SICHERHEITS-AUDIT — RUNDE 7

NetMon Roots v1.1.18

Code Security Review — Juli 2026

Datum	17. Juli 2026
Analysierte Version	v1.1.18 — Runde 7
Neue Angriffsflaeche	RKHunter-Referenz & Analyse, ClamAV-DB
Bewertete Kandidaten	4 (2 kein Finding)
Ausnutzbare Findings	0
Haertungen umgesetzt	2 (Defense-in-Depth)
Konfidenz-Schwelle	>= 8 / 10
Regressionstests	59 / 59 gruen
Analysewerkzeug	Claude Fable 5 (Anthropic)
Projektverantwortlicher	Sven Froehlich

0 AUSNUTZBARE FINDINGS — 2 DEFENSE-IN-DEPTH-HAERTUNGEN UMGESETZT & VERIFIZIERT

1. Zusammenfassung

Die siebte Sicherheitsanalyse konzentrierte sich auf die seit dem Runde-6-Snapshot neu hinzugekommene Angriffsflaeche im Arbeitsstand — vor allem die RKHunter-Referenzuebernahme (rkhunter --propupd ueber run_rkhunter_propupd() / restore_property_database()), den neuen deterministischen Analysedienst (rkhunter_analysis_service.py) mit seinem propupd-Sicherheits-Gate, die im Scan-Worker neu ausgefuehrten privilegierten Hilfskommandos (dpkg -V/-S, file -b, cat /var/log/rkhunter.log) sowie die neue ClamAV-Signaturdatenbank-Auswahl (clamav_database_args) und die konfigurierbaren Schnellscan-Pfade.

Ein Analyse-Sub-Agent identifizierte vier Kandidaten. Jeder wurde gegengeprueft, ergaenzt um eine empirische Reproduktion des Klassifizierer-Verhaltens am realen Modul. Zwei Kandidaten wurden als 'kein Finding' verworfen, zwei lagen unterhalb der Schwelle von 8/10, wurden aber dennoch als Defense-in-Depth direkt umgesetzt und empirisch verifiziert. Grund fuer die Einstufung unter der Schwelle: In einem Einzelnutzer-Desktop-Werkzeug stammt die vermeintliche 'Angreifer'-Eingabe entweder vom selben lokalen Nutzer oder setzt bereits lokale Root-Rechte voraus — es wird keine echte Vertrauensgrenze ueberschritten.

K1 (Konfidenz 3/10, gehaertet): Die FP-Klassifizierung des propupd-Gates war zu breit — _EXPECTED_CONFIG_RE matchte jede *.conf-Datei, _DEVELOPER_ENV_RE bloße Substrings wie 'docker'/'vscode', wodurch selbst eine nicht paketverwaltete /etc/cron.d/persist.conf als 'expected' galt und das Gate oeffnete. Jetzt gehaertet: Whitelist statt Wildcard und Wortgrenzen fuer Entwicklungs-Pfade. Empirisch gegengeprueft — solche Aenderungen fallen nun nach 'review', das Gate bleibt geschlossen (propupd_allowed=False), legitime Faelle bleiben 'expected'.

K2 (Konfidenz 2/10, gehaertet): Die ClamAV-Signaturdatenbank liegt in einem vorhersagbaren /tmp-Pfad (netmon-clamav-db-). Jetzt gehaertet: _clamav_database_dir_is_secure() laesst clamscan --database nur auf ein Verzeichnis zeigen, das dem aktuellen Nutzer gehoert, kein Symlink ist und keine Gruppen-/Welt-Rechte traegt (& 0o077 == 0). Empirisch gegengeprueft — fremd-owned, welt-/gruppen-zugaengliche und symlink-Verzeichnisse werden abgelehnt.

K3 (kein Finding): Die neuen sudo-Hilfskommandos erhalten Pfade aus der rkhunter-Ausgabe, laufen aber alle ueber run_cmd() in Listenform ohne shell=True — keine Command- oder Argument-Injection.

K4 (kein Finding): backup/restore_property_database() nutzen sudo cp -a, beschraenken die Pfade jedoch per _normalize_allowed_path(allow_prop_db=True) fail-closed auf zwei fest kodierte Realpaths; vor der Wiederherstellung wird die Existenz des NetMon-Backups verlangt. Keine Pfad-Traversal.

Alle 59 relevanten automatisierten Tests bleiben nach den Haertungen gruen. Der Codestand v1.1.18 enthaelt keine offenen ausnutzbaren Sicherheitsluecken; die in Runde 6 gehaerteten Bereiche (Secure Browser Manager, Agent-Confinement) waren in diesem Arbeitsstand unveraendert.

2. Methodik

Phase 1 - Diff- & Reconnaissance

Vollstaendiger git diff gegen den letzten Commit (c2087b63) plus Sichtung der neuen unversionierten Dateien (rkhunter_analysis_service.py und zugehoerige Tests); gezieltes Lesen der neuen RKHunter-Pfade und der ClamAV-Datenbankwahl.

Phase 2 - Kandidaten-Analyse

Ein dedizierter Analyse-Sub-Agent verfolgte Datenflussketten von rkhunter-/dpkg-Ausgaben und lokalen Eingaben bis zu sensiblen Operationen (sudo cp -a, subprocess.run, --propupd, clamscan --database). 4

Kandidaten identifiziert.

Phase 3 - False-Positive-Validierung

Fuer jeden Kandidaten lief eine eigene Validierung mit striktem FP-Filter-Regelwerk und Konfidenz-Scoring, ergaenzt um eine empirische Reproduktion des Klassifizierer-Verhaltens am realen Modul. Ergebnis: 2 kein Finding, 2 unter der Schwelle (2-3/10).

Phase 4 - Haertung & Regressionscheck

Die beiden Kandidaten unter der Schwelle wurden als Defense-in-Depth umgesetzt, empirisch am realen Modul gegengeprueft und mit neuen Regressionstests abgesichert. Suiten erneut ausgefuehrt: test_rkhunter_analysis_service.py, test_rkhunter_security_flow.py, test_rkhunter_manager.py, test_clamav_controller.py — 59 Tests, alle gruen.

3. Bewertete Kandidaten

K1 — Zu permissive FP-Klassifizierung im propupd-Gate [Konfidenz 3/10, gehaertet]

`netmon/services/rkhunter_analysis_service.py — _classify_rkhunter_line() / analyze_rkhunter_output(), ausgewertet in _rkhunter_propupd_gate_valid().`

Das Gate entscheidet, ob der aktuelle Dateizustand per rkhunter --propupd als neue vertrauenswuerdige Referenz uebernommen werden darf. _EXPECTED_CONFIG_RE matchte mit dem Teilmuster [^\s:]+\.conf jede Datei mit .conf-Endung (z. B. /etc/cron.d/persist.conf), und _DEVELOPER_ENV_RE matchte bloße Substrings wie 'docker'/'containerd'/'vscode' an beliebiger Stelle. Solche Zeilen wurden 'expected'; empirisch verifiziert fiel das Risiko dann auf 'low' und propupd_allowed wurde True — obwohl die Datei nicht paketverwaltet ist und daher von dpkg -V nicht erfasst wird. Unter der Schwelle, weil das Platzieren einer solchen Datei bereits Root voraussetzt und die Referenzuebernahme manuell, doppelt bestaetigt und sudo-geschuetzt ist — keine Privilegien-Eskalation.

Bewertung: Gehaertet & verifiziert: _EXPECTED_CONFIG_RE wurde auf eine explizite Whitelist bekannter Konfigurationsdateien eingeschraenkt (rkhunter.conf, gdm3/custom.conf, fwupd.conf, systemd/resolved.conf, systemd/timesyncd.conf, mirrors.dat); _DEVELOPER_ENV_RE nutzt jetzt Wortgrenzen und Pfadsegment-Anker. Empirisch gegengeprueft: /etc/cron.d/persist.conf und .../dockerish-malware fallen nun nach 'review', das Gate schliesst (risk=medium, propupd_allowed=False), waehrend /etc/rkhunter.conf, /usr/bin/docker und .vscode/ weiterhin korrekt 'expected' bleiben. Mit Regressionstests abgesichert.

K2 — ClamAV-Signaturdatenbank in vorhersagbarem /tmp-Pfad [Konfidenz 2/10, gehaertet]

`netmon/features/clamav_controller.py — clamav_user_database_dir() / clamav_database_args() / _clamav_database_dir_is_secure().`

Nach einem nutzergetriebenen ClamAV-Update ohne sudo liegen die Signaturen in /netmon-clamav-db- und werden via clamscan --database genutzt. Der Pfad ist vorhersagbar; auf einem gemeinsam genutzten Host koennte ein Fremdnutzer das Verzeichnis theoretisch mit eigenen .cvd/.cld-Dateien vorbelegen und die Signaturbasis beeinflussen. Unter der Schwelle, weil das Zielprofil ein Einzelnutzer-Desktop ist und NetMon dem Verzeichnis erst nach erfolgreichem freshclam-Lauf vertraut.

Bewertung: Gehaertet & verifiziert: Vor jeder Verwendung von clamscan --database prueft _clamav_database_dir_is_secure() das Verzeichnis — es muss dem aktuellen Nutzer gehoeren (st_uid == os.getuid()), darf kein Symlink sein und keine Gruppen-/Welt-Rechte tragen (st_mode & 0o077 == 0). Empirisch gegengeprueft: fremd-owned, gruppen-/welt-zugaengliche, symlink- und fehlende Verzeichnisse werden abgelehnt — nur ein eigenes 0700-Verzeichnis wird als Signaturquelle akzeptiert. Das /tmp-Vorbelegungs-Szenario ist geschlossen; mit Regressionstests abgesichert.

K3 — Neue privilegierte Hilfskommandos im Scan-Worker [kein Finding]

`netmon/features/security_audit_controller.py — rkhunter_scan_worker().`

Der Scan-Worker fuehrt neu sudo -n dpkg -V, dpkg -S , file -b und sudo -n cat /var/log/rkhunter.log aus. Die Pfade stammen aus der rkhunter-Ausgabe (extract_replaced_script_paths / extract_warning_command_paths), also aus extern beeinflussbarem Text. Bewertung: Alle Aufrufe laufen ueber run_cmd(), das subprocess.run() in Listenform ohne shell=True nutzt. Pfade werden als einzelne argv-Elemente an dpkg -S / file -b uebergeben — keine Shell-Interpolation, keine Option-Injection. Kein Command- oder Argument-Injection-Pfad.

Bewertung: Kein Finding. Die neuen Hilfskommandos sind gegen Shell- und Argument-Injection abgesichert.

K4 — cp -a auf die RKHunter-Referenzdatenbank [kein Finding]

`netmon/core/rkhunter_manager.py — backup_property_database() / restore_property_database().`

Beide Funktionen kopieren die rkhunter-Referenz-DB per sudo -n cp -a. Beide Pfade laufen zuerst durch _normalize_allowed_path(allow_prop_db=True), das os.realpath() anwendet und ausschließlich die zwei fest kodierten Zielpfade (rkhunter.dat und ....netmon.bak) zulaesst; alles andere liefert einen leeren String und die Operation bricht mit 'Nicht erlaubter rkhunter-Datenbankpfad' ab. Vor der Wiederherstellung wird zudem die Existenz des NetMon-Backups (test -f) verlangt. Fail-closed, keine Pfad-Traversal.

Bewertung: Kein Finding. Die Referenz-Sicherung ist fail-closed auf eine feste Pfad-Allowlist begrenzt.

4. Umgesetzte Defense-in-Depth-Haertungen (Runde 7)

Konservativeres propupd-Gate — rkhunter_analysis_service.py

_EXPECTED_CONFIG_RE nutzt statt der .conf-Wildcard eine explizite Whitelist bekannter Konfigurationsdateien; _DEVELOPER_ENV_RE ist mit Wortgrenzen und Pfadsegment-Ankern versehen. Eine nicht paketverwaltete /etc-Aenderung faellt jetzt nach 'review' — das Gate bleibt fail-closed (propupd_allowed=False). Empirisch gegengeprueft, legitime Faelle bleiben 'expected'.

Owner-/Rechte-Pruefung der ClamAV-Signaturdatenbank — clamav_controller.py

_clamav_database_dir_is_secure() laesst clamscan --database nur auf ein Verzeichnis zeigen, das dem aktuellen Nutzer gehoert, kein Symlink ist und keine Gruppen-/Welt-Rechte traegt (& 0o077 == 0). Das /tmp-Vorbelegungs-Szenario ist damit geschlossen.

Neue Regressionstests — 59 / 59 gruen

test_rkhunter_analysis_service.py und test_clamav_controller.py wurden um Tests fuer die verschaerfte Klassifizierung bzw. die Verzeichnis-Sicherheitspruefung erweitert. Alle relevanten Suiten bestehen nach den Haertungen vollstaendig.

5. Geprüfte Sicherheitskategorien

Kategorie	Ergebnis
Command Injection (subprocess, shell=True)	**Kein Finding**
Argument Injection (Pfade aus rkhunter-Ausgabe an dpkg/file)	**Kein Finding (K3, list-form)**
Path Traversal (cp -a der Referenz-DB)	**Kein Finding (K4, fail-closed)**
Sicherheitskontroll-Umgehung (propupd-Gate zu permissiv)	**K1 -> gehaertet (Whitelist)**
Signatur-/Datenbank-Vertrauen (ClamAV --database)	**K2 -> gehaertet (Owner/Rechte)**
Privilegierte Kommandos (sudo -n Umfang & Argumente)	**Kein Finding**
TOCTOU / Symlink bei Verzeichniserstellung	**Kein Finding (_safe_make dirs_no_symlink)**
Unsichere Deserialisierung (pickle, yaml.load, eval)	**Kein Finding**
Persistierte Konfiguration (Schnellscan-Pfade in Settings)	**Kein Finding (normalisiert/validiert)**
Hardcoded Credentials / API-Keys	**Kein Finding**
Sensitive Data Exposure (PII in Logs)	**Kein Finding**

Impressum und Haftungsausschluss

Auditor: Claude Fable 5 (Anthropic) — Analyse-Sub-Agent + False-Positive-Validierung mit empirischer Reproduktion am realen Modul

Projektverantwortlicher: Sven Froehlich

Datum: 17. Juli 2026

Codestand: c2087b63 + Arbeitsstand (modified/untracked), Fokus RKHunter-Referenzuebernahme, deterministische Analyse & ClamAV-Datenbankwahl — Runde 7

Methode: Statische Code-Analyse + manuelle Datenfluss-Verifikation + empirische Reproduktion des Klassifizierer-Verhaltens + Konfidenz-gescorte FP-Validierung + Umsetzung der beiden Haertungen mit empirischer Gegenpruefung + Regressionstests (59/59 relevante Tests bestanden) — kein Laufzeit-Penetrationstest