SICHERHEITS-AUDIT — RUNDE 5

NetMon Roots v1.1.16

Code Security Review — Juli 2026

Datum	05. Juli 2026
Analysierte Version	v1.1.16 — Runde 5
Geaenderte/neue Dateien	~13
Sicherheitsfokus-Module	12
Offene Findings	0
Bewertete Kandidaten	3 (alle behoben)
Konfidenz-Schwelle	>= 7 / 10
Methode	Sub-Agent-Review + Fix-Verifikation per Exploit-Repro
Analysewerkzeug	Claude Sonnet 5 (Anthropic)
Projektverantwortlicher	Sven Froehlich

3 FINDINGS GEFUNDEN UND NOCH IN DIESER RUNDE BEHOBEN — 0 OFFEN

1. Zusammenfassung

Die fuenfte Sicherheitsanalyse konzentrierte sich ausschliesslich auf das neue Secure-Browser-Manager-Feature (Commit 9baa6284 + aktueller Arbeitsstand) sowie kleinere Diffs seit Runde 4. Ein dedizierter Sub-Agent identifizierte 3 bestaetigte Schwachstellen (kein False Positive) im brandneuen Feature, alle mit Konfidenz 7-8/10.

K1 (Medium, behoben): Fehlende Zeichensatzvalidierung fuer Container-Namen erlaubte Pfad-Escapes aus dem Container-Basisverzeichnis heraus, u.a. in ~/.config/autostart (Persistenz-Risiko).

K2 (Medium, behoben): homepage_url/new_tab_url wurden unescaped in Firefox user.js eingesetzt -> Preference-Injection, die die eigenen Isolations-Haertungen des Containers untergraben konnte.

K3 (Low, behoben): Fehlende Newline-Bereinigung in .desktop-Feldern (Name=/Comment=) konnte zusaetzliche Desktop-Entry-Schluessel einschleusen.

Alle drei Fixes wurden per Reproduktionsskript verifiziert (Exploit vor Fix erfolgreich, nach Fix abgewiesen/entschaerft) und gegen die bestehenden 121 automatisierten Tests regressionsgetestet - alle bestanden.

2. Methodik

Phase 1 - Reconnaissance

Vollstaendiges Lesen von netmon/modules/secure_browser/*.py (manager/service/ui/models), secure_browser_tab.py und browser_container_builder.py; Abgleich gegen die im README dokumentierten Sicherheitsgarantien.

Phase 2 - Kandidaten-Analyse

Dedizierter Sub-Agent verfolgte Datenflussketten von Nutzereingaben (Containername, Homepage-URL, containers.json) bis zu Dateisystem-Operationen. 4 Kandidaten mit Konfidenz 7-8/10 identifiziert, 2 mit identischer Ursache zusammengefasst zu K1.

Phase 3 - Fix & Exploit-Verifikation

Alle 3 Kandidaten direkt am Quelltext bestaetigt. Patches implementiert: CONTAINER_NAME_PATTERN-Validierung, json.dumps()-Escaping in build_user_js_content(), _desktop_field()-Sanitizer. Mit Python-Repro-Skript verifiziert.

Phase 4 - Regressionscheck

tests/test_browser_container_builder.py (54 Tests) und tests/test_secure_browser_module.py (67 Tests) erneut ausgefuehrt - alle 121 weiterhin gruen nach den Fixes.

3. Bestaetigte Findings (alle in dieser Runde behoben)

K1 — Fehlende Namensvalidierung: Pfad-Escape beim Anlegen von Browser-Containern [Medium]

`tools/secure_browser_manager/browser_container_builder.py — validate_config_payload(); betrifft transitiv netmon/modules/secure_browser/manager.py.`

Das Feld 'name' wurde nur auf 'nicht leer' geprueft. Zielverzeichnis (base_dir / name) und .desktop-Dateiname wurden direkt daraus abgeleitet, ohne Zeichensatz- oder Confinement-Pruefung. Ein Name wie '../../../../.config/autostart/update-check' verschiebt den erzeugten .desktop-Eintrag in den Autostart-Ordner (Persistenz beim naechsten Login).

Fix: CONTAINER_NAME_PATTERN = re.compile(r'^[A-Za-z0-9_.-]+$') in validate_config_payload() ergaenzt. Da CLI-Tool und NetMon-UI ueber denselben load_config()-Pfad laufen, schliesst ein Fixpunkt beide Einstiegspfade. Verifiziert: Escape-Name abgelehnt, regulaerer Name funktioniert weiter, 54 Tests gruen.

K2 — Firefox user.js-Praeferenz-Injection via Homepage-/Neue-Tab-URL [Medium]

`tools/secure_browser_manager/browser_container_builder.py — build_user_js_content().`

homepage_url/new_tab_url wurden per f-string in user_pref("...", "{value}"); eingesetzt, ohne Anfuehrungszeichen zu escapen. Ein Wert mit eingebettetem " beendet den String vorzeitig und haengt beliebige weitere user_pref(...)-Aufrufe an — potenziell die eigene Isolations-Haertung des Containers aushebelnd (z. B. https_only_mode, cookieBehavior).

Fix: Beide Werte werden jetzt mit json.dumps() kodiert. Verifiziert: derselbe Angriffswert erzeugt jetzt eine einzelne, harmlose user_pref-Zeile ohne injizierte Folgeanweisung.

K3 — Fehlende Zeilenumbruch-Bereinigung in .desktop-Launcher-Feldern [Low]

`tools/secure_browser_manager/browser_container_builder.py — build_launcher_content(), build_existing_launcher_content(), build_menu_launcher_content().`

Name=/Comment=-Zeilen wurden direkt aus spec.name/menu_name gebildet, ohne eingebettete Zeilenumbrueche zu entfernen — potenziell zusaetzliche Desktop-Entry-Schluessel injizierbar (parserabhaengig auch ein zweiter Exec=-Eintrag).

Fix: Neue Hilfsfunktion _desktop_field() entfernt \r/\n aus name/menu_name vor Einsatz in Name=/Comment= — an allen drei Launcher-Erzeugungsstellen angewendet.

4. Sicherheitsverbesserungen seit Runde 4

Agent-Runner-Allowlist verkleinert — agent/runner.py

'python' und 'cat' aus ALLOWED_BASE_COMMANDS entfernt. Neu: ALLOWED_GIT_SUBCOMMANDS = {status, diff, log, show} begrenzt git auf Lesebefehle; _validate_repo_confined_arguments() erzwingt fuer black/ruff/pytest/ls, dass Pfadargumente innerhalb von REPO_ROOT liegen.

Repo-Confinement fuer Datei-Tools — agent/tools.py

Neue Hilfsfunktion _ensure_repo_path() prueft in list_files() und read_file(), dass der aufgeloeste Pfad via relative_to(REPO_ROOT) innerhalb des Repositories bleibt.

sys.path-Bereinigung — tools/netmon.py

Alle Eintraege, die auf NETMON_REPO_ROOT aufloesen, werden vor dem Neueinfuegen an Position 0 entfernt. Verhindert, dass ein veralteter Eintrag Vorrang vor dem echten Repository-Root bekommt.

Transparenz: eigene Listener in der Firewall-Ansicht erkennbar

app.py + network_helpers.py: ss -lntupeH (cgroup-Ausgabe) ergaenzt; _is_netmon_listener()/_is_netmon_app_scope() erkennen NetMon-eigene Ports anhand Prozessname/PID/systemd-Scope.

Versions-Bump auf v1.1.16

netmon/config/constants.py: NETMON_APP_VERSION von 1.1.15 auf 1.1.16 angehoben, passend zum in dieser Runde veroeffentlichten Codestand.

5. Offene Hardening-Empfehlung (Runde 6)

Client-seitige Namensvalidierung im Secure-Browser-Dialog

Die Durchsetzung von K1 sitzt aktuell ausschliesslich in browser_container_builder.py (Subprocess-Ebene) — funktional korrekt und ausreichend (fail-closed), aber ein ungueltiger Name fuehrt im NetMon-UI-Dialog erst nach dem Subprocess-Aufruf zu einer Fehlermeldung statt zu sofortigem Inline-Feedback. Empfehlung: dieselbe CONTAINER_NAME_PATTERN-Pruefung zusaetzlich in manager.py::_resolve_create_inputs() spiegeln — reine UX-Verbesserung, keine sicherheitskritische Luecke.

6. Gepruefte Sicherheitskategorien

Kategorie	Ergebnis
Command Injection (subprocess, shell=True)	**Kein Finding**
Path Traversal (Dateioperationen mit Nutzereingabe)	**K1 -> behoben (Secure Browser Manager)**
Privilege Escalation (sudo mit unkontrollierten Argumenten)	**Kein Finding**
Argument Injection (list-form subprocess mit externen Daten)	**Kein Finding**
Unsichere Deserialisierung (pickle, yaml.load, eval)	**Kein Finding**
Firefox user.js/Preference-Injection	**K2 -> behoben (Secure Browser Manager)**
Desktop-Entry-Injection (.desktop Name=/Comment=)	**K3 -> behoben (Secure Browser Manager)**
Hardcoded Credentials / API-Keys	**Kein Finding**
Sensitive Data Exposure (PII in Logs)	**Kein Finding**
SSRF via urllib.request / externe URL-Weitergabe	**Kein Finding**
Kryptographische Schwachstellen	**Kein Finding**
Symlink-Angriffe auf privilegierte Dateioperationen	**Kein Finding**
Agent-Runner Command-/Pfad-Allowlist	**Gehaertet (agent/runner.py, agent/tools.py)**
WireGuard-Hook-Ausfuehrung via Import (R4-K1)	Geschlossen (R4, FP)
PATH-Hijacking via shutil.which (R3-K1)	Geschlossen (R3)

7. Analysierte Module (Runde 5 — Sicherheitsfokus)

Datei / Modul	Fokus	Ergebnis
`netmon/modules/secure_browser/manager.py`	Containername-/Pfad-Ableitung, Apply-Freigabe mit Bestaetigungscode	K1 -> behoben
`netmon/modules/secure_browser/service.py`	Subprocess-Delegation (List-Form, shell=False), Backup-Rendering	Kein Finding

`netmon/modules/secure_browser/ui.py + secure_browser_tab.py`	Tkinter-Formulare, Eingabe-Weiterleitung an manager.py	Kein Finding
`tools/secure_browser_manager/browser_container_builder.py`	Namensvalidierung, Zielpfad-Aufbau, user.js/.desktop-Erzeugung	K1+K2+K3 -> behoben
`tools/secure_browser_manager/containers.example.json`	Konfig-Schema	Kein Finding
`agent/runner.py`	Allowlist verkleinert, git-Subcommand-Allowlist, Repo-Confinement	Kein Finding (Haertung)
`agent/tools.py`	_ensure_repo_path()-Guard fuer list_files/read_file	Kein Finding (Haertung)
`netmon/features/vpn_controller.py`	Preshared-Key-Feld, Validierung via validate_wireguard_key_value()	Kein Finding
`netmon/app.py + network_helpers.py`	cgroup-basierte Selbst-Erkennung eigener Firewall-Listener	Kein Finding
`netmon/ui/tabs/log_tab.py`	Auto-Clear-Log-Checkbox, nutzt bestehende Einstellung	Kein Finding
`tools/netmon.py`	sys.path-Bereinigung vor Neueintrag	Kein Finding (Haertung)
`netmon/app.py (Commit 9d660ae3)`	Runtime-Log-Persistenz, Pfade rein intern (netmon_state_dir)	Kein Finding

---

Impressum und Haftungsausschluss

Auditor: Claude Sonnet 5 (Anthropic) — dedizierter Sub-Agent-Review + manuelle Fix-Verifikation per Exploit-Reproduktionsskript

Projektverantwortlicher: Sven Froehlich

Datum: 05. Juli 2026

Codestand: 920cfc9b + Arbeitsstand (Secure Browser Manager, modified/untracked), ~13 Dateien — Runde 5

Methode: Statische Code-Analyse + manuelle Datenfluss-Verifikation + Regressionstests (121/121 bestanden) — kein Laufzeit-Penetrationstest