FOCUS FAMILY

Technisches Handbuch

Server betreiben, absichern und administrieren

Version 1.3.0 · Stand November 2026 · Software Notion

Inhaltsverzeichnis

1. **Über dieses Handbuch**
    1. Grundsätze des Betriebsmodells
2. **Architektur**
    1. Datenhaltung und Volumes
    2. Rollen in der Datenbank
3. **Voraussetzungen**
4. **Installation Schritt für Schritt**
    1. 1. Dateien auf den Server bringen
    2. 2. Konfiguration anlegen
    3. 3. Zertifikate erzeugen
    4. 4. Signierschlüssel für Offline-Grants
    5. Interne Zertifizierungsstelle (optional)
    6. 5. Erster Start und Erst-Admins
    7. 6. Reverse-Proxy einrichten
    8. Nur für Entwicklungsrechner
5. **Konfigurationsreferenz**
6. **Datenbank und Migrationen**
7. **Sicherung und Wiederherstellung**
    1. Was gesichert werden muss
    2. Backup
    3. Wiederherstellung
    4. Test-Wiederherstellung
    5. Versionskompatibilität
8. **Aktualisierung**
    1. Update aus dem Admin-Panel
9. **Lizenzierung**
10. **Single Sign-On (SSO)**
    1. Voraussetzungen
    2. OpenID Connect oder SAML 2.0?
    3. Die Seite „Single Sign-On“
    4. Microsoft Entra ID
    5. Lokales Active Directory über AD FS
    6. Keycloak
    7. Okta
    8. Google Workspace
    9. Andere Anbieter
    10. Benutzerkonten und Verknüpfung
    11. Anmeldeverfahren umstellen und Verknüpfung lösen
    12. Notfallzugang und „SSO erzwingen“

13. Erneute Anmeldebestätigung
14. Betrieb im Intranet
15. Sicherheit und Geheimnisse
16. Fehlerbehebung bei SSO
17. Audit-Ereignisse
18. Checkliste SSO-Einführung

11. Schnittstellen

12. Betrieb im Intranet oder über VPN

1. Ausgehende Verbindungen des Servers
2. Ausgehender Proxy
3. Geräte, Lizenz und Sync
4. DNS und Zertifikate
5. VPN

13. Sicherheit

1. Transport und Absicherung
2. Autorisierung
3. Gerätebindung
4. Anmeldung
5. Uploads
6. Offline-Grant-Keyset rotieren

14. Technische Bereiche des Admin-Panels

1. System-Status
2. Geräte
3. Absturzberichte
4. Sicherheits-Audit
5. Nachweis-Archiv
6. Lizenzierung

15. Betrieb und Überwachung

1. Health-Endpunkte
2. Protokolle
3. Zeitsynchronisation prüfen
4. Monitoring
5. Sync-Aufbewahrung und vollständiger Abgleich

16. Störungen und Vorfälle

1. Verlorenes oder gestohlenes Gerät
2. Fachkraft aus der App-Sperre ausgesperrt
3. Kompromittiertes Nutzerkonto
4. Kompromittierte Serverschlüssel
5. Datenbank- oder Backup-Diebstahl
6. Verdacht auf bösartigen Upload

17. Fehlerbehebung

18. Checklisten

1. Inbetriebnahme
2. Monatliche Routine

19. Anhang

1. Wichtige Pfade und Ports
2. Nützliche Befehle
3. Weiterführende Dokumentation im Repository

Über dieses Handbuch

Dieses Handbuch richtet sich an **IT-Administration und technische Dienstleister**, die den Focus-Family-Server eines Trägers betreiben. Es beschreibt Architektur, Installation, Konfiguration, Absicherung, Sicherung, Aktualisierung und Fehlersuche – und die technischen Bereiche des Admin-Panels.

Die fachliche Bedienung des Panels steht im **Handbuch für das Admin-Panel**, die Bedienung der Apps in den beiden App-Handbüchern.

Vorausgesetzte Kenntnisse: Linux-Grundlagen, Docker und Docker Compose, ein Reverse-Proxy (nginx, Caddy oder Traefik) sowie der Umgang mit TLS-Zertifikaten.

Grundsätze des Betriebsmodells

- **Ein Träger betreibt eine Serverinstanz** in eigener Infrastruktur. Es gibt keine Hersteller-Cloud.
- Der Stack benötigt **keine Verbindung** zu Analyse-, Crash- oder KI-Diensten des Herstellers.
- Der einzige optionale externe Kontakt ist der **Lizenzserver** (fflicense) für die Organisationslizenz.
- Die Apps funktionieren **vollständig offline**; der Server ist für Teamarbeit und Verwaltung zuständig.

Architektur

Der Stack besteht aus vier Containern und einem optionalen fünften:

Dienst	Aufgabe
db	PostgreSQL 17, nicht öffentlich erreichbar, benanntes Volume `pgdata`
migrator	One-Shot beim Start: legt das Schema an bzw. wendet ausstehende Schema-Migrationen an und provisioniert die App-Rolle
clamav	Malware-Scan aller Uploads, benanntes Volume `clamdb` für die Signaturen
app	der Server selbst, gebunden an `127.0.0.1:8080`, hinter dem Reverse-Proxy des Trägers
updater	optional (`COMPOSE_PROFILES=updater`): spielt Updates aus dem Admin-Panel ein, einziger Dienst mit Zugriff auf den Docker-Socket

Datenhaltung und Volumes

Volume	Inhalt	Folge bei Verlust
`pgdata`	PostgreSQL-Daten	vollständiger Datenverlust
`blobs`	Anhänge, ruhend verschlüsselt	Dokumente und Medien verloren
`dpkeys`	ASP.NET-Data-Protection-Keyring	alle Sitzungen und Antiforgery-Token ungültig
`clamdb`	ClamAV-Signaturen	werden neu geladen (unkritisch)
`licstate`	Identität der Lizenz-Installation	Neu-Provisionierung beim Lizenzserver nötig
`deploy/docker/keys`	Signier- und Verschlüsselungszertifikate (read-only)	Anmeldung nicht mehr möglich

> **Merksatz:** `pgdata`, `dpkeys` und `licstate` bilden zusammen die **Identität der Installation**. Werden sie gelöscht, wird aus einer Wiederherstellung eine Neu-Provisionierung.

In Produktion (`ASPNETCORE_ENVIRONMENT=Production`) verschlüsselt der Server die Schlüssel im `dpkeys`-Volume mit dem Verschlüsselungszertifikat `encryption.pfx` (`CERT_PASSWORD`). Beim ersten Start nach dem Update werden die unverschlüsselt abgelegten Schlüssel aus Version 1.0.0 an Ort und Stelle verschlüsselt (Protokollzeile „Encrypted … data-protection key file(s)“); Sitzungen und geschützte Daten bleiben lesbar, neue Schlüssel werden direkt verschlüsselt geschrieben. Wer nur den Datenträger mit `dpkeys` liest, kann damit keine Sitzungen mehr erzeugen. Sicherungen von vor dem Update enthalten die Schlüssel weiterhin unverschlüsselt und sind entsprechend zu schützen oder zu löschen. Das Zertifikat ist damit Teil der Installationsidentität: ohne `encryption.pfx` sind Sitzungen, die Lizenz-Installationsidentität und die SSO-Geheimnisse nicht mehr lesbar. Sichern Sie `deploy/docker/keys/` zusammen mit `.env`.

Rollen in der Datenbank

- `fokus_migrator` besitzt das Schema und darf DDL ausführen. Diese Rolle nutzt ausschließlich der Migrator-Container.
- `fokus_app` betreibt die Anwendung und hat **nur DML-Rechte**; `CREATE TABLE` und `DROP` sind verweigert.

Die Anwendung selbst führt **nie** DDL aus. Schemaänderungen laufen ausschließlich über den kontrollierten Migrator-Schritt.

Voraussetzungen

- **Server:** Linux (amd64/x86_64), Docker Engine mit Compose v2.
- **Empfehlung:** 4 CPU-Kerne, 8 GB RAM, 50 GB Speicher als Ausgangspunkt. ClamAV allein braucht etwa 1,5–2 GB RAM.
- **Netzwerk:** ein DNS-Name für den Server (öffentlich oder nur intern) und ein TLS-Zertifikat (institutionelle PKI oder Let's Encrypt) am Reverse-Proxy. Aus dem Internet muss der Server nicht erreichbar sein; ausgehend braucht er HTTPS zu `fflicense.software-notion.com` (siehe „Betrieb im Intranet oder über VPN“).

- **Zertifikate:** produktive **Signatur- und Verschlüsselungszertifikate** im PKCS#12-Format für die Token-Ausstellung.
- **Lizenz:** ein **Enrollment-Key** vom Betreiber des Lizenzservers, falls eine Organisationslizenz genutzt wird.
- **Zeitsynchronisation:** Der Wirt muss seine Uhr fortlaufend per NTP stellen (chrony, systemd-timesyncd oder ntpd, bevorzugt gegen einen internen Zeitserver). Die Container haben keine eigene Uhr, sie übernehmen die des Wirts. Gerätenachweise (±5 Minuten), Token, Zwei-Faktor-Codes, SSO-Antworten, Lizenzprüfung, Zeitstempel des Audit-Protokolls und die Fristläufe hängen davon ab (siehe „Zeitsynchronisation prüfen“).

> In einer Entwicklungsumgebung startet der Server mit ephemeren Schlüsseln. In Produktion sind die Zertifikate **Pflicht** – ohne sie startet die Anwendung bewusst nicht.

Installation Schritt für Schritt

1. Dateien auf den Server bringen

Kopieren Sie das Deployment-Verzeichnis (Compose-Datei, `.env.example`, `keys/`, Proxy-Beispiele, Backup-Skripte) auf den Server, zum Beispiel nach `/opt/focusfamily`.

Das Server-Image wird **einmalig auf einem Build-Rechner** erzeugt und als Tarball mitgeliefert – auf dem Server ist kein .NET-SDK nötig:

```
docker load -i image/focusfamily-server-1.3.0-amd64.tar.gz
```

Das geladene Image heißt `fokusfamilie-server`; das ist ein technischer Bezeichner, auf den die Compose-Dateien und der Update-Dienst verweisen.

2. Konfiguration anlegen

Das Partnerpaket enthält `setup.sh`, das `.env`, Zertifikate und Schlüssel in einem Schritt erzeugt (empfohlen). Für die manuelle Einrichtung die Vorlage kopieren und **jeden** `CHANGE_ME`-Wert ersetzen; die Vorlage beschreibt jeden Wert zusammen mit dem Befehl, der ihn erzeugt:

```
cp .env.example .env
chmod 600 .env
```

Wert	Bedeutung	Erzeugen mit
`POSTGRES_PASSWORD`	Passwort des PostgreSQL-Eigentümerkontos `fokus_migrator`; nur der Migrator nutzt es (Schema-Updates), niemand meldet sich damit an	`openssl rand -hex 24`
`FOKUS_APP_PASSWORD`	Passwort des eingeschränkten Laufzeitkontos `fokus_app`, mit dem das Panel auf die Daten zugreift; der Migrator setzt es bei jedem Start	`openssl rand -hex 24`
`SERVER_INSTANCE_ID`	stabile Identität (GUID) dieser Installation; nach dem ersten Start nie mehr ändern	`uuidgen` (Linux ohne `uuidgen`: `cat /proc/sys/kernel/random/uuid`)
`ORGANIZATION_NAME`	Anzeigename des Trägers; mit Leerzeichen in doppelten Anführungszeichen	–
`BOOTSTRAP_ADMIN_EMAIL`, `BOOTSTRAP_ADMIN_PASSWORD` (dazu `BOOTSTRAP_FACHADMIN_*`, `BOOTSTRAP_PRIVACYADMIN_*`, `BOOTSTRAP_ROOT_ADMIN`)	Erst-Admins, siehe Schritt 5; die Passwörter sind Einmal-Passwörter, jeder Wert wird angenommen	z. B. `openssl rand -base64 12`
`SERVICE_CLIENT_ID`, `SERVICE_CLIENT_SECRET`	optionaler vertraulicher Service-Client (Client Credentials); beide leer lassen, wenn nicht benötigt	`openssl rand -hex 24`
`MOBILE_REDIRECT_URI`	zusätzliche Anmelde-Rücksprungadresse der Apps; mit Domain `https://<domain>/callback`, sonst `http://localhost/callback`	–
`OPENIDDICT_ALLOW_INSECURE_HTTP`	`true` **nur** für einen Test ohne TLS (Panel unter `http://127.0.0.1:8080`); hinter HTTPS immer `false`. Bei `false` erhalten in Produktion alle Cookies (Sitzung, Antiforgery, Sprache) das Secure-Attribut, die Verwaltungsoberfläche funktioniert dann nur über HTTPS	–
`CERT_PASSWORD`	Passwort der Zertifikate `signing.pfx` und `encryption.pfx` (Schritt 3)	`openssl rand -hex 16`
`BLOB_ENCRYPTION_KEY_BASE64`	AES-256-Hauptschlüssel (32 Byte, Base64) für Anhänge und Datenkopien	`openssl rand -base64 32`

Wert	Bedeutung	Erzeugen mit
`OFFLINE_GRANT_KEY`	privater EC-P-256-Schlüssel (mehrzeiliges PEM), der die Offline-Grants signiert (Schritt 4)	`openssl ecparam -genkey -name prime256v1 -noout`
`LICENSING_INSTALLATION_KEY`	Einmal-Enrollment-Key des Lizenzbetreibers (`<id>.<secret>`); beim Start optional, ohne ihn können Apps keinen Platz belegen	von Software Notion

Die einfachen Werte erzeugen und direkt in die `.env` schreiben (GNU-`sed`; unter macOS `sed -i ''`):

```
sed -i "s|^POSTGRES_PASSWORD=.*|POSTGRES_PASSWORD=$(openssl rand -hex 24)|" .env
sed -i "s|^FOKUS_APP_PASSWORD=.*|FOKUS_APP_PASSWORD=$(openssl rand -hex 24)|" .env
sed -i "s|^SERVER_INSTANCE_ID=.*|SERVER_INSTANCE_ID=$(uuidgen)|" .env
sed -i "s|^SERVICE_CLIENT_SECRET=.*|SERVICE_CLIENT_SECRET=$(openssl rand -hex 24)|" .env
sed -i "s|^CERT_PASSWORD=.*|CERT_PASSWORD=$(openssl rand -hex 16)|" .env
sed -i "s|^BLOB_ENCRYPTION_KEY_BASE64=.*|BLOB_ENCRYPTION_KEY_BASE64=$(openssl rand -base64 32)|"
.env
nano .env                   # Trägername, Erst-Admins, Redirect-URI, Lizenzschlüssel
```

3. Zertifikate erzeugen

Der Server signiert und verschlüsselt seine Token mit zwei PKCS#12-Zertifikaten in `deploy/docker/keys/`, geschützt durch `CERT_PASSWORD` aus der `.env`:

```
CERT_PASSWORD=$(grep '^CERT_PASSWORD=' .env | cut -d= -f2-)
mkdir -p deploy/docker/keys
T=$(mktemp -d)
openssl req -x509 -newkey rsa:2048 -nodes -keyout "$T/k.pem" -out "$T/c.pem" \
  -days 825 -subj "/CN=fokusfamilie"
openssl pkcs12 -export -inkey "$T/k.pem" -in "$T/c.pem" \
  -out deploy/docker/keys/signing.pfx    -passout pass:"$CERT_PASSWORD"
openssl pkcs12 -export -inkey "$T/k.pem" -in "$T/c.pem" \
  -out deploy/docker/keys/encryption.pfx -passout pass:"$CERT_PASSWORD"
rm -rf "$T"
chmod 644 deploy/docker/keys/*.pfx
```

Der App-Container läuft als unprivilegierter Benutzer und muss beide Dateien lesen können (`chmod 644`). Für echte Deployments verwenden Sie die institutionelle PKI statt eines selbstsignierten Zertifikats.

4. Signierschlüssel für Offline-Grants

`OFFLINE_GRANT_KEY` ist ein privater EC-P-256-Schlüssel. Der Server signiert damit die Offline-Grants, damit die Apps ohne Verbindung weiterarbeiten und ihre Grants auch nach einem Neustart des Servers prüfen können. Der Schlüssel muss stabil bleiben und wird mit der `.env` gesichert; ein neuer Schlüssel macht die bisher ausgestellten Offline-Grants ungültig. Ein Wechsel ist eine geplante Rotation über `OFFLINE_GRANT_KEYSET_VERSION` und `OFFLINE_GRANT_ADDITIONAL_PUBLIC_KEY` (siehe „Offline-Grant-Keyset rotieren“).

Der Wert ist ein mehrzeiliges PEM; das ganze PEM einschließlich der `BEGIN`-/`END`-Zeilen steht in doppelten Anführungszeichen. Die leere Zeile aus der Vorlage entfernen und den erzeugten Schlüssel anhängen:

```
sed -i '/^OFFLINE_GRANT_KEY=$/d' .env
{ printf 'OFFLINE_GRANT_KEY="'; openssl ecparam -genkey -name prime256v1 -noout; printf '"\n'; }
>> .env
```

Das Ergebnis sieht so aus:

```
OFFLINE_GRANT_KEY="-----BEGIN EC PRIVATE KEY-----
MHcCAQEE...
-----END EC PRIVATE KEY-----"
```

Interne Zertifizierungsstelle (optional)

Der Server prüft jede ausgehende TLS-Verbindung (Identitätsanbieter, `fflicense.software-notion.com`, Versions-Feed und Update-Download) gegen die Zertifizierungsstellen des Containers. Nutzt ein Ziel ein Zertifikat einer **internen Zertifizierungsstelle** (typisch für AD FS oder Keycloak im LAN) oder bricht ein Unternehmens-Proxy TLS auf (SSL-Inspection), das Zertifikat der ausstellenden Stelle im PEM-Format (`.crt` oder `.pem`, auch mehrere Zertifikate in einer Datei; DER-`.crt` geht ebenfalls) in `deploy/docker/ca/` ablegen und die App neu starten:

```
cp interne-ca.crt deploy/docker/ca/
docker compose up -d --force-recreate app
docker compose logs app | grep "additional CA"     # Trusting 1 additional CA certificate(s) ...
```

`docker-compose.yml` bindet das Verzeichnis schreibgeschützt unter `/etc/fokusfamilie/ca` ein (Einstellung `OutboundTls__AdditionalCaDirectory`). Diese Zertifizierungsstellen greifen nur, wenn die normale Prüfung allein an der Zertifikatskette scheitert; ein Zertifikat für einen anderen Hostnamen wird immer abgelehnt. Eine Sperrprüfung findet auf diesem Weg nicht statt (wie bei der Standardprüfung der ausgehenden Verbindungen; interne Zertifizierungsstellen veröffentlichen oft keine erreichbare Sperrliste). Der `updater`-Container braucht die Zertifikate nicht: Er spricht nur mit dem Docker-Socket, der Update-Download läuft in der App.

5. Erster Start und Erst-Admins

Solange die Datenbank noch keinen Benutzer enthält, legt der Server beim Start die Erst-Admins aus der `.env` an; sobald Benutzer existieren, tut dieser Schritt nichts mehr. Ein Schalter ist nicht nötig:

```
BOOTSTRAP_ADMIN_EMAIL=admin@ihr-traeger.de
BOOTSTRAP_ADMIN_PASSWORD=<Einmal-Passwort>
BOOTSTRAP_FACHADMIN_EMAIL=leitung@ihr-traeger.de
BOOTSTRAP_FACHADMIN_PASSWORD=<Einmal-Passwort>
BOOTSTRAP_PRIVACYADMIN_EMAIL=datenschutz@ihr-traeger.de
BOOTSTRAP_PRIVACYADMIN_PASSWORD=<Einmal-Passwort>
```

```
docker compose up -d
docker compose logs -f migrator     # Schema anlegen beobachten
curl http://127.0.0.1:8080/health/ready    # erwartet: 200
```

Der Migrator legt das Schema an, danach erzeugt die Anwendung die Erst-Admins. Der Bootstrap legt drei getrennte Konten an: den **technischen Admin** (System, Geräte, Richtlinien, Lizenzierung – keine Fall- oder Datenschutzrechte), den **Fach-Admin** (Fälle, Teams, Fachkräfte, Vorlagen; darf sich auch in der App anmelden) und den **Datenschutz-Admin** (Betroffenenanfragen und Datenschutz-Protokoll). Ein Paar ohne Wert wird übersprungen; für eine neue Installation sind mindestens `BOOTSTRAP_ADMIN_EMAIL` und `BOOTSTRAP_ADMIN_PASSWORD` nötig.

Die Startpasswörter sind **Einmal-Passwörter**: Jeder Wert wird angenommen, die Passwortregeln gelten für sie nicht. Wie bei der Anmeldung werden Leerzeichen am Anfang und Ende sowie unsichtbare Steuerzeichen ignoriert. Jedes Konto muss sein Startpasswort bei der ersten Anmeldung – im Panel wie in der App – durch ein eigenes Passwort ersetzen, das den Regeln entspricht; vorher ist keine andere Seite erreichbar. Das Startpasswort wird **nie** protokolliert. Sobald sich die Admins angemeldet haben, können die `BOOTSTRAP_*`-Werte aus der `.env` entfernt werden; sie werden ignoriert, sobald Benutzer existieren.

Alternative: ein Root-Admin. Kleine Träger, bei denen eine Person alle drei Rollen ausfüllt, können statt der drei Konten einen einzigen Root-Admin anlegen lassen: `BOOTSTRAP_ROOT_ADMIN=true` erzeugt aus `BOOTSTRAP_ADMIN_EMAIL` / `BOOTSTRAP_ADMIN_PASSWORD` ein Konto mit System-, Fach- und Datenschutzrechten; die Werte für Fach- und Datenschutz-Admin werden dann ignoriert. Auch der Root-Admin muss sein Startpasswort bei der ersten Anmeldung ersetzen. Standard ist `false` (drei getrennte Konten), weil die Rollentrennung Datenschutz und Nachvollziehbarkeit stärkt; die Rechte lassen sich später unter **Benutzer** jederzeit aufteilen.

Startprüfung. In Produktion prüft der Server seine Konfiguration, bevor er Anfragen annimmt: `OFFLINE_GRANT_KEY` vorhanden und lesbar, `BLOB_ENCRYPTION_KEY_BASE64` ein Base64-Schlüssel mit 32 Byte, beide Zertifikate vorhanden und mit `CERT_PASSWORD` lesbar, `SERVER_INSTANCE_ID` eine GUID, `ORGANIZATION_NAME` und das Datenbankpasswort gesetzt und – bei leerer Datenbank – mindestens ein Erst-Admin konfiguriert. Fehlt etwas, beendet sich der App-Container mit Code 1 und listet alle Probleme zusammen mit dem Befehl auf, der sie behebt:

```
docker compose logs app
# Focus Family cannot start: the production configuration is incomplete.
# Fix the following settings in .env (see .env.example) and run "docker compose up -d" again:
#   - OFFLINE_GRANT_KEY is empty. Generate it with: openssl ecparam -genkey -name prime256v1 -
noout ...
```

`BOOTSTRAP_ENABLED` aus älteren Installationen wird nicht mehr benötigt und ignoriert.

Sichern: die `.env` (vor allem `CERT_PASSWORD`, `BLOB_ENCRYPTION_KEY_BASE64`, `OFFLINE_GRANT_KEY`, `SERVER_INSTANCE_ID`) und `deploy/docker/keys/*.pfx` getrennt von den Daten-Volumes. Ohne sie lassen sich Anhänge nach einer Wiederherstellung nicht entschlüsseln und Geräte nicht mehr anmelden.

6. Reverse-Proxy einrichten

Terminieren Sie TLS am Proxy und leiten Sie auf `app:8080` weiter. Beispiele für nginx und Caddy liegen im Deployment-Verzeichnis.

Wichtig:

- `X-Forwarded-Proto`, `X-Forwarded-Host` und `X-Forwarded-For` weiterreichen.
- Forwarded-Header werden nur von vertrauenswürdigen Proxy-Adressen akzeptiert. Das Compose-Netz `web` hat dafür das feste Subnetz `REVERSE_PROXY_NETWORK` (Standard `172.30.250.0/24`), das `docker-compose.yml` als `ReverseProxy__KnownNetworks__0` übergibt. Ein Proxy-Container, der an `focusfamily_web` hängt, gilt damit automatisch als vertrauenswürdig. In diesem Netz sollten nur der Proxy und `app` hängen. Den Bereich nur ändern, wenn er mit einem vorhandenen Netz auf dem Host kollidiert (`docker compose up` meldet dann „Pool overlaps“).
- Läuft der Proxy **direkt auf dem Host** (Weiterleitung an `127.0.0.1:8080`), kommen die Anfragen vom Bridge-Gateway des Netzes. Diese Adresse in `REVERSE_PROXY_IP` eintragen: `docker run --rm --network container:focusfamily-app-1 busybox ip route` → `default via <Gateway>`.
- Empfohlen: `REVERSE_PROXY_ALLOWED_HOST` auf den Hostnamen des Panels setzen (z. B. `focusfamily.ihr-traeger.de`). Weitergereichte Header werden dann nur angewendet, wenn `X-Forwarded-Host` diesen Namen trägt; ein gefälschter Host gelangt so nicht in Weiterleitungen, Token-Aussteller und Einrichtungs-QR-Codes. Den Proxy ebenfalls nur für diesen Namen antworten lassen (kein Catch-all-Server). Leer bleibt der Standard `*`.
- `client_max_body_size` (nginx) groß genug für Anhänge wählen.
- Bei nginx die Puffer für Antwort-Header vergrößern: `proxy_buffer_size 16k; proxy_buffers 8 16k; proxy_busy_buffers_size 32k;` (im Beispiel enthalten). Der Standardpuffer von 4 KB reicht für die Antworten der SSO-Anmeldung (Cookies und Weiterleitung) nicht sicher aus; nginx antwortet dann mit „502 Bad Gateway“ und protokolliert `upstream sent too big header while reading response header from upstream`. Caddy hat diese Grenze nicht und braucht keine Anpassung.

Ältere Compose-Dateien legten das Netz `web` ohne festes Subnetz an. Nach dem Austausch der Compose-Datei einmal `docker compose down` (ohne `-v`) und `docker compose up -d` ausführen, damit Docker das Netz neu anlegt; prüfen mit `docker network inspect focusfamily_web -f '{{range .IPAM.Config}}{{.Subnet}} {{end}}'`.

Nur für Entwicklungsrechner

Zwei Overlay-Dateien im Repository sind ausschließlich für Entwicklung und Tests gedacht und gehören **nie** auf einen Server mit echten Falldaten:

- `docker-compose.noclamav.yml` ersetzt ClamAV durch einen Platzhalter und schaltet den Upload-Scan ab (für Rechner ohne ClamAV-Image, etwa arm64).
- `docker-compose.local.yml` setzt `ASPNETCORE_ENVIRONMENT=Local` und `Bootstrap__DevAdmin=true`. Damit erhält `admin@` alle Rechte, keinen Pflicht-Passwortwechsel und bei jedem Start wieder das Passwort aus der `.env`. Der Server lässt das nur in den Umgebungen `Development` und `Local` zu und verweigert es zusätzlich, wenn `ReverseProxy:Enabled` gesetzt ist, der Lizenzserver auf `fflicense.software-notion.com` zeigt oder die Admin-Adresse nicht zu einer Entwicklungsdomain gehört; in `Production` wird die Einstellung ignoriert und im Protokoll gewarnt.

Ein lokaler Entwicklungsstack startet mit `docker compose --env-file .env -f docker-compose.yml -f docker-compose.noclamav.yml -f docker-compose.local.yml -p focusfamily up -d`.

Konfigurationsreferenz

Die Compose-Datei liest folgende Variablen aus der `.env`. Die `.env` gehört **nie** in die Versionsverwaltung; Werte können auch aus Docker-Secrets oder einem Secret-Store kommen.

Variable	Zweck
`POSTGRES_PASSWORD`	Passwort der DDL-Rolle `fokus_migrator`
`FOKUS_APP_PASSWORD`	Passwort der Least-Privilege-Rolle `fokus_app`
`SERVER_INSTANCE_ID`	stabile GUID-Identität dieser Server-Instanz
`ORGANIZATION_NAME`	Anzeigename des Trägers
`BOOTSTRAP_ADMIN_EMAIL`, `BOOTSTRAP_ADMIN_PASSWORD`	technischer Erst-Admin; wird beim ersten Start automatisch angelegt, solange die Datenbank keinen Benutzer enthält (Einmal-Passwort, jeder Wert wird angenommen)
`BOOTSTRAP_FACHADMIN_EMAIL`, `BOOTSTRAP_FACHADMIN_PASSWORD`	Fach-Erst-Admin (wie oben)
`BOOTSTRAP_PRIVACYADMIN_EMAIL`, `BOOTSTRAP_PRIVACYADMIN_PASSWORD`	Datenschutz-Erst-Admin (wie oben)
`BOOTSTRAP_ROOT_ADMIN`	`true` legt statt der drei Konten einen Root-Admin mit allen Rechten an (Standard `false`)
`SERVICE_CLIENT_ID`, `SERVICE_CLIENT_SECRET`	optionaler vertraulicher Service-Client
`MOBILE_REDIRECT_URI`	zusätzliche OAuth-Redirect-URI für die mobile App
`CERT_PASSWORD`	Passwort der PKCS#12-Zertifikate
`REVERSE_PROXY_NETWORK`	festes Subnetz des Compose-Netzes `web`; `X-Forwarded-*` aus diesem Bereich wird vertraut (Standard `172.30.250.0/24`)
`REVERSE_PROXY_IP`	optionale zusätzliche Proxy-Adresse, z. B. das Bridge-Gateway für einen Proxy direkt auf dem Host
`REVERSE_PROXY_ALLOWED_HOST`	optionaler Hostname des Panels; andere `X-Forwarded-Host`-Werte werden ignoriert (Standard `*`)
`HTTPS_PROXY`, `HTTP_PROXY`, `NO_PROXY`	optionaler ausgehender Proxy für Anwendung und ClamAV (siehe „Ausgehender Proxy“)
`COMPOSE_PROFILES`, `FF_SELF_UPDATE`	`updater` bzw. `true` aktivieren das Update aus dem Admin-Panel (siehe „Update aus dem Admin-Panel“)
`BLOB_ENCRYPTION_KEY_BASE64`	**Pflicht:** AES-256-Schlüssel (32 Byte Base64) für Anhänge und Datenkopien
`OFFLINE_GRANT_KEY`	privater EC-P-256-Schlüssel (PEM) zum Signieren der Offline-Grants
`OFFLINE_GRANT_KEYSET_VERSION`	monotone Version des Offline-Grant-Keysets (Standard 1)
`OFFLINE_GRANT_ADDITIONAL_PUBLIC_KEY`	zusätzlicher vertrauenswürdiger öffentlicher Schlüssel während einer Rotation
`LICENSING_INSTALLATION_KEY`	Einmal-Enrollment-Key des Lizenzservers
`FileScanning__Enabled`, `FileScanning__Host`	Malware-Scan konfigurieren (siehe unten)
`Monitoring__ScrapeToken`	**nur Lizenzserver:** Zugriffsschlüssel für `/metrics` und `/monitoring/status`; leer = beide Adressen abgeschaltet

Variable	Zweck
`Monitoring__AllowedNetworks`	**nur Lizenzserver:** zugelassene IP-Adressen oder CIDR-Bereiche, kommagetrennt; leer = jede Adresse mit gültigem Schlüssel

Blob-Verschlüsselung: Den Schlüssel mit `openssl rand -base64 32` erzeugen und **getrennt vom Blob-Archiv** sichern. Ohne ihn ist das Archiv unlesbar. In Entwicklungsumgebungen kann `BlobStorage:AllowGeneratedKey=true` gesetzt werden – in Produktion niemals.

Monitoring: Der Admin-Server wird **nicht** über die `.env` konfiguriert, sondern im Panel unter *System → Monitoring* — der Zugriffsschlüssel liegt dort verschlüsselt in der Datenbank statt im Klartext in einer Datei. Nur der Lizenzserver nutzt die beiden Variablen oben. Einzelheiten im Abschnitt „Betrieb und Überwachung".

Malware-Scan: `FileScanning__Enabled=true` plus `FileScanning__Host` aktiviert ClamAV. Ist in Produktion **nichts** konfiguriert, arbeitet der Server **fail-closed**: Uploads werden abgelehnt, statt ungeprüft angenommen zu werden.

Datenbank und Migrationen

Es gibt **keine EF-Migrations-Ordner**. Stattdessen arbeitet ein handgeschriebener `MigrationRunner`:

- **Frische Datenbank:** `EnsureCreated` baut das vollständige Schema auf.
- **Bestehende Datenbank:** Ausstehende Versionen aus dem `SchemaMigrations`-Ledger werden in **einer Transaktion** unter `pg_advisory_xact_lock` angewendet; der Versionsstempel wird in derselben Transaktion gesetzt. Konkurrierende Migratoren serialisieren sich damit selbst.
- Danach provisioniert der Migrator die Rolle `fokus_app` und beendet sich. Die App startet erst nach erfolgreichem Migrator.

Der Migrator läuft bei **jedem** `docker compose up` – ein zusätzlicher manueller Schritt ist nicht nötig.

Schemastand 1.3.0: Ledger-Version 38. Die jüngsten Schritte: 36 legt die Spalte `Devices.DeviceModel` an (Gerätemodell, höchstens 64 Zeichen), 37 die Tabellen der Schnittstellen (`IntegrationSettings`, `IntegrationClients`, `IntegrationConnections`, `IntegrationCursors`, `IntegrationOutbox`, `IntegrationDeliveries`, `IntegrationDisclosures`, `IntegrationInbox`, `IntegrationExternalReferences`, `IntegrationTokens`), 38 die Spalte `AspNetUsers."PermissionNames"`. Rechte werden dort als Namensmenge gespeichert (`,CaseEdit,CaseView,`); Schritt 38 übernimmt die bisherige Bitmaske `"Permissions"` als Namen. Derselbe Abgleich läuft bei jedem Migrator-Start (Rechte mit Alt-Bit folgen der Bitmaske, Rechte ohne Alt-Bit bleiben erhalten), damit nach einem Rollback auf ein älteres Image dort vergebene oder entzogene Rechte gelten. Behält danach ein Konto Rechte nur als Bitmaske, bricht der Migrator ab und rollt vollständig zurück. Die Spalte `"Permissions"` wird mit den eingefrorenen Alt-Bits weiter mitgeschrieben. Einzelheiten: `docs/PERMISSIONS.de.md`.

Bereiche (Ledger-Version 25): Die Tabellen `OrganizationAreaSettings` (Schalter je Organisation; fehlende Zeile = aus), `AdministrativeAreas` (eindeutiger Name je Organisation, Groß-/Kleinschreibung egal), `AreaStaffAssignments` (eine Fachkraft kann mehreren Bereichen angehören, genau einer ist Hauptbereich; Ledger-Version 27) und `AreaAdminGrants` (`AreaId` leer = organisationsweit; je Konto höchstens ein organisationsweiter Eintrag) sowie die Spalte `Teams.AreaId` (Fremdschlüssel mit `ON DELETE SET NULL`) kommen rein additiv hinzu. Es werden keine Daten migriert; nach dem Update ist der Bereichsmodus aus und das Verhalten unverändert. Die Sammel-GRANTs für `fokus_app` decken die Tabellen ab.

Sicherung und Wiederherstellung

> Ein Backup gilt erst dann als funktionierend, wenn seine **Wiederherstellung getestet** wurde. Und ein Datenbank-Backup **ohne die kryptografischen Schlüssel** ist nicht vollständig wiederherstellbar.

Was gesichert werden muss

Nr.	Gegenstand	Häufigkeit
1	**PostgreSQL** – alle Inhalte, Benutzer, Teams, Zuweisungen, Change-Feed, Audit, Richtlinien	periodisch
2	**Blob-Speicher** – Anhänge aus dem `blobs`-Volume	periodisch
3	**Signier-/Verschlüsselungszertifikate** + `CERT_PASSWORD`	selten, im Secret-Manager
4	**Offline-Grant-Schlüssel** + `OFFLINE_GRANT_KEYSET_VERSION`	selten, im Secret-Manager
5	**Data-Protection-Keyring** (`dpkeys`)	selten
6	**Lizenz-Installationszustand** (`licstate`)	selten

Punkt 2 ist ohne den **Blob-Schlüssel** (`BLOB_ENCRYPTION_KEY_BASE64`) unlesbar – sichern Sie beides **getrennt**.

Backup

```
./deploy/docker/backup.sh      # schreibt ./backups/db-<stamp>.dump + blobs-<stamp>.tgz
```

`db-*.dump` ist ein PostgreSQL-Dump im Custom-Format, direkt aus dem `db`-Container gestreamt. Sichern Sie die Ergebnisse verschlüsselt und versioniert weiter (restic, borg, Objektspeicher). Im Produkt ist **kein** statisches Backup-Passwort eingebaut.

Wiederherstellung

In einen frischen Stack (den alten zuvor mit `docker compose down -v` entfernen):

```
DB_DUMP=./backups/db-<stamp>.dump BLOBS=./backups/blobs-<stamp>.tgz ./deploy/docker/restore.sh
curl http://127.0.0.1:8080/health/ready     # erwartet: 200
```

Das Skript startet eine leere Datenbank, spielt den Dump ein, stellt die Blobs wieder her, lässt den Migrator laufen (dadurch werden ausstehende Schemaschritte angewendet und die Rolle `fokus_app` neu provisioniert – Rollen sind clusterweit und nicht Teil eines Dumps) und startet die Anwendung.

Test-Wiederherstellung

1. Beispieldaten erzeugen (z. B. einen synchronisierten Fall).
2. `./deploy/docker/backup.sh`.
3. `docker compose down -v`.
4. `restore.sh` mit Dump und Blobs ausführen.
5. Schlüssel, Zertifikate und `licstate` aus dem Secret-Manager zurückspielen.

6. Prüfen: Health ready, Anmeldung funktioniert, der Beispielfall ist vorhanden, Anhänge öffnen sich, die Prüfsummen finalisierter Dokumente stimmen.

Versionskompatibilität

Stellen Sie **in die gleiche oder eine neuere** Serverversion wieder her. Ein neuerer Server wendet ausstehende Schemaschritte beim Start an. Spielen Sie **nie** einen neueren Dump in einen älteren Server.

Aktualisierung

Upgrades sind **Backup-first** und nie destruktiv.

```
./deploy/docker/backup.sh                  # 1. sichern
docker compose pull                        # 2. neues Image holen
docker compose up -d                       # 3. Migrator + Neustart der App
curl http://127.0.0.1:8080/health/ready    # 4. Smoke-Test
```

Hinweise

- **Schemaänderungen** laufen nur über den Migrator-Schritt, nie unter der App-Rolle. Prüfen Sie vor einem schemaändernden Upgrade die Release-Notes.
- **API-Kompatibilität:** Server N unterstützt Clients N und N-1, soweit das Schema es zulässt. Über die Richtlinie **Mindest-Clientversion** sperren Sie ältere Clients weich oder hart aus. Lokale Daten bleiben den Clients immer zugänglich.
- **Rolling Upgrade:** Es wird kein kritischer Sitzungszustand im Speicher gehalten – Schlüssel, Keyring und Blobs liegen auf geteilten Volumes. Mehrere App-Instanzen hinter dem Proxy lassen sich nacheinander ersetzen; die Migrationstransaktion serialisiert sich über den Advisory-Lock. Focus Family wird als Einzelinstanz ausgeliefert.
- **Rollback:** das Vor-Upgrade-Backup in die vorherige Image-Version einspielen.

Von 1.0.0, 1.1.0 oder 1.2.0 auf 1.3.0: Zuerst den Server, dann die Apps aktualisieren; Apps ab Version 1.0 arbeiten am aktualisierten Server weiter. Der Migrator wendet die fehlenden Ledger-Versionen an: ab 1.0.0 die Versionen 22 bis 38 (Rechte `LicensePurchase`, `AreaManage` für Fach-Admins und `SsoManage` für Systemadministration, Fallverlauf, Nachweis-Archiv, Bereiche, Abrechnungsbasis, Single Sign-On, Rechnungssteller und Nummernserien, Richtlinien „Eigene Pro-Lizenz“ und „Hilfe & Kontakt“, Gerätemodell, Schnittstellen, Rechte als Namen), ab 1.1.0 die Versionen 34 bis 38, ab 1.2.0 die Versionen 36 bis 38; alle Schritte sind additiv, ein manueller SQL-Schritt ist nicht nötig. Wird die `docker-compose.yml` dieser Version übernommen, das Netz `web` einmal mit `docker compose down` (ohne `-v`) und `docker compose up -d` neu anlegen (siehe „Reverse-Proxy einrichten“). Sobald alle Geräte aktualisiert sind, unter **Richtlinien** die **Mindest-Clientversion** auf `1.3.0` setzen. Einzelheiten: `docs/deploy/UPGRADE-1.0-TO-1.1.de.md`.

Update aus dem Admin-Panel

Meldet der Versions-Feed einen neueren Build, zeigt die Übersichtsseite Administratorinnen und Administratoren mit dem Recht **System konfigurieren** die Karte **System-Update** mit der Schaltfläche **Update jetzt installieren**. Der Ablauf ist derselbe wie oben, nur automatisiert: Die App lädt das Image über TLS herunter und prüft die SHA-256-Prüfsumme; der Dienst `updater` (einziger Dienst mit Zugriff auf den Docker-Socket) sichert die Datenbank in das Volume `backups`, lädt das Image, führt den Migrator aus und startet die

App neu. Schlägt Migration oder Start fehl, stellt er automatisch das vorherige Image wieder her. Vor dem Einspielen liest der Updater Version und Build aus der Server-Assembly des signierten Images: Weichen sie vom Auftrag ab, bricht er mit `version_mismatch` ab; einen Build, der nicht neuer ist als der laufende, lehnt er mit `downgrade_refused` ab – auch dann, wenn er den laufenden Build nicht bestimmen kann. Ein Downgrade über das Panel ist damit ausgeschlossen. Daten-Volumes werden nie angefasst; der Fortschritt ist im Panel sichtbar, jeder Schritt steht im Sicherheits-Audit.

Voraussetzung: Der Dienst `updater` läuft (`docker compose ps updater`) und der Projektordner ist unter demselben Pfad wie auf dem Host eingehängt (Stack aus dem Projektordner starten oder `FF_PROJECT_DIR` in `.env` setzen). Der Updater steckt im selben Image wie das Admin-Panel und wird nur mit anderem Einstiegspunkt gestartet; nach einem erfolgreichen Update ersetzt er sich selbst durch die neue Version. Ältere Compose-Dateien bauen den Updater noch aus `deploy/docker/updater/` (`build:`); stellen Sie ihn auf `image: fokusfamilie-server:<Tag von app>` um (Vorlage: aktuelle `docker-compose.yml`) und führen Sie einmal `docker compose up -d` aus.

Der Updater ist optional, denn er läuft als root und hat Zugriff auf den Docker-Socket. Gesteuert wird er über zwei Werte in der `.env`: `COMPOSE_PROFILES=updater` startet den Dienst, `FF_SELF_UPDATE=true` lässt das Panel die Installation anbieten. Ohne Updater entfernen Sie `COMPOSE_PROFILES`, setzen `FF_SELF_UPDATE=false` und entfernen einen laufenden Updater mit `docker compose rm -sf updater`. Das Panel meldet neue Versionen dann weiterhin, bietet aber nur den Download an und weist darauf hin, dass automatische Updates abgeschaltet sind; eingespielt wird manuell wie oben beschrieben. Fehlt der Dienst, bleibt die Schaltfläche gesperrt und das Panel sagt warum. Protokoll: `docker compose logs updater`.

Manche Updates sind zu umfangreich für den automatisierten Vorgang oder betreffen Bereiche, die er nicht abdeckt. Dann zeigt das Panel weiterhin **Update verfügbar**, bietet aber nur den Download des Images an; eingespielt wird es manuell nach dem Ablauf oben. Ein laufender Vorgang lässt sich im Panel nur zurücksetzen, solange der Updater ihn noch nicht übernommen hat oder wenn der Dienst zehn Minuten kein Lebenszeichen mehr gesendet hat; ein lang laufender Schritt wird als solcher angezeigt.

Prozessorarchitektur: Das Server-Image erscheint für `linux/amd64` und `linux/arm64`; der Versions-Feed nennt beide Downloads (`admin` und `admin_arm64`). Das Panel wählt das Image passend zur eigenen Architektur, für das automatische Update wie für den Download-Link. Enthält der Feed für diese Architektur kein Image, zeigt das Panel **Update verfügbar** mit einem entsprechenden Hinweis, startet aber keinen automatischen Vorgang; eingespielt wird dann manuell, sobald das passende Image veröffentlicht ist. Der Updater lehnt zusätzlich jedes Image ab, dessen Plattform nicht zum laufenden Container passt (`platform_mismatch`).

Beta-Versionen: Unter **Richtlinien** kann nur ein Konto mit **Systemadministration** „Beta-Versionen bei Updates anzeigen“ einschalten (Standard aus, Spalte `OrganizationPolicies.AllowBetaUpdates`, Ledger-Version 32). Die Versionsabfrage an `api.software-notion.de` enthält dann `"includeBeta": true`, und der Feed darf Vorabversionen melden; das Umschalten wirkt sofort, ohne die 24-Stunden-Zwischenspeicherung abzuwarten, und steht im Sicherheits-Audit. Nur für Testinstallationen empfohlen.

Hilfe & Kontakt: Die Richtlinie „Hilfe & Kontakt in den Apps erlauben“ (Spalte `OrganizationPolicies.AllowSupportRequests`, Ledger-Version 35, Standard an) wird mit der Richtlinie an Apps und Desktop verteilt; ältere Server ohne das Feld gelten als „erlaubt“. Meldungen aus dem Panel sendet der Server per HTTPS an `https://software-notion.de/api/apps/focus-family/support-request` (überschreibbar mit `Support:Endpoint`); in Intranet-Installationen ohne Internetzugang bleibt der Weg über **Text kopieren** bzw. **Per E-Mail senden**. Die Apps senden direkt vom Gerät an dieselbe Adresse. Das Umschalten steht im Audit (`policy.support_requests_enabled` / `…_disabled`), gesendete Meldungen als `support.request_sent` ohne Text.

Lizenzierung

Nutzt Ihr Träger eine Organisationslizenz, meldet sich die Installation beim **Lizenzserver** an. Das geschieht **verzögert (lazy)**: Beim Start passiert nichts; erst beim ersten Lizenzkontakt erzeugt der Server sein EC-P-256-Schlüsselpaar, löst den **einmal verwendbaren** `LICENSING_INSTALLATION_KEY` ein und legt die entstehende Installationsidentität im Volume `licstate` ab.

> **Niemals** das Volume `licstate` auf einem bereits angemeldeten Server löschen – der Enrollment-Key lässt sich nicht erneut einlösen. Beim Umzug eines Deployments nehmen Sie `licstate` mit.

Das Verzeichnis `/var/lib/fokusfamilie/licensing` gehört im Image dem App-Benutzer (uid 1654). Ist das Volume `licstate` trotzdem nicht schreibbar (etwa root-owned), legt der Server die Identität in `dpkeys` als `licensing-installation.json` ab und meldet das als Warnung im Protokoll; beim Start liest er beide Orte, der konfigurierte Pfad hat Vorrang. Vor dem Einlösen speichert der Server eine ausstehende Anmeldung (Installations-ID und Schlüsselpaar). Bricht die Verbindung ab oder scheitert das Speichern, verwendet der nächste Versuch dieselbe Identität, und der Lizenzserver bestätigt die Wiederholung, statt den Key als verbraucht abzulehnen. Ist eine vorhandene Zustandsdatei nicht lesbar, meldet sich der Server nicht neu an, sondern zeigt einen Fehler.

Meldungen unter Lizenzierung:

Meldung (Code)	Bedeutung und Abhilfe
Key bereits verwendet (`installation.enrollment_key_used`)	Der Key wurde von einer anderen Installation eingelöst. Beim Lizenzbetreiber einen neuen Key anfordern oder den bestehenden zurücksetzen lassen, in `.env` eintragen, `app` neu starten.
Key ungültig / abgelaufen (`installation.enrollment_key_invalid`, `…_expired`)	Key vollständig und ohne Leerzeichen eintragen bzw. neuen Key anfordern.
Zeitüberschreitung (`licensing.timeout`)	Lizenzserver antwortet nicht rechtzeitig; Firewall und ausgehenden Proxy prüfen.
TLS/Zertifikat (`licensing.tls`)	Meist ein TLS-aufbrechender Proxy: dessen CA-Zertifikat nach `deploy/docker/ca/` legen.
Nicht erreichbar (`licensing.unreachable`)	DNS, Verbindung oder Proxy: Namensauflösung, `HTTPS_PROXY` / `NO_PROXY` und Firewall prüfen.
Lizenzserver oder Proxy meldet Fehler (HTTP 502/503/504)	Später erneut versuchen; dauerhaft: ausgehenden Proxy prüfen.
Identität nicht lesbar (`installation.state_unreadable`)	Dateirechte in `licstate` / `dpkeys` prüfen und sicherstellen, dass `dpkeys` und `encryption.pfx` zur Installation gehören.
Identität nicht speicherbar (`installation.state_write_failed`)	Weder `licstate` noch `dpkeys` schreibbar: Rechte (uid 1654) und Speicherplatz prüfen.

Die Ursache steht im Protokoll (`docker compose logs app | grep "Licensing:"`) mit Adresse, HTTP-Status, Ausnahmetyp und Klassifizierung. Ablauf für Betreiber, um einen verbrauchten Key zurückzusetzen:

`docs/deploy/RUNBOOK.de.md` §6.

Alternativ provisionieren Sie manuell: `--licensing-keygen` gibt eine frische Installations-ID und ein Schlüsselpaar aus; den öffentlichen Schlüssel übergeben Sie dem Betreiber des Lizenzservers.

Lizenzplätze (Seats) weist ausschließlich die Administration im Panel zu. Es gibt kein automatisches Beanspruchen durch Geräte. Ohne Platz kann ein Gerät nicht synchronisieren.

Gesponserte Seats: Der Hersteller kann einer Organisation im Lizenzserver kostenlose Seats zuweisen (optional mit Enddatum). Sie liegen an der Organisation, nicht am Abonnement, und erscheinen in keiner Rechnung. Kapazität = gekaufte Seats (bei aktivem Abonnement mit mindestens einem gekauften Seat) + gesponserte Seats; zugewiesen werden kann bis zu dieser Zahl, auch ohne Abonnement. Die Seat-Liste des Lizenzservers trennt `assignedUserIds` (alle Zuweisungen) und `licensedUserIds` (wirksam lizenziert: bei aktivem Abonnement alle, ohne Abonnement die zuerst zugewiesenen bis zur Zahl der gesponserten Seats). Das Panel sperrt den Sync nach `licensedUserIds` (ein Lizenzserver ohne das Feld: nach den Zuweisungen) und zeigt übrige Zuweisungen als „Zugewiesen, ohne Lizenz“; Anzeige und Abgleich verwaister Seats nutzen die Zuweisungen. Schnittstellen gelten auch mit nur gesponserten Seats als lizenziert.

Kompatibilität älterer Panels: Admin-Panels vor 1.3.0 können gesponserte Seats bereits zuweisen, weil der Lizenzserver die Kapazität prüft, zeigen aber nur die gekaufte Zahl an und kennen `licensedUserIds` nicht (der Sync richtet sich dort nach allen Zuweisungen).

Single Sign-On (SSO)

Optional melden sich Mitarbeitende mit ihrem Organisationskonto an, also mit demselben Konto wie bei E-Mail, Dateiablage und Windows-Anmeldung. Unterstützt werden **OpenID Connect** (OAuth 2.0 Autorisierungscode-Flow mit PKCE) und **SAML 2.0**. Damit lassen sich unter anderem Microsoft Entra ID, das lokale Active Directory über AD FS, Keycloak, Okta, Google Workspace und Authentik anbinden. LDAP wird nicht direkt angebunden.

Der Server **föderiert**: Apps (iOS, Android, Desktop) und Admin-Panel melden sich weiterhin an der Anmeldeseite dieses Servers an. Ist SSO aktiv, zeigt die Anmeldeseite zusätzlich die Schaltfläche **Mit <Name> anmelden**. Der Server leitet zum Identitätsanbieter (IdP) weiter, prüft dessen Antwort und stellt danach wie gewohnt seine eigenen Tokens aus. Lizenzplätze, Gerätebindung, Sync-Rechte und Audit bleiben unverändert. Die Apps (iOS, Android, Desktop) zeigen bei aktivem SSO unter der Passwort-Anmeldung eine eigene Schaltfläche **Mit <Name> anmelden** (ohne hinterlegten Namen: **Mit Organisationskonto anmelden**). Sie öffnet den Browser und führt direkt zum Identitätsanbieter – ohne E-Mail und Passwort in der App. Die Apps erfahren über die Server-Information von aktivem SSO (nur „SSO an“ und der Anzeigename, nie Adressen, Domains oder weitere Einstellungen) und fragen sie bei jedem Öffnen der Anmeldeansicht neu ab; nach einer Änderung im Panel erscheint oder verschwindet die Schaltfläche ohne Neuinstallation. Apps der Version 1.0 erreichen dieselbe Anmeldeseite mit der SSO-Schaltfläche über **Mit Passkey anmelden**. Gibt jemand mit einer Adresse einer erlaubten SSO-Domain in der App E-Mail und Passwort ein, wechselt die App automatisch zur Browser-Anmeldung.

Voraussetzungen

- Der Server ist unter einer **festen HTTPS-Adresse** erreichbar, die Browser und Apps verwenden. Diese Adresse steckt in Redirect-URI, ACS-URL und Entity-ID. Hinter einem Reverse-Proxy muss `ReverseProxy:Enabled` gesetzt sein (siehe *Reverse-Proxy einrichten*), sonst erzeugt der Server falsche Adressen.

- **Ausgehender HTTPS-Zugriff** vom Server zum Identitätsanbieter: bei OpenID Connect auf Discovery-Dokument, Signaturschlüssel (JWKS) und Token-Endpunkt; bei SAML auf die Metadaten-URL – oder Sie fügen das Metadaten-XML ein, dann braucht SAML keinen ausgehenden Zugriff. Der Identitätsanbieter muss den Server **nie** selbst erreichen.
- Eine korrekt gehende **Uhr** (NTP) auf dem Server: Token- und Assertion-Gültigkeit sowie die erneute Anmeldebestätigung werden zeitlich geprüft.
- Ein **lokales** Konto mit dem Recht **„Single Sign-On verwalten“** (`SsoManage`) und Passwort. Technische Administration mit Systemadministration hat dieses Recht automatisch. SSO-Konten können die SSO-Einstellungen nie ändern – so kann ein kompromittierter Identitätsanbieter die Anmeldung nicht umkonfigurieren.
- Mindestens ein **Notfallzugang**: ein aktives lokales Systemadministrationskonto mit starkem Passwort und MFA (siehe *Notfallzugang und „SSO erzwingen“*).
- Administrative Rechte beim Identitätsanbieter, um dort eine Anwendung zu registrieren.

OpenID Connect oder SAML 2.0?

Situation	Empfehlung
Microsoft Entra ID, Keycloak, Okta, Google, Authentik	**OpenID Connect** – einfacher einzurichten, E-Mail-Bestätigung wird mitgeliefert, Erstverknüpfung vorbereiteter Konten ohne Zusatzschritt
Lokales Active Directory über AD FS	beides möglich; **SAML** ist bei AD FS der verbreitetere Weg, OpenID Connect braucht AD FS 2016 oder neuer
Google Workspace mit zentral verwalteter Admin-Konsole	**SAML** (App in der Admin-Konsole) oder OpenID Connect über ein Google-Cloud-Projekt
Server ohne ausgehenden Internetzugang, IdP nur im Intranet	**SAML** mit eingefügtem Metadaten-XML (kein ausgehender Abruf nötig)
Vorgabe der IT (nur SAML-Anwendungen erlaubt)	**SAML**

Bei SAML meldet der Identitätsanbieter nie, ob eine E-Mail-Adresse bestätigt ist. Vorbereitete Konten werden dann erst nach **Erstanmeldung freigeben** oder mit der Option **E-Mail-Adressen des Identitätsanbieters auch ohne Bestätigungsmerkmal vertrauen** verknüpft (siehe *Benutzerkonten und Verknüpfung*).

Die Seite „Single Sign-On“

Panel → **Single Sign-On**. Die Seite zeigt **nur die Felder des gewählten Protokolls**. Wechseln Sie das Protokoll in der Auswahl, blendet die Seite sofort die passenden Felder, Adressen und Hinweise ein. Gespeichert werden nur die Felder des gewählten Protokolls; die Werte des anderen Protokolls bleiben gespeichert und erscheinen wieder, wenn Sie zurückwechseln. „Verbindung testen“ und die Adressen beziehen sich auf die **gespeicherten** Einstellungen – nach einem Protokollwechsel zuerst speichern.



Single Sign-On mit OpenID Connect (Beispiel Microsoft Entra ID): Redirect-URI, Authority, Client-ID, Client-Geheimnis und Scopes.



Single Sign-On mit SAML 2.0 (Beispiel AD FS): ACS-URL, Entity-ID und Metadaten dieses Servers, Metadaten des Identitätsanbieters und SAML-Attribute. Der Abschnitt „Konten und Richtlinie“ entspricht dem bei OpenID Connect.

Status und Adressen

Das Badge zeigt **Aktiv** (eingeschaltet und vollständig), **Eingeschaltet, aber unvollständig** oder **Aus**. Darunter stehen die Adressen, die Sie beim Identitätsanbieter eintragen. Sie entsprechen der Adresse, unter der Sie das Panel gerade aufrufen – rufen Sie das Panel deshalb über die offizielle Adresse auf, nicht über `localhost` oder eine IP.

Protokoll	Angezeigt	Beispiel
OpenID Connect	Redirect-URI	`https://ff.example.org/account/sso/oidc/callback`
SAML 2.0	Assertion Consumer Service (HTTP-POST)	`https://ff.example.org/account/sso/saml/acs`
SAML 2.0	Entity-ID dieses Servers	Standard `https://ff.example.org/account/sso/saml/metadata`
SAML 2.0	SAML-Metadaten dieses Servers (Link zum Herunterladen)	`https://ff.example.org/account/sso/saml/metadata`

Die SAML-Metadaten dieses Servers sind erst abrufbar, wenn SAML 2.0 gespeichert ist. Sie enthalten Entity-ID, ACS-URL und das Zertifikat dieses Servers; viele Identitätsanbieter (AD FS, Keycloak, Entra ID) können eine Anwendung direkt daraus anlegen.

Gemeinsame Felder

Feld	Bedeutung	Beispiel
Single Sign-On einschalten	Zeigt die SSO-Schaltfläche auf der Anmeldeseite. Beim Einschalten prüft der Server, ob das gewählte Protokoll vollständig ist.	–
Protokoll	OpenID Connect oder SAML 2.0	OpenID Connect
Name auf der Anmeldeschaltfläche	Text der Schaltfläche „Mit … anmelden“; leer = „SSO“	Microsoft Entra ID
Claim für E-Mail / Anzeigename / Gruppen	Nur ausfüllen, wenn der Anbieter vom Üblichen abweichende Namen verwendet (siehe *Claim-Zuordnung*). Gilt für beide Protokolle – nach einem Protokollwechsel prüfen.	leer
Nur Mitglieder dieser Gruppen zulassen	Optional, eine Gruppe je Zeile, genau wie der Anbieter sie im Gruppen-Claim liefert (bei Entra die Objekt-ID). Ohne Treffer: `sso_denied`.	`5b7c…-…-…`
Erlaubte E-Mail-Domänen	Optional, durch Komma getrennt. Anmeldungen mit anderer oder fehlender E-Mail-Adresse werden abgelehnt.	`example.org, example.de`
Konten bei der ersten Anmeldung automatisch anlegen (Just-in-time)	Unbekannte Personen erhalten ein Konto ohne Rechte und ohne Lizenzplatz.	aus
E-Mail-Adressen des Identitätsanbieters auch ohne Bestätigungsmerkmal vertrauen	Siehe *Benutzerkonten und Verknüpfung* – Risiko beachten.	aus
SSO für alle Konten außer lokalen Systemadministrationskonten erzwingen	Lokale Passwort-Anmeldung nur noch für den Notfallzugang.	aus
Aktuelles Passwort (ggf. Code oder Passkey)	Erneute Anmeldebestätigung; ohne sie wird nichts gespeichert.	–

Felder für OpenID Connect

Feld	Bedeutung	Beispiel
Authority / Aussteller-URL	Basisadresse des Anbieters; der Server liest `<Authority>/.well-known/openid-configuration`. Nur HTTPS.	`https://login.microsoftonline.com/<Mandanten-ID>/v2.0`
Client-ID (Anwendungs-ID)	Kennung der beim Anbieter registrierten Anwendung	`8d6b1e2f-4c3a-…`
Client-Geheimnis	Geheimer Clientschlüssel. Wird verschlüsselt gespeichert und nie wieder angezeigt; leer lassen behält das gespeicherte Geheimnis, **Gespeichertes Client-Geheimnis löschen** entfernt es.	–
Scopes	Durch Leerzeichen getrennt; `openid` wird immer ergänzt. Standard `openid profile email`.	`openid profile email`

Der Server verwendet immer den Autorisierungscode-Flow mit PKCE (`response_mode=query`) und prüft Signatur, Aussteller, Zielgruppe, Ablauf und Nonce. Die Benutzerdaten stammen aus dem ID-Token; der UserInfo-Endpunkt wird nicht abgefragt – benötigte Claims müssen also im ID-Token stehen.

Felder für SAML 2.0

Feld	Bedeutung	Beispiel
Metadaten-URL des Identitätsanbieters	Wird abgerufen und eine Stunde zwischengespeichert („Verbindung testen“ lädt sie neu). Nur HTTPS.	`https://adfs.example.org/FederationMetadata/2007-06/FederationMetadata.xml`
Oder: Metadaten-XML des Identitätsanbieters	Für Server ohne Zugriff auf die Metadaten-URL. Muss einen HTTP-Redirect-Anmeldeendpunkt und ein Signaturzertifikat enthalten. Bei einem Zertifikatswechsel des Anbieters neu einfügen.	`<EntityDescriptor …>`
Entity-ID dieses Servers	Kennung dieses Servers beim Anbieter (auch „Bezeichner“, „Audience“, „Vertrauensstellungsbezeichner“). Leer = Adresse der SAML-Metadaten dieses Servers.	leer
Anmeldeanfragen signieren	Nur einschalten, wenn der Anbieter signierte AuthnRequests verlangt.	aus

Anmeldeanfragen gehen per HTTP-Redirect an den Anbieter, die Antwort kommt per HTTP-POST an die ACS-URL. Antwort bzw. Assertion müssen mit einem Zertifikat aus den Metadaten signiert sein; verschlüsselte Assertions werden mit dem Zertifikat dieses Servers entschlüsselt. Vom Anbieter gestartete Anmeldungen (IdP-initiiert, etwa über eine Kachel im Microsoft-Portal) werden abgelehnt – die Anmeldung beginnt immer auf der Anmeldeseite von Focus Family. Als **NameID** verwenden Sie ein **dauerhaftes** Format (persistent oder eine unveränderliche Kennung). Eine transiente NameID wechselt bei jeder Anmeldung und verhindert die Zuordnung (Ausnahme: Microsoft Entra ID, dort zählen `tenantid` + `objectidentifier`).

Zertifikat dieses Servers (SAML): Beim ersten Speichern von SAML erzeugt der Server ein selbstsigniertes Zertifikat (RSA 3072, fünf Jahre gültig). Es dient zum Signieren von Anfragen und zum Entschlüsseln von Assertions und steht in den SAML-Metadaten dieses Servers. **Neues Zertifikat erzeugen** (mit Anmeldebestätigung) tauscht es aus; danach die neuen Metadaten beim Anbieter hinterlegen, sonst scheitern signierte Anfragen und verschlüsselte Assertions.

Claim-Zuordnung

Leer lassen genügt fast immer. Der Server sucht dann in dieser Reihenfolge:

Wert	OpenID Connect	SAML 2.0
E-Mail	`email`, `preferred_username`, `upn`, `unique_name`	`…/claims/emailaddress`, `email`, `mail`, `urn:oid:0.9.2342.19200300.100.1.3`, `…/claims/upn`, `upn`, sonst eine NameID im E-Mail-Format
Anzeigename	`name`, `preferred_username`, sonst `given_name` + `family_name`	`…/identity/claims/displayname`, `displayName`, `name`, `urn:oid:2.16.840.1.113730.3.1.241`, `cn`, sonst Vor- und Nachname
Gruppen	`groups`, `roles`	`…/claims/groups`, `groups`, `memberOf`, `Role`, `role`

Tragen Sie einen eigenen Namen ein, wird **nur** dieser verwendet. Bei SAML ist das meist eine URI, z. B. `http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress`. Gruppen-Claims sind nur so vertrauenswürdig wie ihre Quelle: verwenden Sie vom Anbieter verwaltete Gruppenmitgliedschaften, nie Attribute, die Benutzer selbst ändern können.

Verbindung testen

Verbindung testen prüft die gespeicherten Einstellungen: bei OpenID Connect, ob das Discovery-Dokument abrufbar ist und Aussteller, Anmelde- und Token-Endpunkt enthält; bei SAML, ob die Metadaten abrufbar bzw. lesbar sind und ein gültiges Signaturzertifikat enthalten. Client-ID und Client-Geheimnis prüft erst eine echte Anmeldung – testen Sie deshalb abschließend mit einem Testkonto in einem privaten Browserfenster.

Konfiguration per Datei oder Umgebungsvariablen

Ist `Sso:Enabled` in `appsettings` oder als Umgebungsvariable gesetzt, stammt die gesamte SSO-Konfiguration aus der Datei und das Panel zeigt sie nur an:

```
Sso__Enabled=true
Sso__Protocol=oidc
Sso__DisplayName=Microsoft Entra ID
Sso__Oidc__Authority=https://login.microsoftonline.com/<Mandanten-ID>/v2.0
Sso__Oidc__ClientId=<Anwendungs-ID>
Sso__Oidc__ClientSecret=<Geheimnis>
Sso__Oidc__Scopes=openid profile email
Sso__Saml__IdpMetadataUrl=https://adfs.example.org/FederationMetadata/2007-
06/FederationMetadata.xml
Sso__Saml__IdpMetadataXml=
Sso__Saml__SpEntityId=
Sso__Saml__SignAuthnRequests=false
Sso__Claims__Email=
Sso__Claims__Name=
Sso__Claims__Groups=
Sso__RequiredGroups=<Gruppe 1>;<Gruppe 2>
Sso__AllowedEmailDomains=example.org
Sso__JitProvisioning=false
Sso__TrustIdpEmail=false
Sso__EnforceSso=false
```

`Sso__Protocol` ist `oidc` oder `saml`. `Sso__AllowInsecureHttp=true` erlaubt `http://`-Adressen für Authority und Metadaten – nur für Testumgebungen.

Microsoft Entra ID

OpenID Connect (empfohlen):

1. Entra Admin Center → **App-Registrierungen** → **Neue Registrierung**: Name „Focus Family“, unterstützte Kontotypen **Nur Konten in diesem Organisationsverzeichnis**, Plattform **Web**, Redirect-URI aus dem Panel (`https://<server>/account/sso/oidc/callback`).
2. **Übersicht**: *Anwendungs-ID (Client)* und *Verzeichnis-ID (Mandant)* notieren.
3. **Zertifikate & Geheimnisse** → **Neuer geheimer Clientschlüssel**; den *Wert* sofort kopieren (er wird nur einmal angezeigt) und das Ablaufdatum im Kalender vormerken.
4. **Tokenkonfiguration** → **Optionalen Anspruch hinzufügen** → Tokentyp **ID**: `email`, `upn` und `xms_edov`. Der Claim `upn` ist für die Erstverknüpfung vorbereiteter Konten erforderlich; `email` mit `xms_edov: true` (Domäne vom Mandanten bestätigt) wird ebenfalls als bestätigt anerkannt. Für eine Gruppenbeschränkung zusätzlich **Gruppenanspruch hinzufügen** (Sicherheitsgruppen, im ID-Token als Gruppen-ID).
5. **API-Berechtigungen**: Microsoft Graph, delegiert: `openid`, `profile`, `email`; **Administratorzustimmung erteilen**.
6. Optional: **Unternehmensanwendungen** → Focus Family → **Eigenschaften** → *Zuweisung erforderlich* = Ja, dann unter **Benutzer und Gruppen** die berechtigten Personen oder Gruppen zuweisen.
7. Im Panel: Protokoll **OpenID Connect**, Name „Microsoft Entra ID“, Authority `https://login.microsoftonline.com/<Verzeichnis-ID>/v2.0` (**mandantenspezifisch**, nie `common`, `organizations` oder `consumers`), Client-ID = Anwendungs-ID, Client-Geheimnis, Scopes `openid`

`profile email`. Speichern, **Verbindung testen**, mit einem Testkonto anmelden, dann **Single Sign-On einschalten**.

8. Gruppenbeschränkung: die Objekt-IDs der Gruppen unter **Nur Mitglieder dieser Gruppen zulassen** eintragen (eine je Zeile). Ist eine Person in sehr vielen Gruppen (mehr als 200), sendet Entra keine Gruppen im Token – beschränken Sie dann besser über *Zuweisung erforderlich*.

Entra sendet für Arbeitskonten kein `email_verified`. Focus Family erkennt die Adresse trotzdem als bestätigt an, wenn das Token von einem mandantenspezifischen Aussteller stammt (`https://login.microsoftonline.com/<tid>/v2.0`), die Mandanten-ID `tid` zur konfigurierten Authority passt und die Adresse aus `upn` oder aus `email` mit `xms_edov: true` stammt. `preferred_username` gilt nie als bestätigt. Entra-Konten werden über `tid` + `oid` zugeordnet – ein späterer Wechsel zwischen OpenID Connect und SAML behält die Verknüpfung. Erneuern Sie das Client-Geheimnis rechtzeitig vor Ablauf (neues Geheimnis anlegen, im Panel eintragen, altes löschen).

SAML:

1. **Unternehmensanwendungen** → **Neue Anwendung** → **Eigene Anwendung erstellen** („Nicht-Katalog-Anwendung“) → **Einmaliges Anmelden** → **SAML**.
2. *Grundlegende SAML-Konfiguration*: **Bezeichner (Entitäts-ID)** = Entity-ID aus dem Panel, **Antwort-URL** = ACS-URL aus dem Panel. Alternativ *Metadatendatei hochladen* mit den SAML-Metadaten dieses Servers.
3. *Attribute & Ansprüche*: Die Standardansprüche (emailaddress, name, givenname, surname, objectidentifier, tenantid) genügen. Für die Erstverknüpfung über die E-Mail-Adresse den Anspruch *emailaddress* auf `user.userprincipalname` setzen. Für Gruppen **Gruppenanspruch hinzufügen**.
4. *SAML-Zertifikate*: die **App-Verbundmetadaten-URL** kopieren und im Panel als Metadaten-URL eintragen.
5. **Benutzer und Gruppen** zuweisen.
6. Im Panel: Protokoll **SAML 2.0**, Metadaten-URL, Entity-ID leer lassen (oder den eingetragenen Bezeichner), speichern, testen, einschalten. Vorbereitete Konten entweder je Konto mit **Erstanmeldung freigeben** verknüpfen oder **E-Mail-Adressen des Identitätsanbieters auch ohne Bestätigungsmerkmal vertrauen** einschalten – vertretbar, weil Benutzer ihren UPN in Entra nicht selbst ändern können.

Lokales Active Directory über AD FS

AD FS 2016 oder neuer veröffentlicht sowohl SAML als auch OpenID Connect.

SAML (Vertrauensstellung der vertrauenden Seite):

1. AD-FS-Verwaltung → **Vertrauensstellungen der vertrauenden Seite** → **Vertrauensstellung hinzufügen** → *Ansprüche unterstützend* → **Daten über die vertrauende Seite, die online oder in einem lokalen Netzwerk veröffentlicht sind, importieren** mit `https://<server>/account/sso/saml/metadata` (oder die heruntergeladene Metadatendatei importieren). Damit übernimmt AD FS Entity-ID, ACS-URL und das Zertifikat dieses Servers.
2. Zugriffssteuerungsrichtlinie wählen, etwa *Bestimmte Gruppe zulassen*.
3. **Anspruchsausstellungsrichtlinie bearbeiten** → Regel *LDAP-Attribute als Ansprüche senden*, Attributspeicher *Active Directory*: `E-Mail-Addresses` → *E-Mail-Adresse*, `Display-Name` → *Name*, `Token-Groups – Unqualified Names` → *Gruppe*.

4. Zweite Regel *Eingehenden Anspruch transformieren*: *E-Mail-Adresse* → *Namens-ID*, Format **Beständiger Bezeichner** (persistent). Alternativ eine unveränderliche Kennung (z. B. `objectGUID`) als persistente NameID.
5. Im Panel: Protokoll **SAML 2.0**, Metadaten-URL `https://<adfs-host>/FederationMetadata/2007-06/FederationMetadata.xml` (ohne ausgehenden Zugriff: Datei öffnen und das XML einfügen), Gruppen-Claim leer lassen oder `http://schemas.microsoft.com/ws/2008/06/identity/claims/groups`, unter **Nur Mitglieder dieser Gruppen zulassen** die Gruppennamen. Da die Adressen aus dem Verzeichnis stammen und Benutzer sie nicht selbst ändern, kann **E-Mail-Adressen des Identitätsanbieters auch ohne Bestätigungsmerkmal vertrauen** eingeschaltet werden.

OpenID Connect:

1. AD-FS-Verwaltung → **Anwendungsgruppen** → **Anwendungsgruppe hinzufügen** → Vorlage **Serveranwendung, die auf eine Web-API zugreift**.
2. Serveranwendung: Redirect-URI aus dem Panel, **Clientbezeichner** notieren, unter *Anwendungsanmeldeinformationen* **Gemeinsamen geheimen Schlüssel generieren** und kopieren.
3. Web-API: als Bezeichner den Clientbezeichner eintragen; Zugriffssteuerungsrichtlinie wählen; erlaubte Bereiche `openid`, `profile`, `email` und `allatclaims`.
4. In der Web-API unter **Ausstellungstransformationsregeln** eine Regel *LDAP-Attribute als Ansprüche senden*: `E-Mail-Addresses` → *E-Mail-Adresse*, `User-Principal-Name` → *UPN*, `Display-Name` → *Name*, bei Bedarf `Token-Groups – Unqualified Names` → *Gruppe*.
5. Im Panel: Protokoll **OpenID Connect**, Authority `https://<adfs-host>/adfs`, Client-ID = Clientbezeichner, Client-Geheimnis, Scopes `openid profile email allatclaims`. AD FS sendet kein `email_verified`: Vorbereitete Konten mit **Erstanmeldung freigeben** verknüpfen oder **E-Mail-Adressen des Identitätsanbieters auch ohne Bestätigungsmerkmal vertrauen** einschalten. Bei Gruppen den Gruppen-Claim `group` eintragen, falls AD FS ihn so benennt.

Keycloak

- **OpenID Connect:** Realm → **Clients** → **Create client**, Typ *OpenID Connect*, Client-ID z. B. `focus-family`; *Client authentication* **an**, *Standard flow* **an**, alle anderen Flows aus; **Valid redirect URIs** = Redirect-URI aus dem Panel; unter *Advanced* → *Proof Key for Code Exchange* = `S256`. Unter **Credentials** das Client-Geheimnis kopieren. Gruppen: **Client scopes** → `focus-family-dedicated` → **Add mapper** → *Group Membership*, Token Claim Name `groups`, *Full group path* aus, *Add to ID token* an. Im Panel: Authority `https://<keycloak>/realms/<realm>`, Client-ID, Geheimnis. Keycloak sendet `email_verified`; prüfen Sie, dass E-Mail-Adressen im Realm verifiziert sind (*Realm settings* → *Login* → *Verify email*) und Benutzer ihre Adresse nicht ohne Bestätigung ändern können.
- **SAML: Create client**, Typ *SAML*, Client-ID = Entity-ID aus dem Panel; **Valid redirect URIs** und *Assertion Consumer Service POST Binding URL* = ACS-URL; *Name ID format* `persistent`; *Sign assertions* an; *Client signature required* nur an, wenn im Panel **Anmeldeanfragen signieren** aktiv ist (Zertifikat dieses Servers dann unter *Keys* importieren). Mapper *User Property* `email` → SAML-Attribut `email`. Metadaten im Panel: `https://<keycloak>/realms/<realm>/protocol/saml/descriptor`.

Okta

- **OpenID Connect: Applications** → **Create App Integration** → *OIDC – OpenID Connect*, *Web Application*; **Sign-in redirect URIs** = Redirect-URI aus dem Panel; Grant type *Authorization Code*; Personen oder Gruppen zuweisen. Im Panel: Authority `https://<org>.okta.com` (oder ein eigener Autorisierungsserver, z. B. `https://<org>.okta.com/oauth2/default`), Client-ID, Client-Geheimnis.

Gruppen: unter *Sign On → OpenID Connect ID Token → Groups claim* einen Filter setzen (Claim `groups`).

- **SAML: Create App Integration** → *SAML 2.0*; *Single sign-on URL* = ACS-URL, *Audience URI (SP Entity ID)* = Entity-ID, *Name ID format* `Persistent`, Attribut `email` = `user.email`, optional *Group Attribute Statement* `groups`. Die *Metadata URL* der App im Panel eintragen.

Google Workspace

- **SAML (Admin-Konsole): Apps → Web- und mobile Apps → App hinzufügen → Benutzerdefinierte SAML-App hinzufügen**. In Schritt *Google-IdP-Details* **Metadaten herunterladen**. Bei *Dienstanbieterdetails*: **ACS-URL** und **Entitäts-ID** aus dem Panel, *Name-ID-Format* `EMAIL`, *Name-ID* = *Basic Information › Primary email*. Attributzuordnung: *Primary email* → `email`, *First name* → `givenName`, *Last name* → `sn`. Die App unter **Nutzerzugriff** für die gewünschten Organisationseinheiten aktivieren. Im Panel: Protokoll **SAML 2.0**, heruntergeladenes Metadaten-XML einfügen. Google verwendet die primäre E-Mail-Adresse als NameID; wird sie umbenannt, ändert sich die Kennung (dann **Verknüpfung lösen** und neu verknüpfen). Da die Adressen in der Admin-Konsole verwaltet werden, kann **E-Mail-Adressen des Identitätsanbieters auch ohne Bestätigungsmerkmal vertrauen** eingeschaltet werden.
- **OpenID Connect (Google Cloud):** Projekt → **APIs & Dienste → OAuth-Zustimmungsbildschirm** (Typ *Intern*) → **Anmeldedaten → OAuth-Client-ID** → *Webanwendung*, **Autorisierte Weiterleitungs-URIs** = Redirect-URI. Im Panel: Authority `https://accounts.google.com`, Client-ID, Client-Geheimnis und unbedingt **Erlaubte E-Mail-Domänen** auf die eigene Domäne setzen (sonst könnte sich jedes Google-Konto anmelden, für das ein Konto vorbereitet ist).

Andere Anbieter

Jeder OpenID-Connect-Anbieter mit Discovery-Dokument (`<Authority>/.well-known/openid-configuration`) und Autorisierungscode-Flow, bzw. jeder SAML-2.0-Anbieter mit Metadaten, HTTP-Redirect-Anmeldeendpunkt und signierten Antworten funktioniert (z. B. Authentik, Shibboleth, PingFederate, JumpCloud). Allgemein: Anwendung als vertraulichen Web-Client bzw. als Service Provider registrieren, die Adressen aus dem Panel eintragen, E-Mail-Adresse und Namen ins ID-Token bzw. als Attribute ausgeben, bei SAML eine dauerhafte NameID wählen.

Benutzerkonten und Verknüpfung

Art	Anmeldung	Passwort, MFA, Passkeys
Lokales Konto	E-Mail und Passwort (ggf. MFA oder Passkey)	im Panel verwaltet
SSO-Konto (Badge **SSO**)	nur über den Identitätsanbieter	beim Identitätsanbieter; im Panel und in der API gesperrt

Zuordnung: Ein SSO-Konto wird dauerhaft über die Kennung des Anbieters zugeordnet – OpenID Connect `iss` + `sub`, SAML Issuer + NameID, Microsoft Entra ID in beiden Protokollen `tid` + `oid`. Die E-Mail-Adresse dient nur der **Erstverknüpfung**; spätere Adressänderungen beim Anbieter sind unkritisch.

Konten vorab anlegen (empfohlen): Benutzer → **Neues SSO-Konto** (Name, E-Mail-Adresse bzw. UPN, Rechte, Bereich). Rechte und Lizenzplatz können Sie sofort vergeben. Bis zur ersten Anmeldung steht am Badge **ausstehend**. Bei der ersten Anmeldung verknüpft der Server das Konto über die E-Mail-Adresse – aber nur, wenn sie als **bestätigt** gilt:

Anbieter	Gilt als bestätigt
OpenID Connect allgemein	Claim `email` zusammen mit `email_verified: true`
Microsoft Entra ID (OpenID Connect)	`upn` oder `email` mit `xms_edov: true`, nur bei mandantenspezifischer Authority und passendem `tid`
SAML 2.0	nie – SAML kennt kein Bestätigungsmerkmal
Option **E-Mail-Adressen des Identitätsanbieters auch ohne Bestätigungsmerkmal vertrauen**	jede vom Anbieter gelieferte Adresse

Ist die Adresse nicht bestätigt, lehnt der Server die Anmeldung mit `sso_email_unverified` ab. Dann gibt eine Administration mit Benutzerverwaltung die Erstanmeldung frei: Panel → **Single Sign-On** → **Konten ohne Organisationsanmeldung** → **Erstanmeldung freigeben** (bereichsgebundene Administration nutzt dafür die Benutzerseite). Die nächste Anmeldung mit passender E-Mail-Adresse innerhalb von **24 Stunden** wird **einmalig** verknüpft. Geben Sie nur frei, wenn die Person sich gleich anmeldet. Die Freigabe erlischt bei der Verknüpfung, beim Lösen der Verknüpfung und beim Umstellen des Anmeldeverfahrens.

E-Mail-Adressen des Identitätsanbieters auch ohne Bestätigungsmerkmal vertrauen (Standard aus) behandelt jede Adresse des Anbieters als bestätigt – für Erstverknüpfung und Just-in-time. Schalten Sie das nur ein, wenn Benutzer ihre E-Mail-Adresse beim Anbieter **nicht selbst ändern** können (AD FS mit Adressen aus dem Active Directory, Entra-SAML mit `user.userprincipalname`, Google Workspace). **Risiko:** Kann jemand die eigene Adresse beim Anbieter frei wählen (Self-Service-Profil, Social Login, Keycloak ohne E-Mail-Bestätigung), übernimmt er damit ein vorbereitetes fremdes Konto samt Rechten.

Just-in-time (Standard aus) legt unbekannte Personen bei der ersten Anmeldung als Fachkraft **ohne Rechte und ohne Lizenzplatz** an; bis die Administration beides vergibt, ist keine App-Anmeldung möglich. Auch Just-in-time setzt eine bestätigte Adresse (oder die Vertrauensoption) voraus. Ohne Just-in-time erhalten unbekannte Personen `sso_unknown`.

Erlaubte E-Mail-Domänen und Gruppen schränken jede Anmeldung ein. Für ein bereits verknüpftes Konto mit unbestätigter Adresse muss die Domäne der im Konto gespeicherten Adresse erlaubt sein.

Lizenz: Lizenzplätze werden SSO-Konten genauso zugewiesen wie lokalen Konten (Panel → Lizenzierung).

Anmeldeverfahren umstellen und Verknüpfung lösen

Im Menü einer Person unter **Anmeldeverfahren** (Recht Benutzerverwaltung; technische Konten zusätzlich „Technische Admins verwalten“):

- **Auf SSO umstellen:** Passwort, MFA-Schlüssel, Passkeys und offene Einrichtungscodes werden entfernt, alle Sitzungen und Geräte-Anmeldungen beendet. Rechte und Lizenzplatz bleiben. Die E-Mail-Domäne muss erlaubt und SSO aktiv sein. Ein lokales Konto mit gleicher E-Mail-Adresse wird **nie automatisch** übernommen (Meldung `sso_local`), sondern nur so umgestellt.
- **Auf lokale Anmeldung umstellen:** entfernt die Verknüpfung und beendet alle Sitzungen; anschließend ein Passwort vergeben.
- **Verknüpfung lösen:** entfernt nur die Verknüpfung (etwa nach einem Anbieterwechsel). Die nächste Anmeldung verknüpft neu – über eine bestätigte Adresse oder nach **Erstanmeldung freigeben**.
- Das eigene Anmeldeverfahren lässt sich nicht selbst umstellen.

Protokoll oder Anbieter wechseln: Die Kennung hängt von Protokoll und Anbieter ab. Bei Microsoft Entra ID bleibt sie beim Wechsel zwischen OpenID Connect und SAML gleich (`tid` + `oid`). Bei allen anderen Anbietern

ändert sich die Kennung (`iss` + `sub` gegenüber Issuer + NameID); verknüpfte Konten erhalten dann `sso_conflict`. Lösen Sie in diesem Fall vor oder nach dem Wechsel die Verknüpfungen (**Verknüpfung lösen**) und lassen Sie neu verknüpfen. Prüfen Sie außerdem die Claim-Zuordnung, sie gilt für beide Protokolle.

SSO ausschalten: SSO-Konten bleiben gesperrt (fail-closed), bis sie auf lokale Anmeldung umgestellt werden.

Notfallzugang und „SSO erzwingen“

- Lokale Konten mit **Systemadministration**, die nicht auf SSO umgestellt sind, melden sich im **Admin-Panel immer** mit Passwort an – auch bei erzwungenem SSO oder ausgefallenem Identitätsanbieter. Der Notfallzugang ist ein reiner Panel-Weg: Bei erzwungenem SSO erhält auch ein solches Konto in den Apps nur noch per SSO Zugang; Passwort- und Passkey-Sitzungen bekommen keine App-Tokens, bestehende App-Anmeldungen enden bei der nächsten Token-Erneuerung. Halten Sie mindestens ein solches Konto mit starkem Passwort und MFA vor und bewahren Sie die Zugangsdaten sicher auf (Tresor, Vier-Augen-Prinzip).
- Das letzte aktive lokale Systemadministrationskonto mit Passwort lässt sich nicht auf SSO umstellen. Ohne ein solches Konto lässt sich **SSO erzwingen** nicht einschalten; die Seite warnt dann.
- **SSO erzwingen** sperrt die Passwort-Anmeldung aller übrigen lokalen Konten (Panel, App, API: `sso_only`). Stellen Sie diese Konten vorher auf SSO um. Die Apps zeigen bei erzwungenem SSO in Serververbindung, erneuter Anmeldung und Entsperren mit Zugangsdaten kein E-Mail-/Passwortfeld und keine Passkey-Schaltfläche mehr, sondern nur **Mit … anmelden** und den Hinweis „Die Anmeldung erfolgt über das Organisationskonto.“ Ohne Erzwingung bleibt das Formular mit der zusätzlichen SSO-Schaltfläche. Beim Einschalten werden bestehende lokale Sitzungen und App-Tokens dieser Konten sofort beendet.
- **Passwort-Versuche mit einer Adresse einer erlaubten SSO-Domain** (Feld „Erlaubte E-Mail-Domains“, SSO aktiv) ergeben immer `sso_only` – gleich, ob das Konto existiert, SSO-verwaltet, deaktiviert oder gesperrt ist oder das Passwort falsch ist. Anmeldeseite und Apps führen so alle Personen dieser Domains zur SSO-Anmeldung, ohne zu verraten, ob ein Konto existiert. Nur nicht gesperrte Konten (Notfallzugang bzw. lokale Konten ohne „SSO erzwingen“) melden sich mit korrektem Passwort an. Für Adressen außerhalb dieser Domains sowie bei ausgeschaltetem SSO oder ohne eingetragene Domains erscheint wie bisher die allgemeine Fehlermeldung; `sso_only` folgt dann nur auf ein korrektes Passwort.
- Für SSO-Konten sind Passwort-Anmeldung, Passwort setzen und ändern, MFA-Einrichtung, Passkeys und Einrichtungs-QR-Codes serverseitig gesperrt. Die globale MFA-Pflicht gilt als erfüllt; MFA erzwingen Sie beim Identitätsanbieter (z. B. Entra-Richtlinien für bedingten Zugriff).

Erneute Anmeldebestätigung

Geschützte Aktionen verlangen eine frische Anmeldebestätigung. Lokale Konten bestätigen mit Passwort (ggf. Code oder Passkey). SSO-Konten (etwa im Bereichsmodus) bestätigen über **Jetzt erneut anmelden**: Der Server fordert eine vollständige Neuanmeldung beim Anbieter an – bei OpenID Connect mit `prompt=login` und `max_age=0` und wertet den Claim `auth_time` aus, bei SAML mit `ForceAuthn="true"` und wertet `AuthnInstant` aus. Der gemeldete Anmeldezeitpunkt darf höchstens **5 Minuten** alt und nicht älter als der Beginn der Bestätigung sein (1 Minute Uhrentoleranz); die Bestätigung gilt danach **10 Minuten**. Eine normale SSO-Anmeldung zählt nie als Bestätigung. Liefert der Anbieter `auth_time` bzw. `AuthnInstant` nicht oder ignoriert er die Neuanmeldung (etwa durch eine bestehende Sitzung), erscheint `sso_reauth`.

Betrieb im Intranet

SSO arbeitet ausschließlich mit **Browser-Weiterleitungen** (Front-Channel): Der Browser wird zum Anbieter geschickt und kommt mit dem Ergebnis zurück. Der Anbieter ruft den Server nie direkt auf; Panel und Server

können also nur im internen Netz oder per VPN erreichbar sein. Der Browser muss beide erreichen.

- **OpenID Connect:** Der Server braucht ausgehend HTTPS zu Discovery-Dokument, JWKS und Token-Endpunkt des Anbieters (bei Entra `login.microsoftonline.com`). Ein ausgehender Proxy wird wie unter *Ausgehender Proxy* beschrieben genutzt.
- **Interne Zertifizierungsstelle:** Hat der Anbieter ein Zertifikat einer internen Zertifizierungsstelle, deren Zertifikat in `deploy/docker/ca/` ablegen (siehe *Interne Zertifizierungsstelle* bei der Installation). Sonst scheitert die Anmeldung mit `sso_idp`, das Protokoll meldet „certificate chain not trusted – add the issuing CA to deploy/docker/ca“ und **Verbindung testen** nennt denselben Grund.
- **SAML:** Mit eingefügtem Metadaten-XML braucht der Server **keinen** ausgehenden Zugriff – geeignet für abgeschottete Netze mit internem AD FS.
- Es gibt kein Single Logout: Abmelden im Panel beendet nur die Sitzung bei Focus Family, nicht beim Anbieter.

Sicherheit und Geheimnisse

- Client-Geheimnis und privater Schlüssel des SAML-Zertifikats liegen verschlüsselt mit dem Data-Protection-Schlüsselring (Volume `dpkeys`) in der Tabelle `SsoConfigurations` und werden nie angezeigt oder protokolliert. Geht der Schlüsselring verloren, das Geheimnis neu eingeben und ein neues SP-Zertifikat erzeugen.
- Jede Änderung der SSO-Einstellungen verlangt die erneute Anmeldebestätigung eines lokalen Kontos mit dem Recht **„Single Sign-On verwalten“** (`SsoManage`). Dieses Recht ist von „System-Konfiguration“ getrennt; Aktionen je Person (umstellen, Verknüpfung lösen, Erstanmeldung freigeben) gehören zur Benutzerverwaltung (`UserManage`).
- SAML-Antworten werden geprüft auf Signatur, Aussteller, Zielgruppe (Entity-ID), Ablauf, `Destination` / `Recipient`, `InResponseTo` (nur Antworten auf eigene, höchstens 10 Minuten alte Anfragen) und Wiederverwendung; die Antwort ist per HttpOnly-Cookie an den Browser gebunden, der die Anmeldung begonnen hat.

Fehlerbehebung bei SSO

Meldungen auf der Anmeldeseite (Code im Parameter `error=` der Adresse):

Code	Bedeutung	Abhilfe
`sso_only`	Das Konto meldet sich nur per SSO an, SSO ist erzwungen oder die Adresse gehört zu einer SSO-Domain.	Schaltfläche **Mit … anmelden** verwenden (in den Apps unter der Passwort-Anmeldung); Apps der Version 1.0: **Mit Passkey anmelden** öffnet dieselbe Seite. Notfallzugänge einer SSO-Domain: Passwort prüfen.
`sso_disabled`	SSO ist aus oder unvollständig.	SSO-Seite: Status prüfen, einschalten.
`sso_idp`	Anbieter nicht erreichbar (TLS-Zertifikat nicht vertrauenswürdig, DNS, Verbindung, Zeitüberschreitung) oder SAML-Metadaten nicht abrufbar bzw. ohne gültiges Signaturzertifikat.	**Verbindung testen** zeigt den Grund; bei interner Zertifizierungsstelle deren Zertifikat in `deploy/docker/ca/` ablegen und die App neu starten; sonst Netz/Proxy prüfen oder XML einfügen.
`sso_claims`	Anbieter liefert keine Kennung (`sub` /NameID) bzw. keine E-Mail-Adresse.	Claims bzw. Attribute beim Anbieter ausgeben, ggf. Claim-Zuordnung setzen.
`sso_denied`	Domäne oder Gruppe nicht erlaubt, oder Konto deaktiviert.	Erlaubte Domänen, Gruppen, Gruppen-Claim und Kontostatus prüfen.
`sso_unknown`	Kein Konto vorhanden und Just-in-time aus.	Konto als **Neues SSO-Konto** anlegen oder Just-in-time einschalten.
`sso_local`	Lokales Passwortkonto mit gleicher E-Mail-Adresse.	Konto unter **Anmeldeverfahren** auf SSO umstellen.
`sso_conflict`	Konto ist mit einer anderen Kennung verknüpft (Anbieter- oder Protokollwechsel, transiente NameID).	**Verknüpfung lösen**; bei SAML dauerhafte NameID einstellen.
`sso_email_unverified`	Adresse nicht als bestätigt gemeldet.	**Erstanmeldung freigeben**, Anbieter-Claims prüfen (Entra: `upn` , mandantenspezifische Authority) oder Vertrauensoption bewusst einschalten.
`sso_reauth`	Anmeldebestätigung nicht aktuell (`auth_time` / `AuthnInstant` fehlt oder ist zu alt).	Beim Anbieter vollständig neu anmelden; Uhrzeit von Server und Anbieter prüfen.
`sso_nameid`	Der SAML-Anbieter hat eine transiente NameID gesendet.	Beim Anbieter das NameID-Format `persistent` für Focus Family einstellen.
`sso`	Antwort ungültig (Signatur, Ablauf, Wiederholung, fremde oder abgelaufene Anfrage, falsches Client-Geheimnis).	Server-Protokoll (`docker compose logs app`) und Redirect-URI bzw. ACS-URL prüfen.

Meldungen beim Speichern: *gültige HTTPS-Adresse als Authority*, *Client-ID angeben*, *gültige HTTPS-Adresse für die Metadaten*, *Metadaten-URL oder Metadaten-XML angeben*, *Metadaten-XML ungültig* (kein HTTP-Redirect-Endpunkt oder kein Signaturzertifikat), *gültige Domänen angeben*, *SSO kann nicht erzwungen werden* (kein Notfallzugang), *Anmeldebestätigung fehlgeschlagen*.

Weitere häufige Ursachen: Redirect-URI bzw. ACS-URL beim Anbieter weicht ab (Schema `http` statt `https` hinter dem Proxy, anderer Hostname, fehlender Pfad); abgelaufenes Client-Geheimnis; Zertifikatswechsel beim Anbieter bei eingefügtem Metadaten-XML; Anmeldung über eine Kachel im Anbieterportal (IdP-initiiert, wird abgelehnt).

Audit-Ereignisse

Ereignis	Bedeutung
`sso.config_changed`	Einstellungen gespeichert (nur Feldnamen, nie Werte) oder Speichern ohne gültige Bestätigung abgelehnt
`sso.connection_tested`	„Verbindung testen“ mit Ergebnis
`sso.sp_certificate_rotated`	Neues SAML-Zertifikat erzeugt
`sso.login`	Erfolgreiche SSO-Anmeldung (Grund `oidc`, `saml`, ggf. `_reauth`)
`sso.login_denied`	Abgelehnte Anmeldung mit Grund, z. B. `sso:email_unverified`, `sso:domainnotallowed`, `sso:reauth_stale`
`sso.user_linked`	Erstverknüpfung (Grund `verified_email` oder `link_grant`)
`sso.user_provisioned`	Konto per Just-in-time angelegt
`sso.password_login_rejected`	Passwort-Anmeldung eines SSO-Kontos bzw. bei erzwungenem SSO abgewiesen (auch App-Anmeldung eines Notfallzugangs ohne SSO)
`sso.app_token_rejected`	Erneuerung bzw. Ausstellung von App-Tokens ohne SSO-Anmeldung bei erzwungenem SSO abgewiesen
`user.sso_enabled` / `user.sso_disabled`	Konto auf SSO bzw. lokal umgestellt
`user.sso_unlinked`	Verknüpfung gelöst
`user.sso_link_granted`	Erstanmeldung freigegeben

Checkliste SSO-Einführung

1. Notfallzugang prüfen (lokales Systemadministrationskonto mit Passwort und MFA).
2. Anwendung beim Anbieter registrieren, Adressen aus dem Panel übernehmen.
3. Panel-Felder ausfüllen, speichern, **Verbindung testen**.
4. Testkonto als **Neues SSO-Konto** anlegen, im privaten Browserfenster anmelden, Audit prüfen.
5. **Single Sign-On einschalten**, Konten vorab anlegen oder umstellen, Lizenzplätze zuweisen.
6. Erst danach, falls gewünscht, **SSO erzwingen**.
7. Ablaufdatum des Client-Geheimnisses bzw. der Anbieterzertifikate im Kalender vormerken.

Schnittstellen

Der Server bindet Fachverfahren (REST/JSON-API und Webhooks), Dokumentenmanagement und E-Akte (CMIS in beide Richtungen), XÖV-Empfänger (xdomea über Verzeichnis, SFTP, XTA 2 oder FIT-Connect), Identitätsanbieter (SCIM 2.0), Dienstkalender (iCalendar-Abos) und SIEM (Syslog) an. Einrichtung, Referenz, Datenschutz-Voreinstellungen, Betrieb und Fehlersuche beschreibt das **Schnittstellenhandbuch**. Im Panel steht es unter **Schnittstellen** → **Dokumentation der Schnittstellen**, zusammen mit der OpenAPI-Beschreibung der REST-API.

Das Wichtigste für den Betrieb:

- **Rechte:** Eingerichtet wird mit **Schnittstellen verwalten (REST-Clients, CMIS, XÖV, Webhooks, SCIM, SIEM)**; das Recht ist im System-Admin enthalten. Datenflüsse mit Falldaten werden erst aktiv, wenn eine

andere Person mit dem Recht **Datenschutzbeauftragte:r** sie mit einem lokalen Konto freigibt. Die Seite **Eingänge (Schnittstellen)** verlangt das fachliche Recht **Eingänge aus Schnittstellen prüfen und übernehmen**, Zuordnen und Übernehmen zusätzlich **Fälle bearbeiten**.

- **Adressen:** REST-API `/api/integration/v1`, Token-Endpunkt `/connect/token`, CMIS-Repository `/cmis/browser`, SCIM `/scim/v2`, Kalender-Abos `/feeds/calendar/<token>.ics`. Die mitgelieferten Reverse-Proxy-Vorlagen reichen alle Pfade weiter; wer Pfade einzeln freigibt, ergänzt diese. Die IP-Listen der API-Clients brauchen die Einstellungen aus *Reverse-Proxy einrichten*, sonst sieht der Server nur die Adresse des Proxys.
- **Ausgehende Verbindungen:** Die Zielsysteme der Schnittstellen kommen zur Tabelle unter *Ausgehende Verbindungen des Servers* hinzu. Interne Netze sind erlaubt; Loopback, Link-Local und Metadatendienste sind gesperrt. Zertifizierungsstellen aus `deploy/docker/ca/` gelten auch hier. Hinter einem ausgehenden Proxy gehören interne Zielsysteme in `NO_PROXY`.
- **XÖV über Verzeichnis:** Das Stammverzeichnis `/var/lib/fokusfamilie/xoev` ist im Standard-Stack kein Volume. Für die Übergabe an ein Gateway binden Sie ein Host-Verzeichnis ein (Besitzer UID 1654).
- **Geheimnisse** der Verbindungen liegen mit Data Protection verschlüsselt in der Datenbank. Das Volume `dpkeys` gehört deshalb zur Sicherung; ohne diese Schlüssel müssen alle Secrets und Client-Zertifikate der Schnittstellen neu eingetragen werden.
- **Überwachung:** Kennzahlen `fokusfamilie_integration_*` unter `/metrics`, Audit-Ereignisse `integration.*`, **Übertragungsprotokoll** im Panel. Not-Aus: **Alle ausgehenden Übertragungen pausieren (Not-Aus)**.
- **Lizenz:** Ohne gültige Organisationslizenz (weder aktives Abonnement mit gekauften Seats noch gesponserte Seats) ruhen die Zustellungen, und die Schnittstellen lehnen Anfragen ab; nach der nächsten erfolgreichen Lizenzprüfung geht es ohne Datenverlust weiter. Das Ergebnis der Prüfung wird fünf Minuten zwischengespeichert. Ist der Lizenzserver nicht erreichbar, gilt ein zuvor positives Ergebnis noch bis zu 24 Stunden; eine Ablehnung durch den Lizenzserver gilt sofort als „nicht lizenziert“.

Betrieb im Intranet oder über VPN

Das Admin-Panel kann ausschließlich im internen Netz oder über VPN erreichbar sein; dazu gibt der Reverse-Proxy den Hostnamen nur im internen DNS bekannt und nimmt Verbindungen nur aus dem internen Netz und dem VPN-Adressbereich an. Aus dem Internet braucht der Server keine eingehende Verbindung: Lizenzserver, Zahlungsanbieter und Versions-Feed rufen das Panel nie auf. Ausgehend muss der Server aber `fflicense.software-notion.com` **über HTTPS (Port 443)** erreichen.

Ausgehende Verbindungen des Servers

Ziel	Zweck	Pflicht?	Wenn gesperrt
`fflicense.software-notion.com:443`	Erst-Anmeldung der Installation, Seat-Prüfung beim Sync, Seats und Lizenzgeräte verwalten	**ja**	Ohne Erst-Anmeldung keine Lizenzprüfung der Geräte. Danach synchronisieren bereits bestätigte Nutzer weiter, bis das Panel neu startet; dann ruht der Sync, bis der Lizenzserver wieder erreichbar ist. Die Apps bleiben still offline, Daten gehen nicht verloren.
`api.software-notion.de:443`	Hinweis „Update verfügbar“	nein	kein Update-Hinweis
`static.software-notion.de:443`	Update aus dem Panel herunterladen	nein	Update manuell einspielen
`database.clamav.net:443`	Virensignaturen	empfohlen	Signaturen veralten; ohne erste Signaturen startet ClamAV und damit die Anwendung nicht. Signaturdateien dann einmalig in das Volume `clamdb` kopieren.
interner Zeitserver (NTP)	genaue Uhrzeit	**ja**	Anmeldung und Lizenzprüfung schlagen bei Uhrabweichung fehl
Identitätsanbieter (nur bei SSO)	OpenID Connect: Discovery, JWKS und Token-Endpunkt; SAML: Metadaten-URL	nur bei SSO	SSO-Anmeldung scheitert; bei SAML entfällt der Bedarf mit eingefügtem Metadaten-XML

Eingehend genügt eine Freigabe aus dem internen Netz und dem VPN-Adressbereich auf Port 443 des Reverse-Proxys. Handbuch-Links und die Seite „Seats verwalten“ öffnet der Browser der Administration; deren Arbeitsplatz braucht dafür Internetzugang, der Server nicht.

Ausgehender Proxy

Für einen Proxy tragen Sie in der `.env` ein:

```
HTTPS_PROXY=http://proxy.ihr-traeger.intern:3128
HTTP_PROXY=http://proxy.ihr-traeger.intern:3128
NO_PROXY=localhost,127.0.0.1,db,clamav
```

Anwendung und ClamAV verwenden den Proxy danach automatisch (`docker compose up -d --force-recreate app clamav`). `NO_PROXY` muss die internen Dienste `db` und `clamav` sowie `localhost` enthalten. Nehmen Sie `fflicense.software-notion.com` von einer TLS-Inspektion aus. Prüfen: `docker compose exec app curl -sS -o /dev/null -w "%{http_code}\n" https://fflicense.software-notion.com/health/live` liefert `200`.

Geräte, Lizenz und Sync

- Die Apps synchronisieren, sobald sie das Panel im internen Netz oder über VPN erreichen. Außerhalb arbeiten sie vollständig offline weiter, ohne Fehlermeldungen und ohne Sperre; Änderungen werden beim nächsten Kontakt übertragen.

- Die Lizenz prüft jedes Gerät **selbst direkt beim Lizenzserver** über seine eigene Internetverbindung. Das Panel stellt dafür nur eine kurzlebige Bestätigung aus. Für eine erfolgreiche Prüfung muss das Gerät also gleichzeitig das Panel und `fflicense.software-notion.com` erreichen, zum Beispiel im internen Netz mit Internetzugang oder per Split-Tunnel-VPN.
- Gelingt die Prüfung nicht, gilt die Kulanz: Tag 0–3 still, Tag 3–5 ein Hinweis pro Tag, Tag 5–10 stündlich, ab Tag 10 Sperre bis zur nächsten erfolgreichen Prüfung. Geräte sollten deshalb mindestens alle paar Tage verbunden sein.
- Die Richtlinie „Max. Offline-Dauer (Tage)“ sollte zu dieser Nutzung passen. Läuft sie ab, erscheint in der App nur ein Hinweis zur erneuten Anmeldung.

DNS und Zertifikate

- Verwenden Sie für LAN und VPN **denselben Hostnamen** (Split-Horizon-DNS bzw. der interne DNS-Server auch im VPN). Anmeldungen sind an die Adresse gebunden; ein Wechsel des Namens erzwingt eine neue Anmeldung.
- Öffnen Sie das Panel zum Erzeugen von QR-Codes unter genau diesem Namen; der QR-Code übernimmt die aktuelle Adresse.
- Die mobilen Apps verbinden sich nur per HTTPS, und der Server stellt Anmeldungen nur über HTTPS aus. Am einfachsten ist ein öffentlich vertrauenswürdiges Zertifikat für einen internen Namen unterhalb Ihrer Domain, ausgestellt über die Let's-Encrypt-DNS-Prüfung (DNS-01).
- Ein Zertifikat Ihrer internen Zertifizierungsstelle funktioniert auf iOS/iPadOS (Root-Zertifikat per MDM-Profil verteilen oder unter Einstellungen → Allgemein → Info → Zertifikatsvertrauenseinstellungen aktivieren) und in den Desktop-Apps (Zertifikatsspeicher des Betriebssystems). **Die Android-App vertraut nur den System-Zertifizierungsstellen**; Nutzer- und per MDM installierte Zertifikate akzeptiert sie nicht. Für Android-Geräte ist daher ein öffentlich vertrauenswürdiges Zertifikat nötig.

VPN

- **Split-Tunnel:** internes Netz durch den Tunnel, Internet direkt. Einfachste Variante.
- **Full-Tunnel:** möglich, wenn das interne Netz einen Internetausgang zu `fflicense.software-notion.com` bietet.
- **Dauerhaft aktives VPN** (iOS per MDM, Android „Durchgehend aktives VPN“, Windows Always On VPN) hält Sync und Lizenz ohne Zutun aktuell.
- **Per-App-VPN:** Focus Family dem VPN zuordnen; für die Anmeldung über den System-Browser bei Bedarf auch den Browser bzw. die Domain des Panels.

Ausführliche Tabellen und Prüfschritte stehen im Betriebs-Runbook (`docs/deploy/RUNBOOK.de.md` , Abschnitt 4).

Sicherheit

Transport und Absicherung

- In Produktion ausschließlich **HTTPS**; TLS terminiert am Proxy.
- **Forwarded-Header** werden nur aus `REVERSE_PROXY_NETWORK` und von `REVERSE_PROXY_IP` akzeptiert, mit `REVERSE_PROXY_ALLOWED_HOST` zusätzlich nur für den eingetragenen Hostnamen. Damit lässt sich das nach Client-IP partitionierte **Rate Limiting** nicht über gefälschte Header umgehen.

- **Security-Header, CSP, HSTS** und **Antiforgery** im Cookie-Bereich des Panels sind aktiv; die Security-Header gelten auch für Fehlerseiten, und jede Panel-Aktion (`/admin/act/*`) verlangt ein Antiforgery-Token.
- **Rate Limiting** pro IP mit festen Zeitfenstern je Minute: allgemein `RateLimiting:GlobalPermitLimit` (3000), Anmeldung, Konto-, MFA-, Passkey- und SSO-Endpunkte `AuthPermitLimit` (100), Geräteregistrierung `DeviceEnrollPermitLimit` (10), Absturzberichte `DiagnosticsPermitLimit` (20), Uploads in das Nachweis-Archiv `ArchiveUploadPermitLimit` (60). Wiederholte falsche Passwörter sperren das Konto vorübergehend. Übersetzt ein VPN-Gateway alle Geräte auf eine Adresse (NAT), teilen sie sich ein Kontingent.
- **Kontosperre:** Ein deaktiviertes Konto verliert den Zugriff sofort – Panel-Sitzungen, App-Tokens und Sync werden abgewiesen, ebenso Tokens mit veraltetem Security-Stamp (etwa nach Passwortwechsel). Ohne Geräte-ID ist kein Sync möglich.
- **Anmeldung ohne Kontoverrat:** Unbekannte, gesperrte und deaktivierte Konten sowie falsche Passwörter erhalten dieselbe Antwort nach derselben Hash-Prüfung.
- Signier- und Verschlüsselungszertifikate sind **Pflicht** (Fail-fast beim Start, wenn sie fehlen).
- In Produktion sind sensible EF-Protokollierung und die Entwickler-Fehlerseite abgeschaltet; Diagnosen geben nie Verbindungszeichenfolgen preis, Protokolle nie Tokens, Secrets oder Fallinhalte.

Autorisierung

Jeder Lese- und Schreibzugriff wird **serverseitig** gegen den aktuellen Zustand autorisiert, nie gegen Behauptungen des Clients. Die Sichtbarkeit wird bereits in SQL gefiltert. Als **privat** markierte Einträge sind für andere App-Nutzer unsichtbar – auch für Teamleitungen und alle Admin-Rollen, und zwar bei Abfrage, Suche, Export und direktem Zugriff über eine ID.

Fallübertragung aus den Apps: `POST /api/v1/cases/{caseId}/transfer` mit `{ "userId": …, "keepAccess": true|false }` übergibt einen Fall an eine andere Fachkraft. Zulässig ist das nur für die aktuelle Eigentümerin bzw. den aktuellen Eigentümer mit Schreibrecht am Fall; das Ziel muss ein aktives Fachkraft-Konto derselben Organisation sein. Antworten: `204` bei Erfolg, `400` mit `{ "code": "transfer.invalid_target" }` bei ungültigem Ziel, `400` mit `{ "code": "transfer.private_keep_access" }`, wenn ein privater Fall mit `keepAccess: true` übertragen werden soll (private Fälle sind nur ohne Weiterzugriff übertragbar; der Fall bleibt unverändert; `GET /api/v1/cases/{caseId}/participants` liefert dafür je Eintrag `caseIsPrivate`, die Apps deaktivieren „Weiterhin Zugriff behalten“ dann vorab), `403` für Nicht-Eigentümer, `404`, wenn der Fall für den Aufrufer nicht sichtbar ist. Jede Übertragung wird im Audit als „Fall übertragen“ protokolliert. Die Apps rufen den Endpunkt nur online auf und erst nach einer vollständigen Vorab-Synchronisierung; schlägt diese fehl oder bleiben ausstehende Änderungen, offene Konflikte oder nicht hochgeladene Dateien, wird die Übertragung ohne Serveraufruf abgebrochen.

Rechte-Obergrenze: Jede Kontoänderung im Panel und in der Verwaltungs-API (Rechte, Passwort, MFA, Passkeys, Deaktivieren, Anmeldeverfahren, Bereiche) setzt voraus, dass das Zielkonto keine Rechte außerhalb des eigenen Umfangs hält. Bereichsgebundene Admins ändern nur Konten, deren Bereiche und Teams vollständig in ihrem Umfang liegen, und vergeben weder *Fach-Admin* noch *Bereiche verwalten* (`AreaManage`). Das letzte aktive lokale Systemadministrationskonto mit Passwort ist gegen Deaktivieren und Umstellen auf SSO geschützt.

Bereichsmodus: Ist der Bereichsmodus eingeschaltet (Panel → **Bereiche**), begrenzt der `AreaScopeEvaluator` die Sicht bereichsgebundener Fach-Admins zusätzlich zu ihren Rechten auf die Fachkräfte ihrer Bereiche und deren Fälle; die Filter laufen in SQL. Alle Aktionen unter `/admin/act/*` passieren einen Endpoint-Filter, der das Ziel prüft und bei fehlender Deklaration fail-closed ablehnt (Redirect `err=not_found` , Audit

`area.scope_denied`). Ein- und Ausschalten verlangt eine erneute Anmeldebestätigung (Passwort plus TOTP bzw. Passkey). Apps, Sync-Protokoll, `MeResponse` und Richtlinien bleiben unverändert; ist der Modus aus, ändert der Evaluator keine Abfrage.

Gerätebindung

Geräte registrieren sich über einen Challenge- und Besitznachweis-Ablauf. Jede API-Anfrage eines gebundenen Geräts trägt einen signierten **Device-Proof**, dessen Nonce persistiert wird (Replay-Schutz); Access-Tokens sind an das Gerät gebunden. Für Neuinstallationen stellt die Administration **Re-Enrollment-Codes** aus.

Anmeldung

- **OpenIddict** mit Authorization Code + PKCE, kurzlebige und rotierte Tokens.
- **MFA** per TOTP als globale Richtlinie, mit Challenge beim Login – für Apps und Panel.
- **Passkeys** (WebAuthn) als passwortlose Alternative; ist ein Passkey aktiv, ist die Passwortanmeldung für dieses Konto abgeschaltet.
- **Technische Admin-Konten** können sich grundsätzlich **nicht in den Apps** anmelden.
- **Break-Glass** ist eine zeitlich begrenzte (höchstens 120 Minuten), MFA-pflichtige und auditierte Ausnahmesitzung – und nur dann möglich, wenn die Organisationsrichtlinie es erlaubt.

Uploads

Uploads sind größen- und typbeschränkt und werden über ClamAV geprüft. In Produktion gilt **fail-closed**: Ist der Scanner nicht konfiguriert, nicht erreichbar oder läuft er in einen Timeout, werden Uploads abgelehnt. Blob-Inhalte sind ruhend verschlüsselt.

Offline-Grant-Keyset rotieren

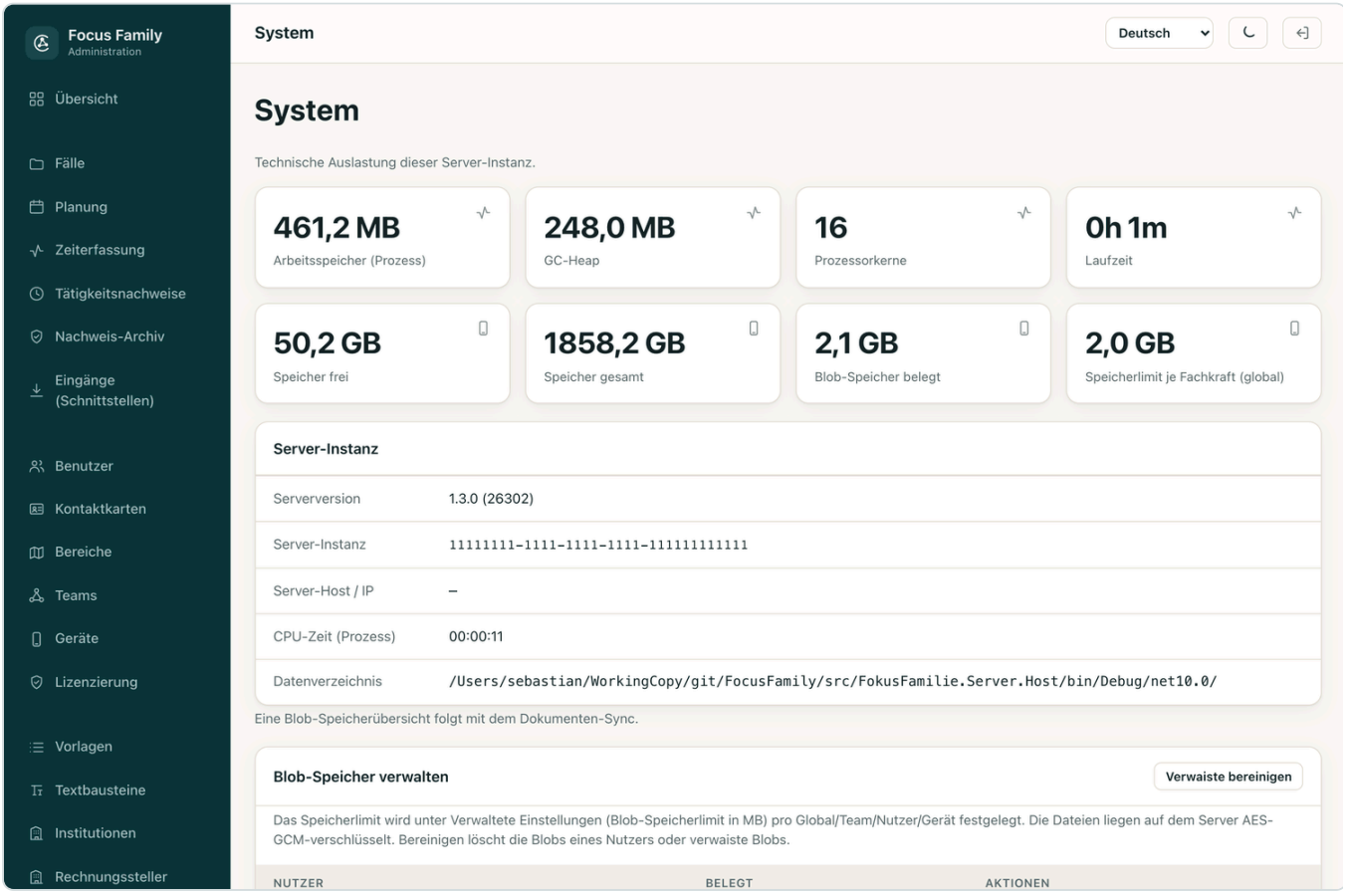
Clients vertrauen Offline-Grant-Schlüsseln nur aus einem **signierten Keyset**, das an die `ServerInstanceId` gebunden ist und eine monotone Version trägt. Die Rotation läuft in **zwei Phasen**:

1. Alten privaten Schlüssel als Signierer behalten, den **neuen öffentlichen** Schlüssel in `OFFLINE_GRANT_ADDITIONAL_PUBLIC_KEY` eintragen und `OFFLINE_GRANT_KEYSET_VERSION` erhöhen. Warten, bis alle Clients mindestens einmal synchronisiert haben.
2. `OFFLINE_GRANT_KEY` auf den neuen privaten Schlüssel umstellen, den **alten öffentlichen** Schlüssel übergangsweise behalten und die Version erneut erhöhen. Den alten Schlüssel entfernen (und die Version wieder erhöhen), sobald die längste Offline-Grant-Laufzeit verstrichen ist.

> Wird die Version beim Ändern der Schlüsselliste **nicht** erhöht, behalten die Clients ihr bisheriges Keyset (fail-closed). Die Version niemals verringern.

Technische Bereiche des Admin-Panels

System-Status



Serverversion, Instanz, Speicherbelegung und die Verwaltung des Blob-Speichers.

Der Menüpunkt **System** (Recht *System-Status ansehen*) zeigt Serverversion, API-Version, Organisation, Server-Instanz-ID, Change-Sequenz und die Auslastung. Hier liegt auch die Verwaltung des **Blob-Speichers**.

> ⚠ **Bereinigen löscht unwiderruflich.** Die Funktion entfernt hochgeladene Dokumente und Medien eines Nutzers oder verwaiste Dateien **endgültig** vom Server. Inhalte, die auf keinem Gerät mehr liegen, sind damit verloren. Vor dem Löschen erscheint eine Sicherheitsabfrage.

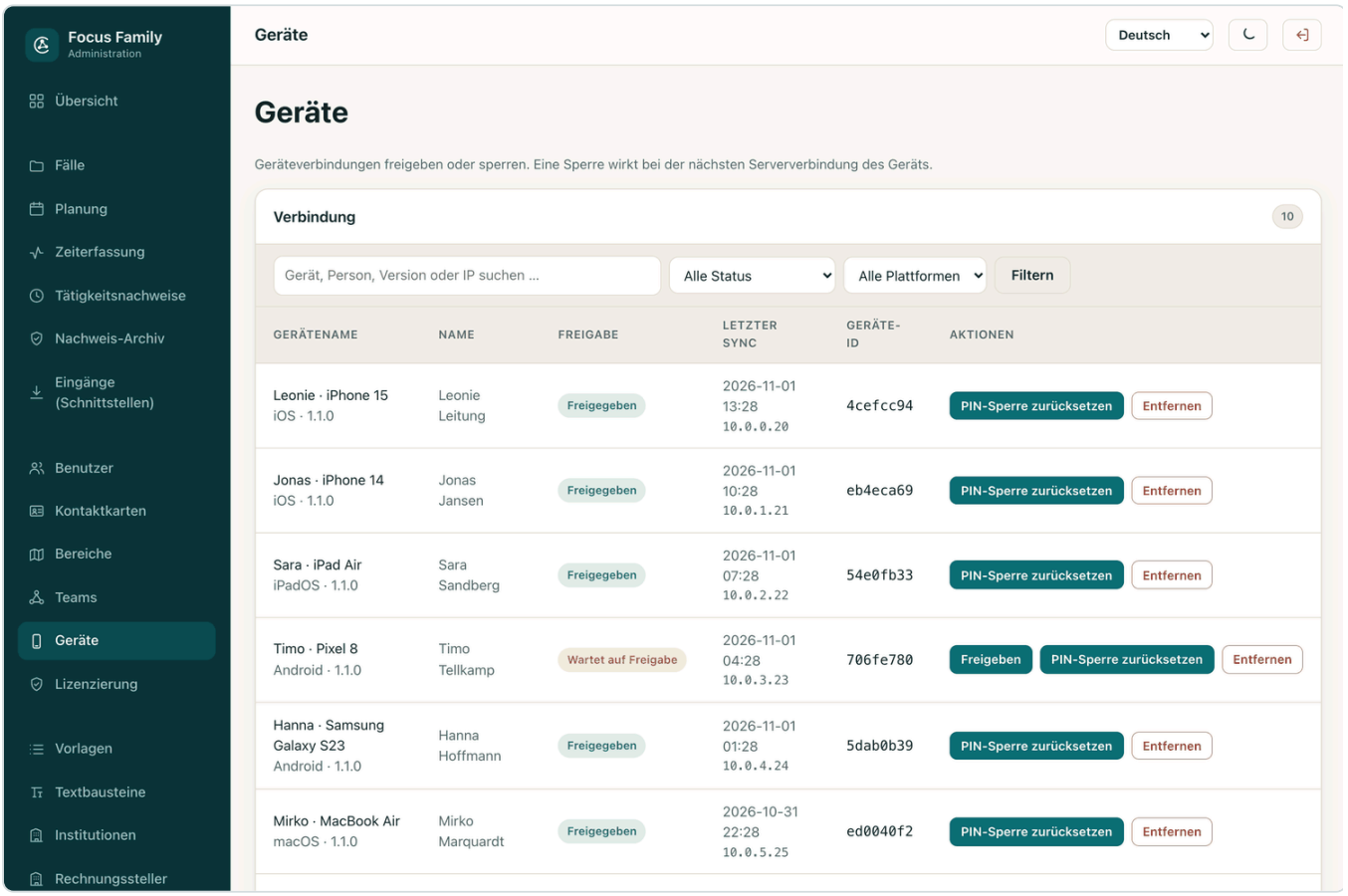
Auf derselben Seite steht die Karte **Schlüssel für Dateianhänge** (Recht *System-Konfiguration*). Sie nennt die aktive Schlüsselkennung und die Zahl der Hüllen, die noch auf einem alten Schlüssel liegen. Nehmen Sie einen neuen Schlüssel **zuerst** in `BlobStorage:EncryptionKeys` auf, zeigen Sie `BlobStorage:ActiveKeyId` darauf, starten Sie neu und lassen Sie dann die **Neuverschlüsselung** laufen. Den alten Schlüssel entfernen Sie erst, wenn die Karte nichts mehr offen meldet.

Der Lauf ersetzt eine Datei nur, solange sie unverändert ist; gleichzeitig bearbeitete Dateien behalten ihre neuere Fassung und erscheinen als übersprungen. Eine Hülle, die sich nicht lesen lässt, wird gesondert ausgewiesen und verhindert die Meldung „vollständig" — das ist Absicht, denn der zugehörige Schlüssel könnte noch gebraucht werden. In einer Installation mit mehreren Instanzen arbeitet immer nur ein Lauf; ein zweiter Aufruf meldet das, statt parallel zu schreiben. Beginn und Abschluss stehen im Audit.

Darunter liegt die Karte **Monitoring** (Recht *Monitoring konfigurieren*). Sie schaltet die Überwachungsschnittstelle ein, erzeugt und erneuert den Zugriffsschlüssel, begrenzt ihn auf bestimmte

Adressen und legt fest, ob Betriebszahlen mitgeliefert werden. Die Karte nennt außerdem die vollständigen Adressen zum Kopieren. Wie Sie das an Prometheus, Grafana oder Uptime Kuma anschließen, steht im Abschnitt „Betrieb und Überwachung".

Geräte

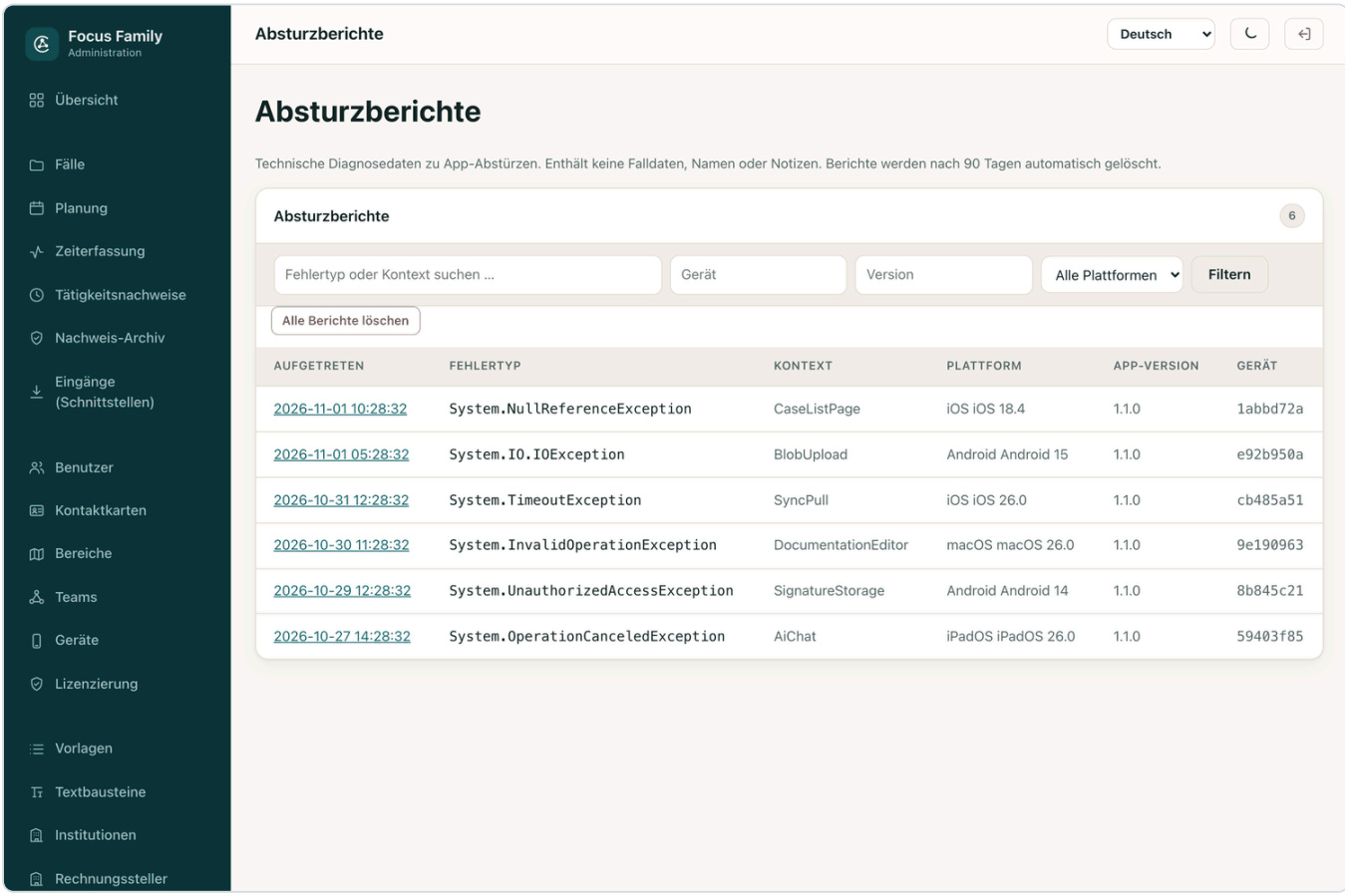


Freigeben, sperren, PIN-Sperre zurücksetzen, Gerät entfernen und Re-Enrollment-Codes ausstellen.

Sperren wirken, sobald das Gerät das nächste Mal Kontakt aufnimmt. **Entfernen** löst die Bindung, widerruft die Geräteregistrierung der Lizenz und gibt den Platz frei.

Die Apps melden bei jeder Anfrage das Gerätemodell (Kopfzeile `X-Device-Model`, etwa „iPhone Duo“, höchstens 64 Zeichen, Spalte `Devices.DeviceModel`) und einen selbst vergebenen Gerätenamen (`X-Device-Name`, höchstens 100 Zeichen, nur wenn er mehr sagt als die Gerätegattung); beide sind URL-kodiert, Steuerzeichen werden entfernt. Die App-Version wird bei jeder Anfrage aus `X-Client-Version` übernommen, sofern sie eine gültige Versionsnummer ist, und als `Major.Minor[.Build]` gespeichert. *Geräte* und die Geräteliste unter *Lizenzierung* zeigen das Modell, darunter den Namen.

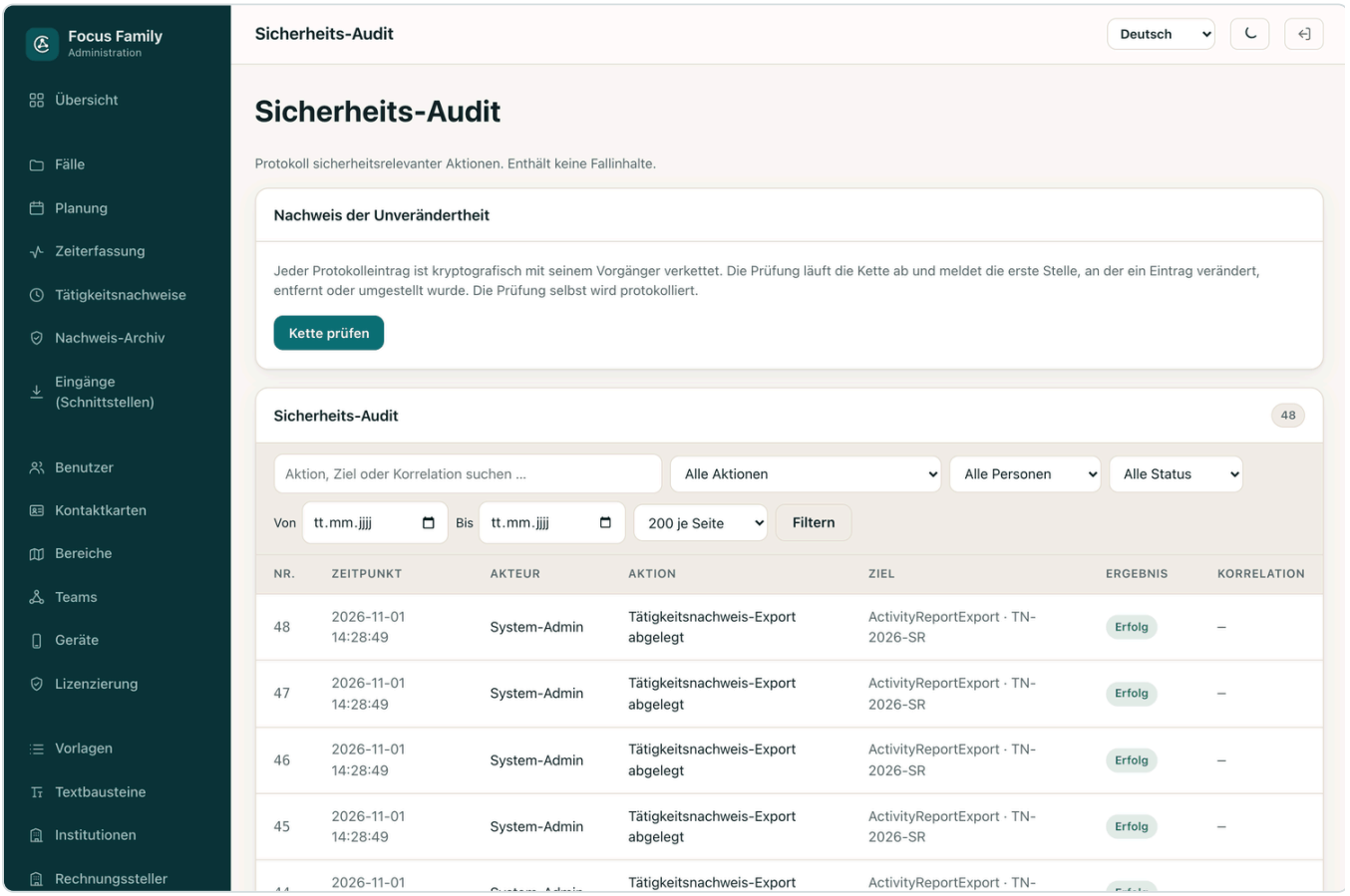
Absturzberichte



Technische Fehlerberichte der Apps – ohne Falldaten, Namen oder Notizen.

Apps können nach einem Absturz einen technischen Bericht senden (Recht *Absturzberichte ansehen*). Die Berichte enthalten Zeitpunkt, Plattform und technische Details – **keine** Inhalte. Der Server löscht Berichte nach `Diagnostics:CrashReportRetentionDays` (Standard 90 Tage) automatisch; im Panel lassen sich mit dem Recht *Absturzberichte verwalten* einzelne Berichte oder alle Berichte der Organisation sofort löschen (Audit-Einträge `diagnostics.crash_report_deleted` und `diagnostics.crash_reports_purged`). Die Annahme ist je Absender-IP auf `RateLimiting:DiagnosticsPermitLimit` Berichte pro Minute begrenzt (Standard 20), zusätzlich auf 50 Berichte je Gerät und Tag; übergroße Berichte werden abgewiesen.

Sicherheits-Audit



Wer hat wann was getan – mit Ergebnis (Erfolg, Abgelehnt, Fehler).

Das Audit ist Ihre erste Anlaufstelle bei Verdachtsfällen: Anmeldungen, Rechteänderungen, Gerätefreigaben, Richtlinienänderungen, Notfallzugriffe, Speicherbereinigungen und Leitungskorrekturen an Zeiteinträgen. Fallinhalte enthält es nicht.

Fallverlauf (Tabelle `CaseTimelineEntries`, Schema-Version 23): Getrennt vom Audit führt der Server je Fall einen Verlauf der Änderungen. Er wird in derselben Datenbanktransaktion geschrieben wie die Änderung selbst – beim Sync-Push (`SyncService`, mit Gerät), bei Panel-Aktionen, den App-Endpunkten für Teilen, Vertretung und Übertragung, der Verwaltungs-API (`assign-team`, `assign-user`, `acl`) und beim zeitgesteuerten Ablauf von Vertretungen (Quelle „System“). Gespeichert werden Zeitpunkt, Konto, Gerät, Quelle, Entitätstyp und -ID, Aktion und die Namen geänderter Felder aus dem Vergleich alter und neuer Nutzdaten – nie Werte und nie Namen. Technische Felder (Zeitstempel, Versionen, Prüfsummen, abgeleitete Felder) zählen nicht; Änderungen ohne fachliches Feld erzeugen keinen Eintrag; fallfreie Datensätze werden nicht erfasst. Die Anwendungsrolle hat auf der Tabelle kein `UPDATE`-Recht. Einträge werden beim endgültigen Löschen eines Falls nach der Löschprüfung mitgelöscht; beim endgültigen Löschen einzelner Datensätze bleibt je Datensatz nur der Vermerk „endgültig gelöscht“. Eine eigene Aufbewahrungsfrist gibt es nicht – der Verlauf lebt so lange wie der Fall, wie das Audit auch. Personennamen werden erst bei der Anzeige aus dem Benutzerkonto aufgelöst.

Nachweis-Archiv

Exportierte Tätigkeitsnachweise legen App, Desktop und Panel verschlüsselt im Blob-Speicher ab (Tabelle `ActivityReportExports`). Jede Ablage zählt gegen das **Blob-Speicherlimit** der Fachkraft; die zulässige Dateigröße folgt der Upload-Richtlinie, Uploads sind über `RateLimiting:ArchiveUploadPermitLimit`

(Standard 60 je Minute und IP) begrenzt. Wird ein enthaltener Zeiteintrag oder Fall endgültig gelöscht, entfernt der Server die Datei; Nummer, Version und Prüfwerte bleiben als Nachweis stehen.

Lizenzierung

Der Menüpunkt **Lizenzierung** zeigt den Zustand der Organisationslizenz (ohne Abonnement, aber mit gesponserten Seats: „Gesponsert“), die belegten Plätze als „zugewiesen / gekauft + gesponsert“, den Hinweis zu gesponserten Seats, nicht lizenzierte Zuweisungen („Zugewiesen, ohne Lizenz“) sowie die registrierten Geräte mit Plattform und App-Version. Ist die Lizenzintegration nicht konfiguriert, weist die Seite darauf hin, dass `Licensing:InstallationId` und `Licensing:PrivateKeyPem` fehlen (Schlüsselpaar mit `--licensing-keygen` erzeugen).

Betrieb und Überwachung

Health-Endpunkte

```
curl http://127.0.0.1:8080/health/live     # Prozess lebt
curl http://127.0.0.1:8080/health/ready    # Prozess + Datenbank bereit
```

Binden Sie `/health/ready` in Ihre Überwachung ein. Der Container bringt außerdem einen eigenen `HEALTHCHECK` mit.

Protokolle

```
docker compose logs -f app
docker compose logs migrator
docker compose logs clamav
```

Protokolle enthalten **niemals** Tokens, Secrets oder Fallinhalte.

Zeitsynchronisation prüfen

Der Server führt alle Zeitstempel in UTC und ist für die Zeit maßgeblich. Eine falsch gehende Uhr des Wirts wirkt sich unmittelbar aus:

Mechanismus	Zeitbezug	Folge bei Abweichung
Gerätenachweis jeder Sync- und Dateianfrage	±5 Minuten	alle Geräte werden abgewiesen
Offline-Grants	Ausstellungszeit höchstens 5 Minuten in der Zukunft	Geräte verwerfen den Grant
Token, Einrichtungs-QR, Passkey-Anmeldung	5 Minuten bis 1 Stunde	Anmeldung scheitert oder Token gelten zu lange
Zwei-Faktor-Codes (TOTP)	30-Sekunden-Schritte	Codes werden abgelehnt
SSO (OIDC und SAML)	Gültigkeitsfenster der Antworten, erneute Anmeldebestätigung höchstens 5 Minuten alt	Fehler `sso` bzw. `sso_reauth`
Lizenzserver	2 Minuten Toleranz	Lizenzprüfung und Seat-Vergabe scheitern
Audit-Protokoll, Aufbewahrungs- und Fristläufe	Zeitstempel und Fristen in Tagen	Ereignisse nicht korrelierbar, Fristen falsch

Prüfung bei der Inbetriebnahme und in der regelmäßigen Betriebskontrolle:

```
timedatectl status                # "System clock synchronized: yes" und "NTP service: active"
chronyc tracking                  # bei chrony: "Leap status : Normal", Abweichung im
Millisekundenbereich
chronyc sources -v                # erreichbare Zeitquellen
docker compose exec app date -u   # dieselbe UTC-Zeit wie auf dem Wirt
```

Ist die Synchronisation aus, hilft `sudo timedatectl set-ntp true` (systemd-timesyncd) oder die Installation von chrony mit dem internen Zeitserver als Quelle. Die Uhr des Wirts gehört in dessen Überwachung; das Admin-Panel zeigt keine Uhrabweichung an. Auf verwalteten Endgeräten empfiehlt sich die automatische Zeiteinstellung per MDM.

Monitoring

Der Server bringt eine eigene Überwachungsschnittstelle mit. Sie sendet nichts nach außen — Ihre Werkzeuge holen die Zahlen ab.

Adresse	Zweck	Zugang
`/health/live`	Prozess lebt	offen
`/health/ready`	Prozess und Datenbank bereit	offen
`/status`	liefert `up` als Klartext	offen
`/metrics`	Kennzahlen im Prometheus-Format	Zugriffsschlüssel
`/monitoring/status`	dieselben Zahlen als JSON, mit `status` = `ok` / `degraded`	Zugriffsschlüssel

Einrichten (Recht *Monitoring konfigurieren*, in der Rechtevergabe als zweites Kreuz in der Zeile *System-Status*): **System** → **Monitoring** → **Schlüssel erzeugen**. Der Schlüssel wird **einmal** angezeigt; danach liegt nur noch seine Prüfsumme in der Datenbank. Anschließend **Schnittstelle einschalten**. Ohne Schlüssel antworten `/metrics` und `/monitoring/status` mit `404` — von außen ist nicht erkennbar, dass es sie gibt. Dasselbe gilt,

solange die Schnittstelle abgeschaltet ist: Erzeugen oder Erneuern des Schlüssels schaltet sie **nicht** ein. Mit eingeschalteter Schnittstelle und falschem oder fehlendem Schlüssel lautet die Antwort `401`.

Optional tragen Sie unter **Zugelassene Adressen** die IP-Adressen oder CIDR-Bereiche Ihres Überwachungsservers ein. Hinter einem Reverse-Proxy zählt die Adresse, die der Proxy weiterreicht; dafür muss `ReverseProxy:Enabled` gesetzt und Ihr Proxy unter `KnownProxies` eingetragen sein. Lässt sich die Adresse nicht bestimmen, wird der Abruf abgewiesen.

Prometheus:

```
scrape_configs:
  - job_name: fokusfamilie
    scrape_interval: 60s
    metrics_path: /metrics
    scheme: https
    authorization:
      type: Bearer
      credentials: ffmon_…
    static_configs:
      - targets: ["server.traeger.example"]
```

Uptime Kuma: Monitortyp *HTTP(s) - Keyword* auf `https://…/monitoring/status`, Keyword `"status":"ok"`, unter *Headers* `{"Authorization": "Bearer ffmon_…"}`. Wer nur Erreichbarkeit prüfen will, nimmt `/health/ready` ohne Schlüssel.

Worauf es sich zu alarmieren lohnt:

Kennzahl	Bedeutung
`fokusfamilie_ready`	0 = Datenbank nicht erreichbar **oder** eine Kennzahlengruppe konnte nicht ermittelt werden
`fokusfamilie_collection_complete`	0 = mindestens eine Gruppe fehlt; welche, steht in `failedGroups` der JSON-Antwort
`fokusfamilie_disk_used_ratio`	Platte des Datenträgers; ab 0,85 wird es eng
`fokusfamilie_blob_used_ratio`	Anteil des belegten Anhangs-Kontingents
`fokusfamilie_privacy_requests_overdue`	Auskunftsersuchen über der gesetzlichen Frist — sollte immer 0 sein
`fokusfamilie_audit_failures_24h`	abgewiesene sicherheitsrelevante Aktionen
`fokusfamilie_break_glass_active`	offene Notfallzugriffe
`fokusfamilie_retention_last_run_age_seconds`	Alter des letzten Laufs der Aufbewahrungsfristen (Änderungsprotokoll und Löschmarkierungen, täglich); deutlich über 90 000 heißt, dass der Lauf ausbleibt
`fokusfamilie_blob_key_rotation_unreadable`	Anhänge, deren Schlüssel fehlt
`fokusfamilie_blob_scan_age_seconds`	Alter des letzten Anhangs-Durchlaufs; dauerhaft hohe Werte heißen, dass der Durchlauf nicht mehr durchkommt
`fokusfamilie_audit_chain_checkpoint_age_seconds`	Alter der letzten Kettenprüfung; −1 = nie geprüft
`fokusfamilie_devices_pending_approval`	Geräte, die auf Freigabe warten

Wie oft darf abgerufen werden: So oft Sie wollen. Der Anhangs-Durchlauf über den Datenträger ist der einzige teure Teil; sein Ergebnis wird **fünf Minuten** wiederverwendet, überlappende Abrufe warten nicht, sondern bekommen das vorhandene Ergebnis. Ein *gescheiterter* Durchlauf wird nur 30 Sekunden gehalten — eine vorübergehende Störung verschwindet damit zügig wieder aus dem Bericht, ein dauerhaft defekter Speicher wird trotzdem nicht bei jedem Abruf durchsucht. Die Einstellungen der Schnittstelle selbst liegen **15 Sekunden** im Zwischenspeicher, damit ein Abruf auch dann noch beantwortet wird, wenn die Datenbank gerade nicht erreichbar ist — in diesem Fall antwortet `/monitoring/status` mit `degraded` und `503` statt gar nicht. Ein Schlüsselwechsel oder das Abschalten der Schnittstelle wirkt sofort, unabhängig vom Zwischenspeicher; eine ältere Fassung kann nicht zurückkommen, weder durch einen gleichzeitig laufenden anderen Speichervorgang noch durch einen gleichzeitig laufenden Abruf, der die Einstellungen kurz zuvor gelesen hatte.

Bei längerem Datenbankausfall endet dieser Rückfall nach **zehn Minuten**: Danach antwortet die Schnittstelle wieder mit `404`, statt weiter mit Angaben zu arbeiten, deren Gültigkeit sie nicht mehr prüfen kann. Ein entzogener Schlüssel gilt also höchstens zehn Minuten über einen Datenbankausfall hinaus — und auf der Instanz, an der Sie ihn entzogen haben, keine Sekunde.

Wenn eine Kennzahl fehlt: Scheitert eine einzelne Gruppe (etwa der Anhangs-Durchlauf oder die Kettenprüfung), liefert die Schnittstelle trotzdem alle übrigen Zahlen, nennt die ausgefallene Gruppe in `failedGroups`, setzt `fokusfamilie_collection_complete` auf 0 und meldet `degraded`. Ein Teilausfall wird also nie als „alles in Ordnung" ausgegeben.

Die Schnittstelle liefert **ausschließlich Zahlen und technische Kennwerte** — keine Namen, keine Fallinhalte, keine personenbezogenen Angaben. Mit **Betriebszahlen mitliefern** schalten Sie zusätzlich die Anzahl von Konten, Teams, Fällen und Geräten ab, wenn schon die Größenordnung nicht nach außen soll.

Ein neuer Schlüssel macht den bisherigen sofort ungültig. Ein- und Ausschalten, Schlüsselwechsel und Änderungen am Umfang stehen im Sicherheits-Audit.

Lizenzserver: Dieselben zwei Adressen (`/metrics`, `/monitoring/status`) gibt es auf dem Lizenzserver. Dort gibt es keine Oberfläche; der Schlüssel kommt aus der Konfiguration (`Monitoring__ScrapeToken`, optional `Monitoring__AllowedNetworks`). Ohne gesetzten Schlüssel sind auch dort beide Adressen unsichtbar. Alarmieren Sie dort auf:

Kennzahl	Bedeutung
`fflicense_database_up`	0 = Lizenzdatenbank nicht erreichbar
`fflicense_signing_key_usable`	0 = der aktive Signaturschlüssel fehlt oder ist abgeschaltet; es können keine Lizenzen mehr ausgestellt werden
`fflicense_signing_key_active`	Anzahl aktiver Schlüssel; **mehrere sind während eines Schlüsselwechsels normal** und kein Fehler
`fflicense_collection_complete`	0 = eine Kennzahlengruppe fehlt (siehe `failedGroups`)
`fflicense_identity_deletions_manual_required`	> 0 = ein Löschauftrag einer Lizenzidentität wartet auf manuelle Bearbeitung (Status `degraded`)
`fflicense_identity_deletions_overdue`	> 0 = ein Löschauftrag ist länger als 24 Stunden offen (Status `degraded`)

Sync-Aufbewahrung und vollständiger Abgleich

Einmal täglich entfernt der Server Einträge des Änderungsprotokolls, die kein Gerät mehr braucht (Richtlinienwerte `changeFeedRetentionDays` und `tombstoneRetentionDays` der Governance-API, Standard je 180 Tage; Audit-Eintrag `retention.enforced`, Kennzahl `fokusfamilie_retention_last_run_age_seconds`). Der neueste Stand jedes vorhandenen Datensatzes bleibt immer erhalten, neue Geräte können sich also jederzeit einrichten.

Ein Gerät, das länger als die Aufbewahrungsfrist nicht synchronisiert hat, kann nicht mehr schrittweise aufholen. App und Desktop-App bauen ihren Datenbestand dann automatisch aus dem aktuellen Serverstand neu auf: Zuerst werden ausstehende lokale Änderungen gesendet, danach wird der vollständige Stand geladen, und Datensätze, die inzwischen auf dem Server gelöscht wurden oder nicht mehr zugänglich sind, werden vom Gerät entfernt (einschließlich Anhängen). Nur auf dem Gerät vorhandene Datensätze, private Datensätze und Datensätze mit noch nicht gesendeten Änderungen bleiben unberührt; ein geänderter Datensatz, den es auf dem Server nicht mehr gibt, erscheint als Konflikt „Zentral gelöscht“ (lokal behalten oder verwerfen). Eine Fehlermeldung erscheint nicht; wird der Neuaufbau unterbrochen, setzt er bei der nächsten Synchronisierung einfach erneut an.

Ältere App-Versionen können diesen Neuaufbau nicht ausführen. Der Server liefert ihnen stattdessen den vollständigen aktuellen Stand, sodass sie weiter synchronisieren. Datensätze, die während der langen Offline-Zeit auf dem Server gelöscht oder entzogen wurden, können auf einem solchen Gerät jedoch bestehen bleiben, bis die App auf die aktuelle Version aktualisiert oder neu installiert wird. Im Protokoll vermerkt der Server Geräte jenseits der Aufbewahrungsfrist („must perform a full resync“).

Störungen und Vorfälle

Verlorenes oder gestohlenes Gerät

1. Im Panel unter **Geräte** das Gerät **entfernen** – die Bindung wird gelöst und der Lizenzplatz frei.
2. Bei Verdacht auf Kompromittierung zusätzlich das **Benutzerkonto deaktivieren** oder das Passwort zurücksetzen, MFA beziehungsweise Passkeys zurücksetzen.
3. Vorgang im Audit nachvollziehen.

Die Daten auf dem Gerät bleiben durch App-Sperre und lokale Verschlüsselung geschützt.

Fachkraft aus der App-Sperre ausgesperrt

Geräte → PIN-Sperre zurücksetzen. Die Sperre löst sich beim nächsten Sync; die Person vergibt eine neue PIN.

Kompromittiertes Nutzerkonto

Konto deaktivieren, Passwort zurücksetzen, MFA und Passkeys zurücksetzen, Geräte prüfen und bei Bedarf entfernen. Alle Schritte stehen im Audit.

Kompromittierte Serverschlüssel

Signier- und Verschlüsselungszertifikate austauschen und die Anwendung neu starten; alle Tokens werden dadurch ungültig und die Clients melden sich neu an. Das bisherige `encryption.pfx` als `deploy/docker/keys/encryption.previous.pfx` aufbewahren und `OpenIddict__PreviousEncryptionCertificatePath: /keys/encryption.previous.pfx` setzen (Passwort optional über `OpenIddict__PreviousEncryptionCertificatePassword`, sonst `CERT_PASSWORD`): Der Data-Protection-Schlüsselring enthält Schlüssel, die mit dem alten Zertifikat verschlüsselt sind, und bleibt nur so lesbar (Lizenz-Installationsidentität, SSO-Geheimnisse). Das alte Zertifikat nicht löschen, solange solche Schlüssel existieren. Den Offline-Grant-Schlüssel über die beschriebene **Zwei-Phasen-Rotation** wechseln.

Datenbank- oder Backup-Diebstahl

Blob-Inhalte sind ruhend verschlüsselt; ohne den Blob-Schlüssel bleiben Anhänge unlesbar. Wechseln Sie Datenbankpasswörter und prüfen Sie im Audit, welche Zugänge betroffen sein könnten. Dokumentieren Sie den Vorfall für die Meldepflichten nach DSGVO.

Verdacht auf bösartigen Upload

ClamAV verweigert betroffene Dateien bereits beim Hochladen. Prüfen Sie die Protokolle des Scanners und informieren Sie die betroffene Fachkraft.

Fehlerbehebung

Symptom	Ursache und Abhilfe
`/health/ready` liefert 503	Datenbank nicht erreichbar oder Migrator nicht gelaufen. `docker compose logs migrator` prüfen; stimmen `POSTGRES_PASSWORD` und `FOKUS_APP_PASSWORD` ?
App-Container beendet sich beim Start (Produktion)	Die Startprüfung hat fehlende oder unlesbare Einstellungen gefunden. `docker compose logs app` listet sie unter „Focus Family cannot start“ mit dem jeweiligen Erzeugungsbefehl (z. B. `OFFLINE_GRANT_KEY` , `BLOB_ENCRYPTION_KEY_BASE64` , `deploy/docker/keys/*.pfx` / `CERT_PASSWORD` , `SERVER_INSTANCE_ID` , Erst-Admin bei leerer Datenbank). Werte in der `.env` ergänzen und `docker compose up -d` ausführen
401- oder Weiterleitungsschleifen über den Proxy	`X-Forwarded-Proto=https` wird nicht weitergereicht, der Proxy liegt weder in `REVERSE_PROXY_NETWORK` noch auf `REVERSE_PROXY_IP` , oder `X-Forwarded-Host` passt nicht zu `REVERSE_PROXY_ALLOWED_HOST`
Alle Geräte teilen sich ein Rate-Limit	Die App vertraut dem Proxy nicht: `docker network inspect focusfamily_web` muss das Subnetz aus `REVERSE_PROXY_NETWORK` zeigen (sonst Netz per `down` / `up` neu anlegen), ein Proxy auf dem Host braucht `REVERSE_PROXY_IP`
`docker compose up` meldet „Pool overlaps“	Der Bereich aus `REVERSE_PROXY_NETWORK` ist auf dem Host belegt – ein freies `/24` eintragen
Update aus dem Panel endet mit `downgrade_refused` oder `version_mismatch`	Das angebotene Image ist nicht neuer als das laufende bzw. trägt nicht die angeforderte Version – kein Fehler der Installation, Feed prüfen
Lizenzierung meldet den Enrollment-Key als bereits verwendet	Volume `licstate` wurde nach dem Enrollment gelöscht – aus dem Backup zurückspielen oder neuen Key anfordern
Uploads bleiben in Quarantäne	ClamAV ist noch nicht bereit (Signatur-Download beim ersten Start dauert Minuten). In Produktion ist der Scan absichtlich fail-closed
Geräte synchronisieren nicht	Gerät nicht freigegeben, Lizenzplatz fehlt oder die Mindest-Clientversion sperrt die App aus
Panel meldet „Lizenzintegration nicht konfiguriert"	`Licensing:InstallationId` und `Licensing:PrivateKeyPem` setzen
Anmeldung über SSO scheitert	Code im Parameter `error=` der Anmeldeseite in der Tabelle *Fehlerbehebung bei SSO* nachschlagen
Build schlägt mit `**/*.resx` fehl (Docker Desktop für macOS)	bekannte MSBuild-Eigenheit unter virtiofs – auf einem Linux-Host bauen oder das vorbereitete Image verwenden

---

Checklisten

Inbetriebnahme

- [ ] Domain, DNS und TLS-Zertifikat vorhanden

- [ ] Image geladen, `.env` vollständig ausgefüllt, keine `CHANGE_ME`-Werte mehr
- [ ] Signier- und Verschlüsselungszertifikate erzeugt, `CERT_PASSWORD` gesetzt
- [ ] `BLOB_ENCRYPTION_KEY_BASE64` erzeugt und **getrennt** gesichert
- [ ] `OFFLINE_GRANT_KEY` erzeugt (ganzes PEM in doppelten Anführungszeichen) und gesichert
- [ ] Proxy im Netz `focusfamily_web` (oder `REVERSE_PROXY_IP` gesetzt), `REVERSE_PROXY_ALLOWED_HOST` auf den Panel-Namen gesetzt, Proxy reicht Forwarded-Header weiter
- [ ] Weder `docker-compose.noclamav.yml` noch `docker-compose.local.yml` im Einsatz
- [ ] Erst-Admins beim ersten Start angelegt, Startpasswörter bei der ersten Anmeldung ersetzt, `BOOTSTRAP_*`-Werte danach aus der `.env` entfernt
- [ ] `/health/ready` liefert 200, Anmeldung am Panel erfolgreich
- [ ] Lizenz-Enrollment durchgeführt, `licstate` gesichert
- [ ] Backup-Skript eingerichtet und ein **Test-Restore** durchgeführt
- [ ] Monitoring eingerichtet: `/health/ready` für die Erreichbarkeit, unter *System → Monitoring* einen Zugriffsschlüssel erzeugt und die Schnittstelle eingeschaltet, Alarme auf `fokusfamilie_ready`, `fokusfamilie_disk_used_ratio` und `fokusfamilie_privacy_requests_overdue`
- [ ] Beim Lizenzserver `Monitoring__ScrapeToken` gesetzt (sonst bleibt dessen Schnittstelle aus)

Monatliche Routine

- [ ] Backups vorhanden und nicht älter als vereinbart
- [ ] Test-Restore in eine Staging-Umgebung
- [ ] Serverversion und Sicherheitsupdates prüfen
- [ ] Audit auf Auffälligkeiten durchsehen
- [ ] Geräte- und Lizenzplatzliste aufräumen
- [ ] Plattenbelegung von `pgdata` und `blobs` prüfen
- [ ] Monitoring: Schlüssel noch gültig, Abrufe kommen an, keine Daueralarme

---

Anhang

Wichtige Pfade und Ports

Gegenstand	Wert
App im Container	Port 8080, gebunden an `127.0.0.1`
PostgreSQL	nur im internen Compose-Netz
Blob-Speicher	`/var/lib/fokusfamilie/blobs` (Volume `blobs`)
Data-Protection-Keys	`/home/app/.aspnet/DataProtection-Keys` (Volume `dpkeys`)
Lizenz-Identität	`/var/lib/fokusfamilie/licensing/installation.json` (Volume `licstate`)
Zertifikate	`deploy/docker/keys/*.pfx` (read-only eingebunden)

Nützliche Befehle

```
docker compose ps                       # Zustand aller Dienste
docker compose logs -f app              # Anwendungsprotokoll
docker compose restart app              # nur die Anwendung neu starten
docker compose down                     # Stack stoppen (Volumes bleiben)
docker compose down -v                  # Stack inklusive Volumes entfernen (Datenverlust!)
./deploy/docker/backup.sh               # Sicherung erstellen
```

Weiterführende Dokumentation im Repository

`docs/server/DEPLOYMENT.de.md`, `docs/server/DATABASE.de.md`, `docs/server/BACKUP-RESTORE.de.md`, `docs/server/UPGRADE.de.md`, `docs/server/SECURITY.de.md`, `docs/server/AUTHENTICATION.de.md`, `docs/server/AUTHORIZATION.de.md`, `docs/server/SYNC-PROTOCOL.de.md`, `docs/server/INCIDENT-RESPONSE.de.md`, `docs/server/THREAT-MODEL.de.md`, `docs/server/SSO.de.md` (Entwicklerreferenz) sowie `deploy/docker/README.de.md`.

Focus Family · Technisches Handbuch · Version 1.3.0 · Stand Oktober 2026