SICHERHEITS-AUDIT — RUNDE 6

NetMon Roots v1.1.16

Code Security Review — Juli 2026

Datum	07. Juli 2026
Analysierte Version	v1.1.16 — Runde 6
Neue Angriffsflaeche	Favicon-Download (Netzwerk)
Bewertete Kandidaten	4 (alle unter Schwelle)
Ausnutzbare Findings	0
Haertungen umgesetzt	3 (Defense-in-Depth)
Konfidenz-Schwelle	>= 8 / 10
Regressionstests	149 / 149 gruen (+6 neu)
Analysewerkzeug	Claude Fable 5 (Anthropic)
Projektverantwortlicher	Sven Froehlich

0 AUSNUTZBARE FINDINGS — 3 DEFENSE-IN-DEPTH-HAERTUNGEN PROAKTIV UMGESETZT

1. Zusammenfassung

Die sechste Sicherheitsanalyse konzentrierte sich auf die seit dem Runde-5-Snapshot neu hinzugekommene bzw. veraenderte Angriffsflaeche im Arbeitsstand — vor allem die erstmals eingefuehrte Favicon-Download-Netzwerkfunktion des Secure-Browser-Managers (die erste ausgehende Netzwerk-Schnittstelle des Features) sowie die Pfad-Confinement-Haertung des Coding-Agenten (agent/runner.py) sowie das damals noch vorhandene GUI-/Docker-Dev-Container-Profil (inzwischen aus dem Setup entfernt).

Ein Analyse-Sub-Agent identifizierte vier Kandidaten. Jeder wurde von einem eigenen, parallel laufenden Validierungs-Sub-Agenten mit striktem False-Positive-Filter und Konfidenz-Scoring geprueft — und allesamt unterhalb der Schwelle von 8/10 eingestuft (2-3/10). Grund: In einem Einzelnutzer-Desktop-Werkzeug stammt die vermeintliche 'Angreifer'-Eingabe (selbst geschriebene containers.json, lokal getippte Agent-Kommandos) vom selben Nutzer, der das Werkzeug ausfuehrt — es wird keine echte Vertrauensgrenze ueberschritten.

K1 (Konfidenz 3/10, gehaertet): fetch_url() reichte URLs ohne Scheme-Pruefung an urllib.urlopen() — file:// (lokaler Dateizugriff) und beliebige Hosts waren erreichbar. Jetzt auf http/https beschraenkt.

K2 (Konfidenz 3/10, gehaertet): Das Repo-Pfad-Confinement des Agenten uebersprang --flag=value-Argumente (z. B. --junitxml=/etc/passwd). Der Wert-Teil solcher Flags wird jetzt mitgeprueft.

K3 (Konfidenz 2/10, gehaertet): Die Containername-Regex matcht auch '.'/'..'. In der Praxis durch mkdir(exist_ok=False) blockiert, jetzt zusaetzlich explizit fail-closed abgelehnt.

K4 (Konfidenz 3/10, Empfehlung offen): Loser Substring-Match "netmon" in process_name bei der Listener-Zuordnung — rein kosmetisch in einem Diagnose-Report, bewusst nicht geaendert (Aenderung wuerde echte NetMon-Listener faelschlich als 'fremd' markieren koennen). Als Empfehlung fuer Runde 7 dokumentiert.

Drei der vier Punkte wurden als Defense-in-Depth direkt umgesetzt und mit 6 neuen Regressionstests abgesichert. Alle 149 relevanten automatisierten Tests bleiben nach den Haertungen gruen. Die drei in Runde 5 behobenen Findings (K1-K3) wurden am aktuellen Code gegengeprueft und sind unveraendert wirksam.

2. Methodik

Phase 1 - Diff- & Reconnaissance

Vollstaendiger git diff gegen den letzten Commit plus Sichtung aller neuen unversionierten Dateien; gezieltes Lesen der neuen Netzwerkpfade in browser_container_builder.py und der Haertungslogik in agent/runner.py.

Phase 2 - Kandidaten-Analyse

Ein dedizierter Analyse-Sub-Agent verfolgte Datenflussketten von Nutzer-/Konfig-Eingaben (Containername, homepage_url, Agent-Kommandozeile) bis zu sensiblen Operationen (urlopen, mkdir/copytree, subprocess.run). 4 Kandidaten identifiziert.

Phase 3 - False-Positive-Validierung

Fuer jeden Kandidaten lief ein eigener, paralleler Validierungs-Sub-Agent mit striktem FP-Filter-Regelwerk und Konfidenz-Scoring. Ergebnis: alle vier unter der Schwelle (2-3/10) — kein Angriffspfad ueber eine echte Vertrauensgrenze.

Phase 4 - Haertung & Regressionscheck

3 der 4 Punkte als Defense-in-Depth umgesetzt und mit 6 neuen Regressionstests abgesichert. Bestehende Suiten erneut ausgefuehrt: test_runner.py, test_secure_browser_module.py und test_browser_container_builder.py — 149 Tests, alle gruen.

3. Bewertete Kandidaten (alle unter der Konfidenzschwelle)

K1 — Ausgehende Netzwerk-Fetches ohne Scheme-Allowlist [Konfidenz 3/10, gehaertet]

`tools/secure_browser_manager/browser_container_builder.py — fetch_url(), aufgerufen aus resolve_favicon_download() (Standard-Icon-Modus 'favicon').`

fetch_url() reichte homepage_url (und daraus per urljoin() abgeleitete Icon-Links) ohne Scheme-Pruefung an urllib.request.urlopen(). Der Default-Opener behandelt neben http/https auch file:// — ein homepage_url wie file:///etc/passwd haette einen lokalen Dateizugriff ausgeloest, ein beliebiger Host waere als SSRF-Primitive erreichbar gewesen. Unter der Schwelle, weil homepage_url nur vom selben lokalen Nutzer gesetzt wird (kein Remote-Eingabepfad) — Self-SSRF ohne Vertrauensgrenze.

Massnahme: fetch_url() prueft jetzt das Scheme und lehnt alles ab, was nicht http/https ist. Neuer Test test_fetch_url_rejects_non_http_schemes (file://, ftp://, gopher://, absoluter Pfad) verifiziert die Ablehnung.

K2 — --flag=value-Luecke im Pfad-Confinement des Agenten [Konfidenz 3/10, gehaertet]

`agent/runner.py — _looks_like_path_argument() / _validate_repo_confined_arguments().`

Die Repo-Confinement-Pruefung behandelte jedes mit '-' beginnende Argument sofort als 'kein Pfad' und uebersprang es. Flags wie --junitxml=/pfad, --basetemp=/pfad oder --output-file=/pfad (pytest/ruff) trugen ihren Pfad am Check vorbei. Unter der Schwelle, weil der einzige Aufrufer (agent/cli.py) das Kommando aus sys.argv baut — ein manuell vom Betreiber gestartetes lokales CLI, keine an untrusted Input angebundene LLM-Schleife.

Massnahme: _looks_like_path_argument() zerlegt jetzt --flag=value-Argumente und prueft den Wertteil; neue Hilfsfunktion _flag_value() sorgt dafuer, dass auch _is_repo_confined_path() den reinen Pfad bewertet. Drei neue Tests decken innerhalb/ausserhalb Repo ab.

K3 — '.'/'..' passieren die Containername-Validierung [Konfidenz 2/10, gehaertet]

`browser_container_builder.py (validate_config_payload()) und netmon/modules/secure_browser/manager.py (_resolve_create_inputs()).`

Das Runde-5-Muster ^[A-Za-z0-9_.-]+$ erlaubt den Punkt und matcht damit auch '.' und '..'. base_dir / name zeigt fuer '..' lexikalisch auf das uebergeordnete Verzeichnis. Unter der Schwelle, weil in der Praxis mkdir(parents=True, exist_ok=False) den Schreibvorgang blockiert (empirisch verifiziert: FileExistsError vor copytree) — kein Schreibzugriff ausserhalb base_dir heute moeglich.

Massnahme: An beiden Validierungsstellen wird name in {'.', '..'} jetzt explizit abgelehnt — die Namenspruefung ist selbst fail-closed und haengt nicht mehr an der mkdir-Semantik. Zwei neue Tests sichern das ab.

K4 — Loser Substring-Match bei der NetMon-Listener-Zuordnung [Konfidenz 3/10, Empfehlung offen]

`netmon/app.py — _is_netmon_listener().`

Die Firewall-Transparenz stuft einen Listener u. a. per 'if "netmon" in process_name' als eigenen ein. Ein Prozess namens netmon-helper wuerde so gelabelt. Unter der Schwelle und bewusst nicht geaendert: die Klassifikation ist rein anzeigend (steuert keine ufw-Regel, blendet keine Zeile aus der Haupttabelle aus); ein Angreifer, der Prozesse frei benennen kann, hat bereits lokale Codeausfuehrung. Ein Entfernen des Matches koennte echte NetMon-Listener faelschlich als 'fremd' markieren.

Massnahme: Empfehlung fuer Runde 7: Zuordnung nur ueber schwer faelschbare Signale (pid == os.getpid() und _is_netmon_app_scope() / app-gnome-netmon-*.scope) stuetzen und den Substring-Match entfernen, sobald die Selbst-Erkennung darueber verlaesslich abgedeckt ist.

4. Umgesetzte Haertungsmassnahmen (Runde 6)

Scheme-Allowlist fuer Netzwerk-Fetches — browser_container_builder.py

fetch_url() akzeptiert nur noch http/https. file:// (lokaler Dateizugriff via urllib-Default-Opener) und andere Nicht-Web-Schemes sind ausgeschlossen, bevor die URL urlopen() erreicht.

Vollstaendiges Pfad-Confinement inkl. --flag=value — agent/runner.py

_looks_like_path_argument() zerlegt jetzt --flag=value-Argumente und prueft den Wertteil; neue Hilfsfunktion _flag_value() stellt sicher, dass auch _is_repo_confined_path() den reinen Pfad bewertet. --junitxml=/etc/passwd wird nun abgewiesen.

Fail-closed Containername-Validierung — browser_container_builder.py + manager.py

'.' und '..' werden an beiden Validierungseinstiegen explizit abgelehnt — die Namenspruefung haengt nicht mehr an der mkdir(exist_ok=False)-Semantik als letzter Schutzlinie.

6 neue Regressionstests — 149/149 gruen

test_runner.py (+3: --flag=value innerhalb/ausserhalb Repo), test_browser_container_builder.py (+3: '.'/'..'-Namen, Pfad-Escape-Name, Nicht-HTTP-Schemes). Alle drei relevanten Suiten bestehen nach den Fixes vollstaendig.

5. Runde-5-Fixes — am aktuellen Code gegengeprueft

Runde-5-Fix	Status im Codestand v1.1.16
K1 — CONTAINER_NAME_PATTERN (Namensvalidierung)	Vorhanden (builder:35, manager:54), in R6 um '.'/'..'-Ablehnung erweitert
K2 — json.dumps()-Escaping in user.js	Wirksam (builder:1312-1313) — Preference-Injection geschlossen
K3 — _desktop_field() gegen \r/\n	Wirksam (builder:1463 ff.), an allen drei Launcher-Stellen

6. Geprüfte Sicherheitskategorien

Kategorie	Ergebnis
SSRF / ausgehende URL-Fetches (Host-/Scheme-Kontrolle)	**K1 -> gehaertet (Scheme-Allowlist)**
Local File Read via file:// im URL-Opener	**K1 -> gehaertet**
Path Traversal (Dateioperationen mit Nutzereingabe)	**K3 -> gehaertet (fail-closed)**
Agent-Sandbox: Pfad-Confinement der Kommando-Allowlist	**K2 -> gehaertet (--flag=value)**
Command Injection (subprocess, shell=True)	**Kein Finding**
Argument Injection (list-form subprocess mit externen Daten)	**Kein Finding**
Firefox user.js / Preference-Injection	**Geschlossen (R5-K2)**
Desktop-Entry-Injection (.desktop Name=/Comment=)	**Geschlossen (R5-K3)**
Unsichere Deserialisierung (pickle, yaml.load, eval)	**Kein Finding**
Privilege Escalation (historisches Dev-Container docker.sock-Mount)	**Historisch / entfernt aus dem aktuellen Setup**
Sicherheitskontroll-Umgehung (Listener-Zuordnung)	**K4 -> Empfehlung (offen)**
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 + vier parallele False-Positive-Validierungs-Sub-Agenten + manuelle Fix-Umsetzung und Regressionstests

Projektverantwortlicher: Sven Froehlich

Datum: 07. Juli 2026

Codestand: 9baa6284 + Arbeitsstand (modified/untracked), Fokus Favicon-Download & Agent-Pfad-Confinement — Runde 6

Methode: Statische Code-Analyse + manuelle Datenfluss-Verifikation + Konfidenz-gescorte FP-Validierung + Regressionstests (149/149 relevante Tests bestanden) — kein Laufzeit-Penetrationstest