pem`	`hermes_postfix_dkim` (mounted)	Combined key + fullchain PEM per cert, referenced from `sni_maps`
Per-SAN nginx vhost files	`hermes_nginx` (mounted)	One vhost per validated SAN
`certbot/certbot:latest` image	docker.io	Pulled on demand for SAN cert issuance + renewal
`verify.hermesseg.io`	external (Pro)	Returns expected IP for a given SAN to gate ACME issuance

Every certbot invocation is `docker run --rm` against the public `certbot/certbot:latest` image — same pattern as the single-domain ACME path on System Certificates. The container shares the host network (`--network host`) so the HTTP-01 challenge can reach port 80 on the public IP.

Related

- System Certificates — the certificate store these SANs land on. The Mailbox certificate purpose on Generate CSR auto-fills its SAN list from this page; Pro's auto-managed ACME path mints SAN certs from the same source.
- Domains — per-mailbox-domain Cert Status column summarizes the per-SAN validation state this page's prefixes drive. Adding a domain calls `sync_mailbox_sans.cfm`, so new SANs appear immediately under existing prefixes.
- Mailboxes — mailbox users hit IMAP/Submission via the `imap` / `mail` / `smtp` prefixes configured here. Apple iOS and Outlook reach autodiscover via the system prefixes.
- Settings — Dovecot IMAP/POP TLS is gated on the validated mailbox cert; the SNI map for Postfix is generated from the same `mailbox_sans` table this page populates.
- Aliases / Shared Mailboxes — both ride on the same per-domain cert; no separate SAN entries needed.
- SMTP TLS Settings — binds the **single** cert Postfix presents on the public SMTP banner. The SNI map this page feeds into is an **additional** layer that overrides the banner cert when the client sends a matching SNI hostname.
- Email Relay > Relay Recipients — relay recipients use Submission via the same `mail.<domain>` hostnames as local mailboxes; the SAN prefixes here cover both topologies.

Revision #14
Created 2026-05-31 12:52:16 UTC by Dino Edwards
Updated 2026-06-13 12:30:14 UTC by Dino Edwards