LDAP RemoteAuth

LDAP RemoteAuth

Pro Edition feature. Maps to **System > LDAP RemoteAuth** (`view_remoteauth.cfm`, `edit_remoteauth_mapping.cfm`).

RemoteAuth lets Hermes authenticate selected users against an **upstream LDAP or Active Directory** server instead of storing their password in Hermes's own OpenLDAP. The page configures the upstream-to-domain mapping, global TLS settings, a one-shot bind test, and the apply-to-LDAP sync. Active Directory, OpenLDAP, 389 Directory Server, and FreeIPA are all supported through the same plumbing.

What RemoteAuth is — and isn't

Is	Isn't
A pass-through bind: at web login, Hermes binds against the upstream DN with the supplied password and accepts or rejects accordingly	A directory sync. Hermes does not import users, groups, photos, or attributes from upstream.
Per-user opt-in, via `auth_type = 'remote'` + `remoteauth_domain` on the recipient/system-user row	A whole-installation toggle. Local-auth and remote-auth users coexist in the same directory and the same UI.
Implemented as an **OpenLDAP** `remoteauth` **overlay** in Hermes's `hermes_ldap` container	A reinvented bind proxy. The heavy lifting is `slapo-remoteauth(5)` against a stub user with a `seeAlso` pointer.
The credential path for **web login only** — `/users`, `/nc`, `/admin` (via Authelia → LDAP bind)	The credential path for **IMAP/SMTP/CalDAV/CardDAV**. Those continue to authenticate against Hermes-issued app passwords; see Credential Model for the full picture.

> **Operational consequence.** A remote-auth user's mail-client / DAV passwords still live in Hermes (`app_passwords` table, hashed). The upstream directory password is never exposed to Dovecot or Nextcloud DAV — only to the web gate. If the customer's IT team rotates the upstream password, the user's app passwords keep working until they are explicitly revoked. This is by design (see

> Credential Model § Local-auth users vs. remote-auth users).

How it works under the hood

```
Web login (/admin, /users, /nc)
        |
        ▼
   Authelia
        |  LDAP bind to Hermes OpenLDAP
        ▼
hermes_ldap  (slapd)
        |
        |  user entry has seeAlso=<upstream DN>
        |  user entry has associatedDomain=<mapping key>
        |
        ▼
slapo-remoteauth overlay
        |  matches associatedDomain → upstream server URI
        |  rewrites the bind to the seeAlso DN
        ▼
External AD / LDAP server  (customer's DC)
        |
        ▼  bind result returned up the chain
   Authelia decision: PASS or FAIL
```

The overlay is configured in `cn=config` on the `mdb` database. Hermes's CFML never bind-checks the upstream itself at login time — that is the overlay's job. The CFML only **writes** the overlay configuration when an admin clicks **Apply Settings**.

OpenLDAP remoteauth is a singleton overlay

This is the single most important constraint to understand when reasoning about why the page works the way it does.

Constraint	Consequence in the UI
`slapo-remoteauth` allows **only one overlay instance** per database	All mappings live inside the same overlay
`olcRemoteAuthMapping` is multi-valued **but has no equality matching rule**	You cannot `ldapmodify add` a single mapping to an existing overlay. The entire overlay must be rebuilt.
`olcRemoteAuthTLS` is a single string applied **to all mappings inside the overlay**	TLS settings (STARTTLS, certificate verification, CA cert path, retry count) are **global**, not per-mapping

`inc/ldap_remoteauth_sync_all.cfm` therefore implements **full replacement on every save**: delete the existing overlay, rebuild it from `remoteauth_mappings` + `remoteauth_settings`. There is no incremental update path. The page's pending-changes badge reflects this — every edit marks `ldap_synced = 0` on both tables, and **Apply Settings** flips it back to `1` only after the full rebuild succeeds.

Multiple upstream servers with different CAs

Because TLS is global, an installation that binds to multiple upstream LDAP servers signed by different CAs must upload a **concatenated CA bundle**:

```
cat dc01-ca.pem dc02-ca.pem dc03-ca.pem > ca-bundle.pem
```

The page accepts the bundle as-is in the **CA Certificate** file picker. OpenLDAP walks the bundle when validating any of the configured upstream servers.

Database schema

Two tables drive the page. Both are in the `hermes` database.

Table	Role
`remoteauth_settings`	Six rows, key/value: `enabled`, `tls_starttls`, `tls_reqcert`, `ca_cert_file`, `retry_count`, `ldap_synced`
`remoteauth_mappings`	One row per upstream-LDAP-to-domain mapping (`domain_name` UNIQUE, `server_address`, `server_port`, `remote_dn_pattern`, `description`, `enabled`, `ldap_synced`)

Two user-bearing tables carry RemoteAuth references:

Table	Columns	Role

`recipients`	`auth_type ENUM('local','remote')`, `remoteauth_domain VARCHAR(255)`	Relay recipients can be RemoteAuth-mode
`system_users`	`auth_type ENUM('local','remote')`, `remoteauth_domain VARCHAR(255)`	Console admins / reader users can be RemoteAuth-mode

The `mailboxes` table does **not** carry `auth_type` yet. RemoteAuth-for-mailboxes is planned but not yet wired (see Future work).

DN pattern placeholders

The `remote_dn_pattern` column stores the upstream DN with four substitutable tokens. Substitution happens in `inc/ldap_add_user_remoteauth.cfm` at user-create time, baked into the `seeAlso` attribute on the local stub entry.

Token	Source	Notes
`{username}`	Local part of email (`jsmith@company.com` → `jsmith`) — uses `ListFirst(..., "@")`. For console admins where the username has no `@`, the whole string is used.	Matches `sAMAccountName` / `uid` patterns
`{firstname}`	`givenName` field on the add form	Required if the DN pattern uses it
`{lastname}`	`sn` field on the add form	Required if the DN pattern uses it
`{email}`	Full email address as entered	Useful for `mail=` patterns

Common patterns the in-page help surfaces:

Directory type	Pattern
AD (display name as CN)	`cn={firstname} {lastname},ou=Users,dc=example,dc=com`
AD (sAMAccountName as CN)	`cn={username},ou=Users,dc=example,dc=com`
OpenLDAP / FreeIPA	`uid={username},ou=People,dc=example,dc=com`

The pattern must match the upstream's actual naming convention **exactly**. A wrong pattern produces `ldap_bind: Invalid DN syntax` or `Invalid credentials` at login time; use the **Test** button before saving to confirm.

The local stub entry

For each RemoteAuth user, Hermes creates a normal `inetOrgPerson + domainRelatedObject` entry in `ou=users,dc=hermes,dc=local` with **no** `userPassword` **attribute** and the two overlay-driving attributes

set:

```
dn: cn=jsmith,ou=users,dc=hermes,dc=local
objectClass: inetOrgPerson
objectClass: domainRelatedObject
givenName: John
sn: Smith
displayName: John Smith
mail: jsmith@company.com
uid: jsmith
seeAlso: cn=John Smith,ou=Users,dc=company,dc=com    <-- expanded from
{firstname}/{lastname}/etc.
associatedDomain: company                            <-- the mapping key
```

At bind time the overlay reads `associatedDomain`, looks up the matching `olcRemoteAuthMapping`, opens an LDAP connection to that upstream URI, and re-binds as `seeAlso` with the supplied password. The local entry has no password to validate against, so the overlay's decision is the only decision.

Test Connection button

The Test modal does **not** consult the saved settings end-to-end — it does its own `ldapwhoami` against the mapping's `server_address:server_port`, applying the same DN pattern substitution the overlay would and honoring the global STARTTLS setting. The credentials entered in the modal are used for one bind attempt:

```
docker exec hermes_ldap ldapwhoami -x -H ldap://<server>:<port> \
    -D "<DN expanded from pattern>" -w "<password>"  [-ZZ if STARTTLS]
```

Success is detected by `dn:` or `u:` in the response. Failure surfaces the raw stderr from `ldapwhoami`. The bind credentials are never stored — they live only for the duration of the request, then disappear.

This is intentionally **separate from the overlay flow**: it lets an admin verify the DN pattern and network path before clicking Apply Settings (which would rebuild the overlay and potentially break live logins).

DNS resolution prerequisite

The `hermes_ldap` container resolves hostnames through Hermes's own Unbound resolver — by default, public recursive DNS. **Internal-only AD/LDAP hostnames** (typical: `dc01.corp.example.com` on a split-horizon zone) will not resolve, and bind attempts fail with `remoteauth_bind operations error`.

Fix before creating a mapping: add a **DNS Local Record** at **System > DNS Resolver** pointing the upstream FQDN to its actual IP. Verify from inside the container:

```
docker exec hermes_ldap getent hosts <ad-hostname>
```

Publicly-resolvable hostnames don't need this step.

TLS settings reference

Setting	Values	Notes
Use STARTTLS	`yes` / `no`	Upgrades the connection on the standard `389` port. Mutually exclusive with LDAPS on `636` (use one or the other).
TLS Certificate Requirement	`never`, `allow`, `try`, `demand`	Maps directly to `TLS_REQCERT` in the libldap conf. `never` is the only mode that does **not** require a CA cert; the others all expect a valid `ca_cert_file` to compare against.
CA Certificate	PEM file (`.pem`, `.crt`, `.cer`)	Stored at `/opt/hermes/certs/remoteauth/global_remoteauth_ca.pem` (single canonical filename — uploading replaces). For multi-server installs, concatenate all CAs into a bundle.
Retry Count	`1`–`10` (default `3`)	Number of bind retries before reporting failure

The CA field hides itself when `tls_reqcert = never` (purely a UX hint — the file still exists on disk if previously uploaded).

Apply Settings — the sync flow

Every save handler (`add_mapping`, `update_mapping`, `delete_mappings`, `update_tls_settings`, `set_remoteauth_status`) sets `ldap_synced = 0` on the touched rows AND on `remoteauth_settings`. The page banner switches from green **Synced** to amber **Pending Changes**. Nothing has actually changed in LDAP yet.

Apply Settings runs `inc/ldap_remoteauth_sync_all.cfm`, which is a hard three-step sequence:

1. **Delete** the existing overlay (`ldap_remoteauth_delete_overlay.cfm`) — succeeds whether or not one exists.
2. If `enabled = 1` and at least one mapping has `enabled = 1`: **fetch the next overlay index** and the MDB database index (`ldap_remoteauth_get_overlay.cfm`), then **create** the new overlay with all enabled mappings baked in (`ldap_remoteauth_add_overlay.cfm`). The LDIF template is `/opt/hermes/templates/ldap_remoteauth_add_overlay.ldif`, populated via `REReplace` against `THE_OVERLAY_INDEX`, `THE_MDB_INDEX`, `THE_DEFAULT_DOMAIN`, `THE_MAPPING_LINES`, `THE_STARTTLS`, `THE_TLS_REQCERT`, `THE_TLS_CACERT`, `THE_RETRY_COUNT`.
3. **Flip** `ldap_synced = 1` on both tables.

If step 1 or 2 fails, the database `ldap_synced` flags are **not** flipped — the page stays amber, and the next attempt will retry from scratch. There is no half-applied state to clean up because the overlay is rebuilt from zero each time.

> **Failure semantics.** While the overlay is being rebuilt (typically subsecond), live remote-auth web logins will fail with `Operations error` until step 2 completes. Plan Apply Settings during low-login windows. Local-auth users are unaffected.

Deletion validation

A domain mapping cannot be deleted if any user references it. The check runs against **two** tables at delete time:

```
SELECT remoteauth_domain, COUNT(*) FROM system_users
 WHERE auth_type = 'remote' AND remoteauth_domain IN (...);

SELECT remoteauth_domain, COUNT(*) FROM recipients
 WHERE auth_type = 'remote' AND remoteauth_domain IN (...);
```

If either returns rows, the delete is rejected with a list of the blocked domains. The admin must either reassign those users to a different mapping or delete the users first.

> **Known gap (#102 and the mailbox/relay TODO).** When RemoteAuth is extended to **mailboxes** (a planned feature), this validation must add a third query against the `mailboxes` table. Both `view_remoteauth.cfm` (bulk delete, line ~330) and `edit_remoteauth_mapping.cfm` (single delete, line ~129) need to be

> updated together — they implement the check independently.

Adding RemoteAuth users in bulk — CSV format

`add_internal_recipients.cfm` (Relay Recipients > Add) supports a RemoteAuth dropdown when the page detects an enabled mapping. When the selected mapping's DN pattern uses `{firstname}` or `{lastname}`, the textarea switches to **CSV mode** because email-only input doesn't carry enough data to expand the pattern.

DN pattern tokens used	Textarea format
`{username}` and/or `{email}` only	One email address per line
Includes `{firstname}` or `{lastname}`	`First,Last,Email` per line — one recipient per row

Header rows (`"GivenName","Surname","Mail"`) are auto-detected and skipped. Unknown columns are ignored, so common export formats work as-is:

- **PowerShell**: `Get-ADUser -Filter * -Properties GivenName,Surname,Mail | Select GivenName,Surname,Mail | Export-Csv users.csv -NoTypeInformation`
- **CSVDE** (Windows Server built-in): `csvde -f users.csv -l "givenName,sn,mail"`
- **Excel / manual**: three columns saved as CSV

Each row is inserted with `auth_type = 'remote'` and `remoteauth_domain = <mapping key>`. The local LDAP stub is created via `ldap_add_user_relay_remoteauth.cfm`, which calls the same template/placeholder machinery described above. A welcome email is sent via `send_recipient_welcome_email_remoteauth.cfm` — the message tells the user to sign in with their **organization (AD/LDAP) password**, not a Hermes-issued one.

Status, enable, disable

The **RemoteAuth Status** dropdown (`enabled = 0/1`) is the master switch. Disabling does **not** delete the overlay's mappings — it just causes the next Apply Settings cycle to skip step 2 entirely, leaving the overlay absent. Re-enabling and re-applying rebuilds it from the same `remoteauth_mappings` rows. This is useful for emergency cutover back to a local-only state without losing the mapping configuration.

The **LDAP Overlay** badge on the page reads the live state from `cn=config` (via `ldapsearch -Y EXTERNAL` against `(objectClass=olcRemoteAuthCfg)`) and reports **Active** or **Not configured**. This is independent of the DB-side `enabled` flag — if the two disagree (e.g., DB says enabled but the badge says Not configured), the next Apply Settings will reconcile.

License gating

The page is wrapped in the standard Pro-only guard:

```
<cfif NOT isDefined("session.edition") OR session.edition NEQ "Pro">
    <cfinclude template="./inc/license_pro_required.cfm">
    <cfabort>
</cfif>
```

Community-edition installs see the standard "Pro feature required" panel and cannot reach the configuration UI. Pre-existing RemoteAuth-mode users continue to authenticate (the overlay itself is in `cn=config` and not license-checked), but no new mappings can be added or edited until a Pro license is activated.

Files and containers touched

Path	Owner	Role
`config/hermes/var/www/html/admin/2/view_remoteauth.cfm`	`hermes_commandbox`	Main page
`config/hermes/var/www/html/admin/2/edit_remoteauth_mapping.cfm`	`hermes_commandbox`	Edit single mapping
`config/hermes/var/www/html/admin/2/inc/ldap_remoteauth_sync_all.cfm`	`hermes_commandbox`	Apply Settings orchestrator
`config/hermes/var/www/html/admin/2/inc/ldap_remoteauth_add_overlay.cfm`	`hermes_commandbox`	LDIF render + `ldapadd`
`config/hermes/var/www/html/admin/2/inc/ldap_remoteauth_delete_overlay.cfm`	`hermes_commandbox`	`ldapdelete` of existing overlay
`config/hermes/var/www/html/admin/2/inc/ldap_add_user_remoteauth.cfm`	`hermes_commandbox`	Create local stub entry with `seeAlso` / `associatedDomain`
`config/hermes/opt/hermes/templates/ldap_remoteauth_add_overlay.ldif`	`hermes_commandbox`	Overlay LDIF template (placeholder-substituted)
`config/hermes/opt/hermes/templates/ldap_adduser_remoteauth.ldif`	`hermes_commandbox`	Stub-user LDIF template
`/opt/hermes/certs/remoteauth/global_remoteauth_ca.pem`	`hermes_ldap` (mounted)	CA / CA-bundle for upstream TLS

Path	Owner	Role
`/opt/hermes/tmp/<token>_remoteauth_add_overlay.ldif`	`hermes_commandbox`, `hermes_ldap`	Ephemeral rendered LDIF; deleted after `ldapadd`
`cn=config` (in `hermes_ldap`)	`hermes_ldap`	Live overlay configuration

Every shell-out uses `docker exec hermes_ldap …` per the standard Hermes Docker pattern.

Future work

- **#102** — when RemoteAuth is wired to mailboxes (currently relay-recipients and console users only), deletion validation in `view_remoteauth.cfm` and `edit_remoteauth_mapping.cfm` must add a third query against `mailboxes`.
- **Position-2 mapping unique index hardening** — `remoteauth_mappings.domain_name` is `UNIQUE` but the upstream `server_address` is not; an admin can accidentally create two mappings to the same DC under different domain keys. Not a bug, but worth surfacing in a validation hint.
- **Group-based authorization** — current model is "if the upstream bind passes, the user is in." There's no upstream-group filter (e.g., "only members of `cn=hermes-users` may log in"). For installs that need this today, restrict at the upstream side with a dedicated OU.

Related

- Credential Model — full picture of how RemoteAuth slots into the four-credential architecture (web vs. mail vs. DAV)
- System Users — creating console admins/readers with RemoteAuth mode
- DNS Resolver — required prerequisite for internal-only AD hostnames

---

Revision #14
Created 2026-05-31 12:51:59 UTC by Dino Edwards
Updated 2026-06-13 12:30:03 UTC by Dino Edwards