Souvera Mail – Admin

Operator-Doku für Souvera Mail: OIDC-Setup, occ-Kommandos, Migration/Cronjobs, provider.tools-Integration, Fehlerbehebung.

Import-Wizard — Betrieb & Fehlerbehebung

Der **Mail-Import-Assistent** überträgt Mail-Konten von einem beliebigen IMAP-Anbieter nach Souvera Mail. Diese Seite dokumentiert den vollständigen Ende-zu-Ende-Fluss aus Operator-Sicht: Aktivierung, Cronjobs, DB-Schema, Fehlerbilder, Rollback und Datenschutz.

> **Kurzfassung:** Der Wizard ist ab v0.14.9 aktiv, nutzt `provider.tools` als externen IMAP-Migrations-Dienst und benötigt einen zentralen API-Token in *Souvera Central*. Ohne diesen Token bleibt das UI komplett unsichtbar (kein User-facing-Fehler).

1. Aktivierung / Voraussetzungen

- **Souvera Mail ≥ v0.14.9** (Import-Backend + Wizard-Vue-App).
- **Souvera Central** mit gesetztem `provider.tools`-API-Token (siehe *Central-Admin → Integrationen*). Der Token wird zur Laufzeit via `\OCA\SouveraCentral\Service\ProviderTokenService::getToken()` gelesen — Souvera Mail speichert selbst keinen Token-Wert.
- **Stalwart-Admin-API** muss vom NC-Server erreichbar sein (wird verwendet, um ein temporäres Ziel-App-Passwort pro Import zu erstellen und nach Abschluss zu widerrufen).
- **Cron** muss laufen — Ajax-Cron reicht, systemd-Cron empfohlen. Ohne Cron blockiert der Poller den Fortschritt bis zum nächsten Seitenaufruf.

Sichtprüfung

```
occ souvera_mail:status --json | jq '.migration'
```

Erwarteter Output bei sauber aktivierter Integration:

```
{
  "available": true,
  "providerToolsTokenSet": true,
  "stalwartAdminReachable": true,
  "activeJobs": 0,
  "lastJobs": [ ... ]
}
```

Ist `available: false`, taucht das Wizard-Widget bei End-Usern nicht auf — kein Popup, kein Menü-Eintrag im Snappymail-Dropdown, keine Pill. Das ist Absicht (fail-silent).

2. Datenmodell

Zwei DB-Tabellen tragen den Import-Zustand:

`oc_souvera_migrations`

Spalte	Typ	Bemerkung
`id`	bigint PK	
`user_id`	varchar(64)	NC-UID
`status`	varchar(16)	`pending` · `running` · `completed` · `failed` · `cancelled` · `dismissed`
`source_host`	varchar(255)	alter IMAP-Server
`source_user`	varchar(255)	alter Login-Name
`provider_job_id`	varchar(64)	provider.tools `migrationId`
`stalwart_app_id`	varchar(64) NULL	ID des temporären Ziel-App-Passworts. NULL nach Widerruf.
`progress_json`	text	Cache des letzten `getStatus`-Responses: `{progress:{foldersDone,foldersTotal,messagesDone,messagesTotal,currentFolder}, queue:{position,totalInQueue}}`
`error_message`	text NULL	Bei `failed` / `cancelled`
`created_at`, `updated_at`, `finished_at`	bigint (Unix-ts)	

Keine Passwörter werden in dieser Tabelle gespeichert. Das Quell-Passwort geht 1:1 an provider.tools und wird sofort nach der API-Antwort aus dem PHP-Speicher freigegeben; das Ziel-App-Passwort lebt ausschliesslich in Stalwart (referenziert per `stalwart_app_id`).

3. Fluss aus Operator-Sicht

1. **User füllt IMAP-Formular aus** → `POST /apps/souvera_mail/migration/test-connection` → provider.tools `/imap/test` mit den Quell-Credentials. HTTP 200 + `{result:{success:true}}` bedeutet Login klappt.
2. **Ordner-Preview** → `POST /apps/souvera_mail/migration/list-folders` → provider.tools `/imap/list-folders`.

3. **User wählt Ordner + startet** → `POST /apps/souvera_mail/migration/start`. Der Controller ruft `MigrationService::startForUser`, die:
   1. Erstellt ein *temporäres Stalwart-App-Passwort* via `AppPasswordService::createStalwartOnlyForMigration`. Label: `Souvera Import YYYY-mm-dd HH:MM`.
   2. Legt einen `oc_souvera_migrations`-Row mit Status `pending` an (Audit-Trail-First).
   3. Ruft `POST /imap/migrate` bei provider.tools mit Quelle, Ziel und ausgewählten Ordnern.
   4. Speichert die zurückgegebene `migrationId`.
4. **Fortschritt** → das Frontend polled alle 5 s `GET /migration/status`. Ist die DB-Zeile älter als 10 s, ruft der Controller synchron `MigrationService::refreshFromProvider` — das schliesst die Lücke zwischen zwei Poller-Cron-Ticks (60 s).
5. **Cronjob** `MigrationPoller` läuft alle 60 s, ruft `refreshFromProvider` für jede Zeile in `ACTIVE_STATUSES`.
6. **Terminaler Zustand** (`completed`/`failed`/`cancelled`) → `stalwart_app_id` wird widerrufen und auf NULL gesetzt.
7. **Cronjob** `MigrationCleanup` läuft täglich, widerruft eventuell vergessene Ziel-Passwörter älter als 24 h (Belt-and-Suspenders gegen Poller-Ausfälle).

4. Cancel-Flow (v0.14.16)

End-User können einen Job **nur im Zustand** `pending` abbrechen (via UI-Button, oder mit `POST /apps/souvera_mail/migration/cancel/{jobId}`). Das Backend:

1. Prüft Ownership (403 CONFLICT falls fremder Job) und Status (409 CONFLICT falls nicht mehr pending).
2. Ruft `AppPasswordService::revokeStalwartOnlyForMigration` — *bevor* irgendetwas anderes passiert. Damit ist provider.tools ab jetzt vom Zielsystem ausgesperrt: sollte ein Worker den Job zu spät aufgreifen, scheitert er am IMAP-AUTH → Job stirbt still upstream.
3. Setzt `status='cancelled'`, `finished_at=now()`, `error_message='Vom Benutzer abgebrochen …'`.

provider.tools hat **kein** Cancel-Endpoint (Design-Entscheidung, dokumentiert im `ProviderToolsClient.php`-Header). Der Weg über den App-PW-Widerruf ist der offizielle Ersatz.

Für `running`-Jobs ist Cancel bewusst gesperrt — mid-transfer-Abbruch würde einen unvollständigen Ordner hinterlassen. Falls das jemals nötig sein sollte, ist `forceCancel` als zukünftiges Verb im MigrationService reserviert.

5. Wizard aus Sicht des End-Users

Der End-User sieht den Wizard an zwei Stellen:

- **Beim ersten Login**: Welcome-Popup öffnet automatisch. Klick auf *Nicht mehr zeigen* setzt `oc_appconfig_users`-Key `souvera_mail::migration_welcome_dismissed=1`.
- **Später**: neuer Menü-Eintrag **„□□ Alte Mails importieren"** im Snappymail-Top-Right-Dropdown (zwischen ⚙ Einstellungen und □□ Hilfe). Injiziert clientseitig durch `plugins/nextcloud/js/dropdown-menu.js` — kein PHP-Nav-Eintrag im NC-Global-Menü.

6. Häufige Fehlerbilder

„Import-Dienst ist auf dieser Instanz nicht aktiviert" (HTTP 503)

Der `provider.tools`-Token in Souvera Central fehlt. Setzen mit `occ souvera_central:integrations:set provider_tools_token <TOKEN>`.

„user must not be empty; password must not be empty" (HTTP 400 auf `/test-connection`)

Regressions-Bug aus v0.14.11-12, seit v0.14.13 gefixt. Bei modernen Souvera-Mail-Versionen darf das nicht mehr passieren; falls doch, Frontend-Bundle prüfen (`js/souvera_mail-migration-wizard.js` muss aus `src/` gebaut sein — nicht mehr das alte vanilla-JS).

„folders must be a non-empty array" (HTTP 400 auf `/start`)

Seit v0.14.14 gefixt: das Frontend erzwingt ausdrückliche Ordner-Auswahl via *Mapping-Screen*. Bei älteren Installationen: Upgrade auf ≥ 0.14.14 oder als Notfall-Workaround alle Ordner via API senden.

UI hängt bei „Warteschlange…" für ? 60 s

Der `MigrationPoller`-Cron läuft nicht. Prüfen mit `occ background:list`, dann Cron-Modus in *Grundeinstellungen → Hintergrund-Aufgaben* auf *Cron (systemd)* stellen.

Seit v0.14.15 macht der `/status`-Endpoint einen synchronen On-Demand-Poll wenn die Row älter als 10 s ist — die Wartezeit ist damit auf ~15 s begrenzt. Wenn es trotzdem länger dauert, ist provider.tools selbst überlastet (Prüfung: `curl https://provider.tools/status`).

„Migration is already active for this user" (HTTP 409)

Pro User ist maximal ein aktiver Job erlaubt (Design-Entscheidung, verhindert konkurrierende Transfers auf dasselbe Zielpostfach). Löschen via `DELETE FROM oc_souvera_migrations WHERE user_id='<uid>' AND status IN ('pending','running');` ist als *Notfall-Ausstieg* möglich — sauberer ist:

```
occ souvera_mail:migrations:cancel <uid> --job-id=<n>
```

(occ-Kommando ist ab v0.14.18 in Arbeit — bis dahin bitte SQL nutzen.)

7. Rollback / Datenlöschung

Bei DSGVO-Löschanfragen oder wenn ein Kunde die Instanz verlässt:

```
-- Widerrufe alle noch bestehenden temp Ziel-App-Passwörter
occ souvera_mail:migrations:list --user=<uid> --active | while read job_id; do
  occ souvera_mail:migrations:cancel <uid> --job-id=$job_id
done
-- Lösche den Verlauf
DELETE FROM oc_souvera_migrations WHERE user_id='<uid>';
```

8. Datenschutz

- Quell-IMAP-Passwörter werden **nie** in der NC-DB gespeichert. Sie werden im Frontend-Formular gehalten (bis zum Klick auf „Import starten"), an provider.tools weitergegeben und danach aus dem Browser-Speicher gelöscht.
- provider.tools verwahrt die Quell-Credentials verschlüsselt bis Job-Ende, danach werden sie gelöscht (siehe deren Datenschutzerklärung).
- Die Ziel-App-Passwörter sind auf 24 h begrenzt und werden bei Job-Abschluss (Erfolg, Fehler, Abbruch) sofort widerrufen. Die `MigrationCleanup`-Cron kehrt zusätzlich täglich vergessene Reste weg.

9. Nützliche `occ`-Kommandos

```
# Alle Import-Jobs eines Users listen
occ souvera_mail:migrations:list --user=<uid>

# Job manuell abbrechen (auch running — Operator-Override)
occ souvera_mail:migrations:cancel <uid> --job-id=<n> --force

# Verwaiste Ziel-App-Passwörter aufräumen (dry-run)
occ souvera_mail:migrations:cleanup --dry-run

# Sofortigen Poller-Lauf triggern (statt auf den 60-s-Cron zu warten)
occ souvera_mail:migrations:poll-all
```

Die letzten drei Kommandos sind ab v0.14.18 geplant (P2-Backlog). Bis dahin geht Cancel nur über den User-UI-Button bzw. via SQL, und das Cleanup läuft ausschliesslich automatisiert per Cron.

10. Verwandte Seiten

- User-Doku: Alte Mails importieren (Endnutzer-Anleitung mit Screenshots)
- Souvera Central → Integrationen → provider.tools-Token
- Stalwart-Admin-API — siehe Souvera-Central-Admin-Doku

Client-Resync — was es kann und was nicht

Der Menü-Eintrag **„□□ Postfach neu synchronisieren"** im Snappymail-Top-Right-Dropdown ist bewusst nur ein *Client-Resync*, nicht ein serverseitiger FTS-Rebuild. Diese Seite beschreibt was er tut, was er nicht tut, und warum das die richtige Entscheidung ist.

Was der Klick tut

1. **Backend-Endpoint** `POST /apps/souvera_mail/stalwart/resync` — nur Audit-Log. Der Endpoint schreibt eine `Souvera Mail: user-initiated mailbox resync uid=…`-Zeile in `nextcloud.log`. Kein Stalwart-Aufruf.
2. **Frontend** löscht alle `localStorage`-Keys mit den Präfixen `rl.`, `snappymail.`, `rainloop.`, `smail.`.
3. **Full page reload** — Snappymail bootstrapt neu, macht einen frischen `GET /jmap/session` gegen Stalwart, baut Ordner-Baum und Message-Liste komplett neu im Speicher auf.

Was der Klick *nicht* tut

> Er triggert **keinen** serverseitigen FTS-Reindex. Stalwart 0.16 hat dafür kein API-Endpoint — die FTS-Indizierung läuft automatisch im Hintergrund als Task-Warteschlange innerhalb Stalwarts selbst.

Falls du wirklich einen FTS-Rebuild brauchst (z. B. nach Datenmigration oder wenn der FTS-Store korrumpiert war), musst du direkt an der Stalwart-Instanz ansetzen:

```
# Nur im Server-Shell — nicht via Souvera Mail
stalwart-cli --url https://stalwart.souvera.eu --credentials admin:$SECRET \
   database maintenance
```

Details in der offiziellen Stalwart-Doku unter stalw.art/docs — *Server* → *Storage* → *Maintenance*. Der Server-side Rebuild ist absichtlich nicht ins Souvera-Mail-UI eingebaut, weil er (a) Instanz-weit läuft, (b) DB-Blocking-Charakter hat und (c) Admin-Credentials braucht.

Wann hilft der Client-Resync tatsächlich?

- **Stale IMAP-Statuscodes** — Snappymail cached UIDVALIDITY-Werte. Nach Server-side-Umzügen von Postfächern (z. B. Migration) sind die veraltet.
- **Fehlende Ordner nach Quota-Erhöhung** — der Client hatte den Ordner mangels Platz aus dem lokalen State geworfen.
- **Ungelesen-Counter-Drift** — kann passieren wenn ein anderer IMAP-Client (Handy, Thunderbird) parallel Flags ändert und Snappymails Sync einen Roundtrip verpasst hat.
- **Verhakte JS-Zustände** — Race-Conditions zwischen Ordner-Wechsel und Nachrichten-Load-Callback.

Wo landen die Log-Zeilen?

```
grep 'user-initiated mailbox resync' /var/www/nextcloud/data/nextcloud.log
```

Jeder Klick auf das Menü erzeugt einen INFO-Log-Eintrag. Bei häufigen Klicks eines Users ist das ein guter erster Hinweis, dass mit dessen Postfach etwas nicht stimmt.

Verwandte Seiten

- User-Doku: Postfach neu synchronisieren
- Admin-Doku: Import-Wizard — Betrieb & Fehlerbehebung