SEC-1c Plan-Review — Transport / Privilege / Operational Hardening

Projekt: Steve-TradingBot · Phase: SEC-1c · Author: claude-opus-4-7[1m]
Generated: 2026-05-13 10:46 UTC · master HEAD: `2fbc7b5` (SEC-1b-6 closed)
Status: **NO CODE** Plan-Review only — Operator-GO erforderlich vor SEC-1c-Code-Phase
Empfohlene Reihenfolge:
SEC-1c-1 HTTPS → SEC-1c-2 Sessions → SEC-1c-3 DB-LP → SEC-1c-4 Admin-Rotation → CSP → Audit-Alerts → D Net-Seg → C SSH+A+B → E Secret-Rotation

Ziel: Nach SEC-1b-6 (Identity-Layer abgesichert) jetzt Transport-Sicherheit, Privilege-Separation, Operational-Security und Browser-Hardening abschließen, BEVOR MH-5b/c/d Mutation-Actions ausgeliefert werden. **Wichtigste Operator-Pin**: *„DB-Least-Privilege ist kein optional nice-to-have mehr — nach dem DB-Wipe-Incident ist es ein struktureller Sicherheitsblocker für Production-Härtung.“*

---

1 — Live-State-Check (Boundaries)

Domain	Item	Current	Risk
Transport	nginx server_name	`81.169.213.37` (IP-only)	**HIGH** — blockt SEC-1d Passkey; HSTS sinnlos auf IP
Transport	TLS cert	self-signed `/root/steves-tradingbot/ssl/dashboard.crt`	**HIGH** — Browser-Trust-Warnung; kein Let's Encrypt installiert
Transport	`APP_URL`	`http://127.0.0.1:8090` plain HTTP	**HIGH** — Filament generiert HTTP-Links; Secure-Cookies wirkungslos
Transport	nginx proxy-target	`proxy_pass http://127.0.0.1:8050` (Bot-Dashboard, 502 currently)	**MEDIUM** — GUI auf :8090 ist nicht hinter nginx; nur lokal erreichbar
Transport	Bot-Dashboard exposure	`0.0.0.0:8050` (compose port)	**MEDIUM** — öffentlich erreichbar ohne Auth-Layer
Session	`SESSION_ENCRYPT`	`false`	**HIGH** — Session-Payload Klartext in DB
Session	`SESSION_SECURE_COOKIE`	nicht gesetzt	**HIGH** — Cookie ohne `Secure`-Flag
Session	`SESSION_HTTP_ONLY`	nicht gesetzt (default true)	**MEDIUM** — default ist sicher; explizit pinnen
Session	`SESSION_SAME_SITE`	nicht gesetzt (Laravel-default: `lax`)	**MEDIUM** — `strict` empfohlen
Session	`TRUSTED_PROXIES` / Middleware	nicht konfiguriert (`bootstrap/app.php` ohne `trustProxies()`)	**HIGH** — Laravel erkennt HTTPS-via-nginx falsch → Secure-Cookie verliert Secure-Flag
Session	`SESSION_LIFETIME`	`120` (Minuten)	**OK** — Operator-Empfehlung 15–30 min für Admin-Panel
Privilege	DB-Runtime-User	`tradingbot_gui` = **SUPERUSER + CREATE ROLE + CREATE DB + REPLICATION + BYPASS RLS**	**CRITICAL** — Worst-Case; kompromittierter GUI-Prozess kann komplette DB löschen (genau der INCIDENT-Vektor)
Privilege	Migration-User getrennt	nein — gleicher User	**HIGH** — INCIDENT 2026-05-12 Vektor weiterhin offen
Ops	Admin-User	`admin@example.local`	**MEDIUM** — Default-User, technisch+organisatorisch Risiko
Browser	CSP-Header	kein `Content-Security-Policy` gesetzt	**MEDIUM** — XSS-/Clickjacking-Schutz fehlt
Browser	`X-Frame-Options`	nicht gesetzt	**MEDIUM**
Browser	`X-Content-Type-Options`	nur für `/shares/` gesetzt (`nosniff`)	**MEDIUM** — nicht für Admin-Panel-Route
Network	GUI-Container port	`127.0.0.1:8090 → container:8000`	**OK** — bereits localhost-bound
Network	GUI-DB port	kein Host-Mapping (intern `gui-db:5432`)	**OK**
Network	Docker-Netze	`steve-tradingbot_clawbot-net` + `bridge` · GUI+DB im gleichen Netz	**OK** — aber Segmentierung prüfen
Network	Bot-Dashboard 8050	`0.0.0.0:8050`	**MEDIUM** — offen; sollte hinter nginx-auth oder localhost-only
Visibility	`audit_events`-Table	vorhanden (RECON-MH-Pattern)	**OK** — Hook-Punkte vorhanden, aber keine Alerts
Visibility	Telegram-Notifier	vorhanden (Notifier-1)	**OK** — nutzbar für Audit-Alerts
Hygiene	fail2ban	aktiv (sshd-jail)	**OK** — SEC-1b-0 closure
Hygiene	SSH `PasswordAuthentication`	SEC-1b-0: `prohibit-password` für root; gen sshd ungeprüft	**MEDIUM** — global `no` + `AllowGroups` prüfen
Boundaries	Bot PID (clawbot)	290 unverändert (`python3 main.py --paper`)	**OK**
Boundaries	Worker PID (clawbot-worker)	1 unverändert	**OK**
Boundaries	BINANCE_TESTNET	`true`	**OK**
Boundaries	cmd 13 / managed_proposals	cancelled / 0 rows	**OK**

Kritischste Live-Findings

1. **DB-Runtime-User ist SUPERUSER** — Worst-Case für Privilege-Separation. SEC-1c-3 ist post-INCIDENT der wichtigste strukturelle Block.
2. **Plain-HTTP + self-signed Cert + IP-only** — HTTPS-Setup ist Voraussetzung für alles andere (Cookies, HSTS, Passkey-BACKLOG).
3. **Keine FQDN registriert** — Operator-Decision erforderlich (Domain-Wahl blockt SEC-1d dauerhaft, wenn ungeklärt).
4. **TrustProxies-Middleware fehlt** — muss zusammen mit HTTPS aktiviert werden, sonst entwertet sich SEC-1c-2 selbst.

---

2 — SEC-1c-1 HTTPS / Let's Encrypt HIGH

Production-Topology (Soll)

```
Internet
   |
   v
nginx :443 (Let's Encrypt cert)   ----+
   |                                  |
   +---> Admin GUI Filament @127.0.0.1:8090
   |
   +---> /shares/ static (unchanged)
   |
nginx :80  ---> force-redirect :443 (außer /.well-known/acme-challenge)
```

Sub-Items

1. **Domain-Wahl** — Operator-Decision erforderlich:
   - (a) Eigene Subdomain (z. B. `trading.kw-baustoffe.de` oder `steve.kw-baustoffe.de`) — **Empfehlung**: erlaubt später SEC-1d Passkey ohne Detour.
   - (b) Neue dedizierte Domain (kostet evtl. Registrar-Fee + DNS-Setup).
   - (c) IP-only mit self-signed bleiben — **NICHT empfohlen**: Passkey-BACKLOG bleibt dauerhaft blockiert; Operator-Warnung dokumentiert.
2. **Let's Encrypt Setup** — `certbot` + `--nginx`-Plugin ODER `--standalone` ODER `--webroot`:
   - **Empfehlung**: `--webroot /var/www/letsencrypt` + dedizierter `/.well-known/acme-challenge`-Block in nginx port 80. Kein nginx-Downtime nötig.
   - Cron / systemd-timer für `certbot renew --quiet` alle 12h.
3. **nginx-Config-Patch** (Plan only):
   - port 443 ssl — cert von Let's Encrypt
   - `ssl_protocols TLSv1.2 TLSv1.3`
   - `ssl_ciphers HIGH:!aNULL:!MD5` (bestehend)
   - `ssl_prefer_server_ciphers on`
   - `ssl_session_timeout 1d; ssl_session_cache shared:SSL:50m`
   - `add_header Strict-Transport-Security "max-age=31536000; includeSubDomains" always;` (HSTS, kein preload zunächst)
   - port 80 → 301 redirect zur HTTPS-Variante (Außer ACME-Challenge)
   - neuer `location /admin`-Block → `proxy_pass http://127.0.0.1:8090` (GUI Filament)
   - `proxy_set_header X-Forwarded-Proto $scheme;` + `X-Forwarded-For` + `Host`
4. **Laravel-Konfiguration**:
   - `APP_URL=https://<FQDN>`
   - kombiniert mit SEC-1c-2 TrustProxies (siehe nächstes Item).
5. **Rollback-Pfad**: alte nginx-conf als `.bak` sichern; bei Cutover-Fehler < 60s zurückspielen.

Risiken

- **HSTS-Falle**: einmal an Browser ausgeliefert, blockt es HTTP-Fallback für `max-age`. Empfehlung: max-age zunächst niedrig (z. B. 86400 = 1 Tag), nach 1 Woche auf 1 Jahr hochziehen.
- **Let's Encrypt Rate-Limit**: 50 Zertifikate/Domain/Woche. Im Test-Pfad `--staging` verwenden, dann produktiv.
- **Mixed-Content**: Filament asset-paths müssen HTTPS sein → `config/filament.php` `asset_url` ggf. setzen, oder `APP_URL` mit Laravel `URL::forceScheme('https')`.

Aufwand

Domain-Setup (DNS A-Record) 5–30min, certbot 5min, nginx-config 30min, Test 15min. **Σ 1–2h inkl. Rollback-Plan**.

---

3 — SEC-1c-2 Session Hardening HIGH

Minimum-Set (.env)

```
SESSION_ENCRYPT=true
SESSION_SECURE_COOKIE=true
SESSION_HTTP_ONLY=true
SESSION_SAME_SITE=strict
SESSION_DOMAIN=<FQDN>             # exakt die Domain ohne Schema
TRUSTED_PROXIES=*                  # in unserem nginx-localhost-Pfad sicher
```

TrustProxies-Middleware (Plan only)

In `gui/bootstrap/app.php` im `->withMiddleware()`-Block:

```
$middleware->trustProxies(at: '*', headers: \Illuminate\Http\Request::HEADER_X_FORWARDED_FOR
  | \Illuminate\Http\Request::HEADER_X_FORWARDED_HOST
  | \Illuminate\Http\Request::HEADER_X_FORWARDED_PORT
  | \Illuminate\Http\Request::HEADER_X_FORWARDED_PROTO);
```

Begründung: nginx läuft lokal (localhost-only Proxy), `at: '*'` ist akzeptabel; bei externalisiertem Reverse-Proxy müssten wir die CIDR-Liste pinnen.

Session-Lifetime (Operator-Pin: 15–30 min Admin-Idle-Timeout)

- **Variante A**: `SESSION_LIFETIME=30` global (alle User-Rollen).
- **Variante B**: `SESSION_LIFETIME=120` bleibt; Admin-Panel-spezifisch `30` via Filament-Middleware-Override (komplexer, aber Viewer-friendlier).
- **Empfehlung**: **A** — einfacher, geringeres Drift-Risiko; 30 min Idle ist auch für Operator-/Viewer-Rollen vertretbar.

Risiken

- **Logout aller offenen Sessions** beim Cutover (Session-Encrypt ändert das Serialisierungsformat). Operator + bisherige Test-User müssen sich neu anmelden — in unserem 1-Admin-Setup unkritisch.
- **strict SameSite** kann externe Links auf `/admin/...` (z. B. Telegram-Notifier-Links) brechen — prüfen, ob aktuell solche Links existieren. `lax` als Fallback dokumentiert.

Aufwand

Σ 30–60min inkl. Test der Cookie-Flags + Re-Login.

---

4 — SEC-1c-3 DB Least-Privilege (2-User-Modell) CRITICAL

Aktueller Zustand (live verifiziert): `tradingbot_gui` ist `Superuser, Create role, Create DB, Replication, Bypass RLS`. Das ist exakt der Vektor, der den INCIDENT 2026-05-12 möglich gemacht hat.

Soll-Modell

Rolle	Zweck	Privs erlaubt	Privs verboten
`tradingbot_gui_app`	Runtime (Laravel im GUI-Container)	`CONNECT` auf DB; `USAGE` auf `public`-Schema; `SELECT/INSERT/UPDATE/DELETE` auf alle Tables; `USAGE/SELECT` auf alle Sequences; `EXECUTE` auf Functions; `TEMPORARY` auf DB	kein `CREATE`, kein `DROP`, kein `ALTER`; kein `SUPERUSER/CREATE ROLE/CREATE DB/REPLICATION`
`tradingbot_gui_migrator`	Schema-Migrationen (`php artisan migrate`)	`CONNECT`; `USAGE/CREATE` auf `public`; `CREATE/ALTER/DROP TABLE/INDEX/SEQUENCE/FOREIGN KEY/CONSTRAINT`; `SELECT/INSERT/UPDATE/DELETE` auf alle (für data-migrations)	kein `SUPERUSER/CREATE DB/CREATE ROLE/REPLICATION`
`tradingbot_gui` (legacy)	Owner / DBA-Notfall (von außen SSH+psql)	SUPERUSER bleibt	aber: nie in GUI-Container `.env`

Cutover-Sub-Plan

1. **pg_dump-Backup** (durable rule) vor jedem Schritt.
2. Neue Rollen anlegen via `psql` (kein Migration, keine Schema-Änderung).
3. `GRANT`s explizit setzen + `ALTER DEFAULT PRIVILEGES` für zukünftige Tables.
4. **Test-Phase**: GUI-Container weiterhin auf `tradingbot_gui` (SUPERUSER); parallel `psql`-Connect-Tests mit `tradingbot_gui_app`:
   - SELECT auf alle Tables darf gehen
   - `CREATE TABLE x()` muss fehlschlagen
   - `DROP TABLE users` muss fehlschlagen
   - `ALTER SYSTEM` muss fehlschlagen
5. **Cutover**: `.env` in GUI-Container wechseln `DB_USERNAME=tradingbot_gui_app` → `php artisan config:clear` → GUI-Container restart (oder `DB::reconnect()` via Artisan, je nach Decision).
6. **Migration-Pfad**: `scripts/migrate.sh` ODER manueller Befehl `DB_USERNAME=tradingbot_gui_migrator php artisan migrate` ODER eigener Migration-Container.
7. **SUPERUSER-Lock** (Final): `ALTER USER tradingbot_gui WITH NOSUPERUSER NOCREATEROLE NOCREATEDB` — nur noch für Notfall-Zugriff.

Decision-Punkt

Cutover-Method	Pro	Contra
(a) GUI-Container restart nach `.env`-Edit	Sauber, eindeutiger Cut	~10s Downtime für GUI; bestehende Sessions invalidiert (kombiniert sich mit SEC-1c-2 Re-Login — gut!)
(b) `DB::reconnect()` via Artisan ohne Restart	Keine Downtime	Brittle: Filament-internes Connection-Pool bleibt evtl. auf altem User; PHP-FPM-Worker müssen alle reconnecten

Empfehlung: (a) — klar, atomic, kombiniert mit SEC-1c-2-Cutover sinnvoll.

Risiken

- **Verpasste GRANTs** → Laufzeit-Fehler nach Cutover. Mitigation: vor Cutover Test-Connection-Suite mit `tradingbot_gui_app` gegen alle Eloquent-Models (manuell oder via Health-Endpoint).
- **Sequences/Composite-PKeys** brauchen separate GRANTs.
- **Future-Tables**: `ALTER DEFAULT PRIVILEGES` für `tradingbot_gui_app` + `tradingbot_gui_migrator` setzen, sonst muss bei jeder neuen Tabelle nachgegrantet werden.

Aufwand

Σ 2–3h inkl. Test-Suite (manual SQL + 1 PHPUnit-Test „db-user-cannot-DROP“).

---

5 — SEC-1c-4 Admin Rotation HIGH

Plan

1. **Neuen Admin anlegen** via Filament-CLI ODER seeder:

- Email: Operator-Pin erforderlich (z. B. `steve@kw-baustoffe.de` oder `talk@kw-baustoffe.de`)
- Initial-Passwort: stark, im Password-Manager gespeichert
- Role: `UserRole::Admin`
- `app_authentication_secret`: NULL (forced-enrollment on first login — SEC-1b-6 Default)

2. **Login + MFA-Enrollment** mit echtem Authenticator-App; 8 Recovery-Codes sicher ablegen.
3. **Test-Phase**: Login + Admin-Aktion (z. B. Resource lesen).
4. **Alten Admin deaktivieren** (NICHT löschen — Audit-Trail erhalten):
   - (a) Neue Spalte `users.is_active` + Filament-Login-Pre-Check — **komplexer** (extra Migration + Test-Suite-Update)
   - (b) `UPDATE users SET email='disabled-admin@local.invalid', password='', role='viewer', app_authentication_secret=NULL WHERE id=23` — **minimal-invasiv**, kein Schema-Touch

   Empfehlung: (b) — passt zu Scope-Lock „klein/atomar“. Spalte `is_active` kann in BACKLOG SEC-1c-FU-1 nachgezogen werden.

Übergangsphase

Variante	Pro	Contra
(i) 2 Admins parallel während Test-Phase	Lockout-Safety: alter Admin als Backup	Längerer high-risk-Zustand
(ii) Hard-Cutover	Schnell, klar	Wenn neuer Admin MFA-Setup verfehlt → Lockout, Break-Glass-Pfad nötig

Empfehlung: (i) — 2 Admins parallel; alten Admin erst deaktivieren nachdem neuer Admin **vollständig** (Login + MFA + Recovery-Codes gesichert) verifiziert wurde.

Aufwand

Σ 30min.

---

6 — SEC-1c-5 CSP / Security Headers (2-phasig) MEDIUM

Phase 1 — Safe-Defaults (gleichzeitig mit HTTPS-Cutover deploybar)

Header in `nginx`-Server-Block (HTTPS) — `add_header` mit `always`-Flag:

```
add_header Strict-Transport-Security "max-age=86400; includeSubDomains" always;
add_header X-Frame-Options "DENY" always;
add_header X-Content-Type-Options "nosniff" always;
add_header Referrer-Policy "strict-origin-when-cross-origin" always;
add_header Permissions-Policy "geolocation=(), microphone=(), camera=()" always;
```

Phase 2 — Moderate CSP (später, gestaffelt)

```
Content-Security-Policy:
  default-src 'self';
  img-src 'self' data:;
  style-src 'self' 'unsafe-inline';
  script-src 'self' 'unsafe-inline';
  font-src 'self' data:;
  connect-src 'self';
  frame-ancestors 'none';
  base-uri 'self';
  form-action 'self';
```

Operator-Warnung: Filament nutzt Inline-Styles + Livewire-Inline-Scripts; strict-CSP würde die Admin-UI brechen. `unsafe-inline` bleibt zunächst, später nonce-basiert in BACKLOG SEC-1c-FU-2.

Risiken

- CSP-Report-Mode (`Content-Security-Policy-Report-Only`) zuerst nutzen für 1 Woche; dann auf enforce schalten.
- HSTS-max-age erst nach 1 Woche fehlerfrei auf `31536000` + `preload` hochziehen.

Aufwand

Phase 1: **20min**. Phase 2: **1–2h** (Test + Adjust).

---

7 — SEC-1c-6 Audit-Alerts MEDIUM

Trigger

Event	Quelle	Notification-Severity
MFA Setup completed	Filament `SetUpAppAuthenticationAction`-Hook	info
MFA disabled / regenerated	Filament Disable+Regenerate-Actions	**warning**
MFA Break-Glass (DB-direkt-UPDATE)	DB-Trigger ODER manueller Audit-Log-Eintrag	**critical**
Failed login attempt	Filament Login-throttle	info (Aggregat alle 10)
3 fehlgeschlagene MFA-Codes	Filament MFA-Step-Validator	warning
Successful login from new IP	session create + last_login_ip-cache	warning
Successful login from new ASN/country	+ GeoIP-Lookup (SEC-2 / BACKLOG)	warning
Role escalation (UPDATE users.role)	Eloquent observer ODER DB-Trigger	**critical**
Command requested by viewer (denied)	Filament canCreate/canEdit-checks	warning

Hook-Design

1. Bestehende `audit_events`-Tabelle nutzen (RECON-MH-Pattern, schon im Einsatz).
2. Eloquent-Observer auf `User`-Model für role-changes + MFA-changes.
3. Filament-Action-Hook (after) für MFA-Setup/Disable/Regenerate.
4. Telegram-Notifier (Notifier-1 bereits live) nutzt audit_event-Inserts als Trigger via Queued Job ODER Eloquent Observer.
5. Rate-Limit: max 3 Alerts/Minute/Severity, sonst Aggregat-Mode.

Aufwand

Σ 3–4h inkl. Tests.

---

8 — A + B Defense-in-Depth (Session-Lifetime + nginx-Rate-Limits) MEDIUM

A. Session-Lifetime-Review

- `SESSION_LIFETIME=30` (Variante A in §3) → Idle-Logout nach 30 min.
- `SANCTUM_STATEFUL_DOMAINS` — prüfen; aktuell vermutlich nicht relevant (kein API-Token-Setup), aber dokumentieren.
- Filament `EditProfile` „Sign out from all other browsers"-Aktion aktivieren (Built-in Feature) — **BACKLOG**.

B. nginx Rate-Limits

```
# /etc/nginx/conf.d/limits.conf
limit_req_zone $binary_remote_addr zone=admin_login:10m rate=10r/m;
limit_conn_zone $binary_remote_addr zone=admin_conn:10m;

# in HTTPS server-block:
location /admin/login {
    limit_req zone=admin_login burst=5 nodelay;
    limit_conn admin_conn 3;
    proxy_pass http://127.0.0.1:8090;
    ...
}
```

Defense-in-Depth zu Laravel-Login-Throttle (`MAX_LOGIN_ATTEMPTS=3`); blockt L7-Floods bevor sie Laravel erreichen.

Aufwand

Σ 30min.

---

9 — C SSH-Härtung LOW

Aktueller Stand (SEC-1b-0)

- `PermitRootLogin prohibit-password` aktiv (key-only für root)
- fail2ban-jail aktiv (sshd)

Zusätzlich

- `PasswordAuthentication no` global (nicht nur für root)
- `ChallengeResponseAuthentication no`
- `UsePAM yes` (default)
- `AllowGroups ssh-users` — explicit-Whitelist auf eine Unix-Group; nur Member dürfen SSH
- `MaxAuthTries 3`
- `LoginGraceTime 30`
- **Port-Knocking NICHT nötig** (Operator-Pin)

Risiken

- **Lockout-Gefahr** wenn AllowGroups gesetzt wird ohne dass aktuelle User in der Gruppe sind. Mitigation: zuerst Gruppe anlegen + User adden, dann sshd-config ändern, dann `sshd -t` testen, dann reload.
- Konsole-Zugriff (lokales Login) als Fallback bestehend.

Aufwand

Σ 30min.

---

10 — D Docker Netzwerksegmentierung MEDIUM

Aktuell

- `steve-tradingbot_clawbot-net`: enthält GUI + GUI-DB
- Bot-Container in `clawbot-docker_clawbot-net` (separates Netz)
- Bot-Dashboard `0.0.0.0:8050` public — **Prüfen ob hinter nginx-Auth oder localhost-only stellen**
- GUI-DB hat kein Host-Port-Mapping (intern via `gui-db:5432`) — **OK**

Soll-Plan

1. **Prüfung**: `docker network inspect steve-tradingbot_clawbot-net` → sind GUI + DB die einzigen Member?
2. **Bot-Dashboard 8050**: 3 Optionen:
   - (a) localhost-only binden: `127.0.0.1:8050:8050` in compose; nginx-proxy mit Basic-Auth davor.
   - (b) Bot-Dashboard hinter Filament-Admin-Auth migrieren (größere Refactor — BACKLOG).
   - (c) Bot-Dashboard **löschen** wenn ohnehin nicht mehr aktiv genutzt (Operator-Decision).

   Empfehlung: (a) als SEC-1c-Item; (c) als BACKLOG.
3. **Netzwerk-Audit**: `iptables -L` + `ss -tlnp` auf Host — finden aller 0.0.0.0-Bindings; dokumentieren.

Aufwand

Σ 1h Prüfung + 30min Cutover Bot-Dashboard.

---

11 — E Secret-Rotation (mittelfristig) MEDIUM

Rotations-Liste (post-INCIDENT-Hygiene)

Secret	Risiko bei Rotation	Schritte
`APP_KEY`	**HOCH** — rotiert encryptet alle `encrypted`-cast-Spalten (MFA-Secret, Recovery-Codes, ggf. weitere)	Eigene Sub-Phase mit `APP_PREVIOUS_KEYS`-Compat-Layer; pg_dump vorher; alle Admins reset MFA
DB-Passwort `DB_PASSWORD`	Mittel — GUI-Container muss reconnecten	2-Step: neuer User mit neuem PW (siehe SEC-1c-3); altes PW dann ALTER USER ... PASSWORD
Telegram-Bot-Token	Niedrig	Neuen Token erzeugen, alten widerrufen; Notifier testen
OpenAI-Key (falls genutzt)	Niedrig	Im OpenAI-Dashboard rotieren, neuen Key in `.env` setzen
SSH-Authorized-Keys	Mittel — Lockout-Gefahr	Neuen Key generieren, hinzufügen, alten *nach* Test-Login entfernen
Recovery-Codes (alle bestehenden)	Niedrig	Filament-Action „Regenerate Recovery Codes“; alte werden invalidiert

Empfehlung

Eigene Sub-Phase SEC-1c-7 als letztes Item. APP_KEY-Rotation ist riskant; sollte separat geplant werden mit eigener pg_dump-Vorab-Sicherung + Test-Run im Staging.

Aufwand

Σ 3–4h inkl. APP_KEY-Compat-Layer + Test-Cycle.

---

12 — Atomicity & Commit-Block-Plan

Vorschlag (operator-anpassbar)

#	Block	Items	Restart-Pflicht	Risiko	Aufwand
1	**HTTPS + Sessions + Phase 1 Headers + B nginx-RL**	SEC-1c-1 + SEC-1c-2 + SEC-1c-5 Phase 1 + B	nginx-reload + GUI-Container-Restart	HIGH (TLS-Cutover, Cookie-Cutover atomar)	2–3h
2	**DB-Least-Privilege (2-User-Modell)**	SEC-1c-3	GUI-Container-Restart	HIGH (Connection-Cutover)	2–3h
3	**Admin-Rotation**	SEC-1c-4	kein Restart	MEDIUM (Lockout-Gefahr)	30min
4	**Audit-Alerts + A Session-Lifetime**	SEC-1c-6 + A	config-clear, kein Restart	LOW	3–4h
5	**Phase 2 moderate CSP**	SEC-1c-5 Phase 2	nginx-reload	MEDIUM (UI-Breakage-Risk)	1–2h
6	**D Docker-Netzwerksegmentierung**	D + Bot-Dashboard localhost-bind	compose-up für Bot-Dashboard	LOW	1–2h
7	**C SSH-Härtung**	C	sshd-reload	LOW (lokaler Konsolen-Fallback bleibt)	30min
8	**E Secret-Rotation** (SEC-1c-7)	E	variabel	HIGH (APP_KEY)	3–4h

Bot/Worker bleiben in allen 8 Blöcken unberührt. `BINANCE_TESTNET=true` bleibt. Bot-PID 290 + Worker-PID 1 unverändert pinnen.

Gesamt-Aufwand SEC-1c: **14–22h** über 8 atomic commits, jeder mit eigenem pg_dump-Backup + Safe-Runner-Tests + Live-State-Verify.

---

13 — Operator-Decision-Points (Pflicht vor Block 1)

1. **Domain für GUI**:
   - (a) Subdomain einer bestehenden Domain (z. B. `trading.kw-baustoffe.de`) — **Empfehlung**
   - (b) Neue dedizierte Domain

    - (c) IP-only bleiben (blockt SEC-1d Passkey dauerhaft)
2. **DB-Cutover-Method** (SEC-1c-3):
    - (a) Container-Restart nach `.env`-Edit — **Empfehlung**
    - (b) `DB::reconnect()` ohne Restart
3. **Admin-Rotations-Übergang**:
    - (i) 2 Admins parallel für 1–7 Tage Test-Phase — **Empfehlung**
    - (ii) Hard-Cutover (alten Admin sofort deaktivieren)
4. **Neue Admin-Email**: konkreter Wert? (z. B. `talk@kw-baustoffe.de` oder `steve@kw-baustoffe.de`)
5. **Session-Lifetime**: 30 min global ODER 30 min Admin-only + 120 min sonst?
6. **Bot-Dashboard 8050**: localhost-bind + nginx-Auth davor (a) ODER löschen wenn ungenutzt (c)?
7. **HSTS-Strategie**: max-age 1 Tag → 1 Jahr nach 1 Woche fehlerfrei (Empfehlung) ODER sofort 1 Jahr?
8. **CSP Phase 1 jetzt + Phase 2 später** ODER beide zusammen?

---

14 — NO-GO-Bedingungen (durable, pro Block)

- **NO-GO**: jeder Block ohne pg_dump-Backup vorab.
- **NO-GO**: jeder Test ohne `bash gui/scripts/run_tests_safe.sh`.
- **NO-GO**: Bot-Code-Touch, Worker-Code-Touch, docker-cp auf Bot-Files, Bot/Worker-Restart.
- **NO-GO**: Mainnet-Enable, `BINANCE_TESTNET=false`, irgendein Mainnet-Order-API-Touch.
- **NO-GO**: Push ohne Operator-GO.
- **NO-GO**: HTTPS-Cutover ohne Rollback-Plan (alte nginx-conf als `.bak`).
- **NO-GO**: DB-Cutover ohne vorab Test-Connection-Suite mit neuem App-User.
- **NO-GO**: Admin-Rotation ohne dass neuer Admin Login + MFA + Recovery-Codes voll verifiziert hat.
- **NO-GO**: APP_KEY-Rotation ohne `APP_PREVIOUS_KEYS`-Compat oder ohne pg_dump.

---

15 — Empfehlung & nächster Schritt

Empfehlung: Start mit **Block 1 (HTTPS + Sessions + Phase 1 Headers + nginx-RL)** sobald Operator-Decisions 1, 5, 7 beantwortet sind.

Parallel kann **Block 2 (DB-Least-Privilege)** in Test-Phase laufen (Rollen + GRANTs anlegen, Connect-Tests mit `tradingbot_gui_app`) **ohne** Cutover — Cutover dann nach Block 1.

Block 3 (Admin-Rotation) direkt nach Block 1, weil 1) HTTPS aktiv + Cookies sicher = neues Enrollment läuft über sicheren Channel, 2) Operator hat dann frischen Admin im neuen System.

Block 8 (Secret-Rotation, vor allem APP_KEY) ist *letztes* Item — nicht riskieren bevor alles andere stabil läuft.

Status

PLAN-REVIEW Plan komplett. Warte auf:

- (a) Antworten auf Operator-Decision-Points 1–8 (oder Sub-Set — Block 1 braucht nur 1, 5, 7), und/oder
- (b) explizites `GO SEC-1c-1 + 2 + 5 Phase 1 + B Block 1` (oder anderer Start-Block), und/oder
- (c) abweichende Phasen-Vorgabe.

Kein Code geschrieben. Kein git / docker / test-Touch durchgeführt — pure analysis + scope-pin only.