NetMon Dokumentation

Version: 1.1.19

Kompakte Zusammenstellung der zentralen Projekt- und Nutzerdokumentation.

Inhaltsverzeichnis

CONFIGURATION

SECURITY

ARCHITECTURE

DEB PACKAGING PLAN

CHANGELOG

2026-07-17 - Version 1.1.18

2026-07-11 - Version 1.1.17

2026-07-07 - Version 1.1.16

2026-07-05 - Version 1.1.16

2026-06-29 - Version 1.1.15

2026-06-28 - Version 1.1.14

2026-06-21 - Version 1.1.13

2026-06-14 - Version 1.1.10

README

Quelle: docs/README.md

NetMon

Dokumentierter Paket- und App-Stand: 1.1.19.

NetMon ist eine Linux-GUI fuer Netzwerk-, DNS-, Firewall-, VPN-, Security-, Radio- und Hilfsfunktionen. Der Startpfad liegt in netmon/main.py, die Hauptoberflaeche in netmon/app.py, und die Funktionslogik ist in Controller-, Service- und Tab-Module aufgeteilt.

Kurzbeschreibung

NetMon kombiniert klassische Netzwerk-Werkzeuge, DNS-/DoT-Steuerung, Firewall- und Sicherheitspruefungen, VPN-Slots, Radio, Theme Studio, Linux-Hilfe und Chad-Analyse in einer Tk-/ttk-Oberflaeche. Der Code verwendet fuer die GUI tkinter, fuer den Start einen Splash-/Lock-/Crash-Flow und fuer viele Teilbereiche klar getrennte Controller- und Service-Module.

Hauptfunktionen

- GUI-Start ueber python -m netmon mit optionalen Modi --headless, --headless-smoke und --theme.
- Haupttabs fuer Log, Trace, Verbindungen, DNS, Chad, Download-Scan, Einstellungen sowie verschachtelte Bereiche System, Tools und Security.
- DNS-Funktionen inklusive DNS Guard, DNS-over-TLS-Profilen und Resolver-Pruefungen.
- Firewall-, UFW-, Killswitch- und Packet-Capture-Funktionen.
- Security-Monitoring mit Security-Events, Startup Guard, Process Guard, Repository Guard, Roots Baseline Guard, ClamAV, Lynis und RKHunter.
- ClamAV-Schnellscan mit dauerhaft konfigurierbaren Standard- und eigenen Ordnern, sichtbarer Laufzeit, Abbruchfunktion sowie Fund- und Quarantaeneverwaltung.
- RKHunter-Auswertung mit deterministischer Risikoklassifizierung, ergaenzender dpkg -V-Pruefung und streng geschuetzter manueller Referenzuebernahme.
- Radio-Modul mit eingebetteter und losgeloester Radio-App, Datenbank-/Favoriten-/Settings-Dateien und optionalem projectM-Visualizer.
- Theme Studio fuer das Pop-Testprofil inklusive Speichern/Laden von Profilen.
- Linux-Hilfe mit Such-/Glossaransicht, Prompt-Kopie fuer externe PDF-Erstellung und importierbarer PDF-Bibliothek.
- Cleanup Assistant, HDD-/Link-/VPN-Slot-Verwaltung und Chad-Analyse.
- Secure Browser fuer getrennte, gehaertete Firefox-Instanzen pro Webdienst mit Sicherheitsprofilen, Domain-Whitelist, Dry-Run und verwalteten Cookie-/Login-Optionen.
- Gefuehrte Touren im Setup, einschliesslich einer zweisprachigen Secure-Browser-Tour mit ausfuehrlicher Datenschutz- und Grenzen-Erklaerung.
- VPN-Profilverwaltung mit konsistenter Slot-/Privacy-Bereinigung und automatischer Aktualisierung der oeffentlichen IP nach einem Profilwechsel.

- About-Dialog mit Online-Versionspruefung gegen netmonroots.com und automatischer Weiterleitung auf die Downloadseite bei neueren Website-Releases.
- Support-/Diagnosebereich mit lokalem Supportpaket-Export und sichtbarem Datenschutzhinweis vor dem Versand.
- Paketierter Deinstaller mit optionaler Bereinigung von NetMon-, WireGuard-, NetworkManager- und Testsystem-Resten.
- Aktueller Security-Audit-Stand fuer die Produktdoku: Runde 7 auf Basis von 1.1.19.

Screenshots

Screenshot-Platzhalter: Hauptfenster mit Haupttabs.
Screenshot-Platzhalter: Radio-Tab mit aktuellem Sender, Steuerung & EQ und projectM.
Screenshot-Platzhalter: Security-Bereich mit Firewall-/Monitoring-Ansicht.

Installation

Im Repository deklarierte Python-Abhaengigkeiten

- requirements.txt: psutil, PySide6
- pyproject.toml: requires-python >=3.11

Im Startpfad explizit vorausgesetzt oder optional importiert

- tkinter wird in netmon/main.py fuer die GUI vorausgesetzt.
- Pillow (PIL) wird im Start-Splash optional verwendet.
- Pillow.ImageTk wird fuer Bild-/Log-Overlay-Pfade benoetigt; unter Debian/Ubuntu ist dafuer typischerweise python3-pil.imagetk erforderlich.
- requests wird im Online-AI-Service optional verwendet; ohne requests faellt der Code auf urllib zurueck.

Debian-Paketstand

- Aktueller Test-/Paketstand in diesem Repository: 1.1.19
- Build-Skript: scripts/build_netmon_deb.sh
- Ergebnisdatei: dist/netmon_<version>_all.deb
- Harte Laufzeitabhaengigkeiten im Paket: udisks2, cryptsetup
- Paket liefert derzeit unter anderem:
- netmon
- netmon-uninstall
- netmon.desktop
- netmon-uninstall.desktop
- netmon.png
- netmon-radio.png als Radio-spezifisches Asset
- Secure-Browser-Manager und dessen Builder-/Policy-Ressourcen

- aktuelle HTML-/PDF-Dokumentation fuer Version 1.1.19

Beispiel-Setup unter Linux

```
python3 -m venv .venv
./.venv/bin/pip install -r requirements.txt
./.venv/bin/python -m netmon --headless
./.venv/bin/python -m netmon
```

Wenn tkinter fehlt, meldet netmon/main.py explizit den Debian/Ubuntu-Hinweis sudo apt install -y python3-tk.

Erste Schritte

1. Optional im Headless-Modus pruefen: ./.venv/bin/python -m netmon --headless
2. GUI starten: ./.venv/bin/python -m netmon
3. Im Bereich Settings / Theme Studio Theme und Sprache pruefen.
4. DNS-, Firewall- und Security-Bereiche nur mit passender Linux-Umgebung und ggf. sudo nutzen.
5. Weitere Details: USER_GUIDE.md, DEVELOPER_GUIDE.md, CONFIGURATION.md

Externes Testlabor

- Fuer ein externes Linux-Testlabor auf /media/happy/Work2/netmon-lab siehe LAB_WORKFLOW.md.
- Ein neuer Run-Ordner laesst sich mit scripts/lab/init_test_run.sh vorbereiten.

Lizenz

NetMon verweist im aktuellen Stand auf GPL-3.0-or-later. Relevante Dateien:

- LICENSE
- THIRD_PARTY_NOTICES.md
- AppStream-Metadaten unter packaging/netmon.metainfo.xml

Kontakt

Projekt-/Support-Kontakte im aktuellen Stand:

- info@netmonroots.com
- support@netmonroots.com

Changelog

Siehe CHANGELOG.md.

USER GUIDE

Quelle: docs/USER_GUIDE.md

Benutzeranleitung NetMon

Einfuehrung

NetMon ist eine Linux-Anwendung mit grafischer Oberfläche auf Basis von tkinter/ttk. Der Code verbindet Netzwerkwerkzeuge, DNS-/Firewall-Steuerung, Sicherheitspruefungen, Radio, Theme Studio, Linux-Hilfe, Cleanup-Funktionen und Chad-Analyse in einer einzigen Anwendung.

Was ist NetMon?

NetMon ist laut Codebasis eine Desktop-Anwendung fuer Netzwerk- und Security-Aufgaben mit zusaetzlichen Hilfsfunktionen. Der zentrale Einstiegspunkt ist netmon/main.py, die Hauptklasse ist App in netmon/app.py.

Fuer wen ist NetMon gedacht?

- Fuer Linux-Nutzer, die DNS, Firewall, VPN, Security-Events und Netzwerk-Tools ueber eine GUI bedienen wollen.
- Fuer Nutzer, die Logik in Tabs statt in einzelnen Shell-Kommandos nutzen moechten.
- Fuer Entwickler oder Power-User, die zusaetzlich Radio, Theme Studio, Linux-Hilfe und Chad-Analyse verwenden.

Installation

Voraussetzungen

- Python >=3.11 laut pyproject.toml.
- GUI-Unterstuetzung ueber tkinter laut Startpfad in netmon/main.py.
- Python-Pakete aus requirements.txt: psutil, PySide6.
- Optionale Pakete im Code: Pillow fuer Splash/Logo-Verarbeitung, requests fuer Onlinezugriffe, openpyxl fuer Linux-Hilfe-Import.

Installation unter Linux

```
python3 -m venv .venv
./.venv/bin/pip install -r requirements.txt
./.venv/bin/python -m netmon --headless
./.venv/bin/python -m netmon
```

Wenn beim GUI-Start tkinter fehlt, verweist netmon/main.py auf sudo apt install -y python3-tk.

Abhaengigkeiten

- In vielen Bereichen werden externe Linux-Tools nur verwendet, wenn sie vorhanden sind, z. B. ufw, wg-quick, resolvectl, traceroute, mtr, nmap, iftop, htop, tcpdump, wireshark, flatpak oder projectM.
- Der Code prueft fehlende Tools an mehreren Stellen und gibt Hinweise im UI oder Log aus.

Bedienung

Netzwerkueberwachung

- Zweck: Die Tabs Trace, Verbindungen, MTR, Nmap, iftop und htop bilden den Kern fuer Netzwerkbeobachtung und klassische Netzwerktests.
- Beschreibung: Navigation ueber Haupttab Trace/Verbindungen sowie Tools-Untertabs.
- Screenshot-Platzhalter: Screenshot-Platzhalter: Trace-, Verbindungen- und Nmap-Ansicht.
- Typische Anwendung: Typische Nutzung: Hosts pruefen, Hops analysieren, offene Ports scannen, Traffic beobachten.

DNS Guard und DNS-Steuerung

- Zweck: Der DNS-Tab kombiniert DNS-Check, DNS-Profile, DNS Guard und DNS-over-TLS.
- Beschreibung: Die Auswertung steckt im DNSControlService; der App-Code fuehrt Statusvariablen fuer Resolver, Konflikte, DNS Guard und DoT.
- Screenshot-Platzhalter: Screenshot-Platzhalter: DNS-Tab mit DNS Guard und DoT-Profilen.
- Typische Anwendung: Typische Nutzung: Resolver pruefen, Abweichungen erkennen, DoT-Konfiguration setzen.

Firewall und Security-Monitoring

- Zweck: Der Security-Bereich enthaelt Firewall/UFW, Security-Events, Process Guard, Startup Guard, Repository Guard, Roots Baseline Guard, ClamAV, Lynis, RKHunter und Packet Capture.
- Beschreibung: Die Oberflaechen entstehen in netmon/ui/tabs/security_tabs.py; die Logik steckt in separaten Controllern.
- Screenshot-Platzhalter: Screenshot-Platzhalter: Security-Notebook mit Firewall- und Monitoring-Ansicht.
- Typische Anwendung: Typische Nutzung: UFW-Status pruefen, Security-Logs sichten, Baselines erstellen, Paketmitschnitte starten.
- VPN/UFW-Hinweis: Nach einem erfolgreichen VPN-Profilwechsel synchronisiert NetMon die referenzierten Interface- und UDP-Endpoint-Regeln wieder mit UFW, auch wenn der Killswitch aus ist.

Radio

- Zweck: Der Radio-Tab ist ein eigener eingebetteter Player mit Stationen, Favoriten, EQ, Miniplayer und optionalem projectM-Visualizer.
- Beschreibung: Funktionen liegen in netmon/netmon_radio_app.py, netmon/netmon_radio_core.py und netmon/features/radio_controller.py.
- Screenshot-Platzhalter: Screenshot-Platzhalter: Radio-Tab mit aktueller Station und projectM.
- Typische Anwendung: Typische Nutzung: Senderdatenbank laden, Station abspielen, Favoriten pflegen, Player losloesen.
- Standalone-Verhalten: Das losgeloeste externe Radio startet nur noch im Darkmode; dort gibt es keine Theme-Umschaltung mehr.

Chat / Chad

- Zweck: Chad bietet lokale Prompt-Erzeugung und optionale Onlineanalyse.
- Beschreibung: Es gibt zwei Tabs: Analyse und Einstellungen. Onlinezugriffe laufen ueber AiOnlineService, Prompt-Aufbereitung ueber AiAnalysisService.
- Screenshot-Platzhalter: Screenshot-Platzhalter: Chad Analyse und Chad Einstellungen.

- Typische Anwendung: Typische Nutzung: Log-/Security-/DNS-Ausgaben zusammenfassen und erklaeren lassen.

Systeminformationen

- Zweck: Im Bereich System liegen Untertabs fuer allgemeine Systemansicht, HDD, Links und VPN-Slots.
- Beschreibung: Die Builder liegen in netmon/ui/tabs/system_tab.py, hdd_tab.py, links_tab.py und vpn_slots_tab.py.
- Screenshot-Platzhalter: Screenshot-Platzhalter: System-Notebook mit Status, HDD und Links.
- Typische Anwendung: Typische Nutzung: Laufende Systemdaten, HDD-Slots, Schnelllinks und VPN-Profile pflegen.
- Link-Slots rechts: Der Schnellzugriff bietet 20 frei beschreibbare Slots fuer interne Systemlinks oder externe HTTP-/HTTPS-Ziele. Vorhandene Belegungen der bisherigen Slots 1 bis 15 bleiben bei der Erweiterung erhalten.
- VPN-Slots rechts: Der Schnellzugriff bietet 10 VPN-Slots in einheitlicher Groesse. Vorhandene Belegungen der bisherigen Slots 1 bis 5 bleiben erhalten; die Slots 6 bis 10 werden bei bestehenden Konfigurationen leer ergaenzt.

VPN-Profile und Statusanzeige

- Eine markierte Zeile im VPN-Slot-Tab bestimmt das zu bearbeitende Profil auch dann, wenn das Formularfeld noch leer ist.
- Beim Loeschen eines Profils bereinigt NetMon normale VPN-Slots, primaere VPN-Slots und die Privacy-VPN-Auswahl, sofern sie auf dieses Profil verweisen.
- Nach einem erfolgreichen Profilwechsel fragt NetMon die oeffentliche IP und das Exit-Land neu ab. Die VPN-Ampel im Kernstatus wird nach Abschluss der Abfrage automatisch aktualisiert.

Speedtest und Server-ID

- Das optionale Feld Server-ID legt einen festen Speedtest-Server fest. Ein leeres Feld verwendet die automatische Serverauswahl des installierten Speedtest-CLI.
- Der Bedienblock ist vertikal aufgebaut: oben Server-ID, darunter Speichern, dann Server-IDs abrufen, Speedtest starten und Speedtest-Verlauf.
- Server-IDs abrufen fragt asynchron bis zu 20 verfuegbare beziehungsweise nahe Server ab und schreibt ID, Anbieter, Ort, Land und vorhandene Entfernungsangaben in das NetMon-Log.
- Die gewuenschte ID kann anschliessend in das Feld Server-ID eingetragen und mit Speichern uebernommen werden. Die Abfrage traegt keine ID automatisch ein und startet noch keinen Geschwindigkeitstest.
- Unterstuetzt werden sowohl das offizielle Ookla-CLI speedtest als auch das Python-Werkzeug speedtest-cli; die konkret verfuegbare Serverliste wird vom jeweiligen Online-Dienst geliefert.

Secure Browser

- Zweck: Der Haupttab Secure Browser erstellt fuer ausgewaehlte Webdienste getrennte, gehaertete Firefox-Instanzen mit jeweils eigenem Profil.
- Installationsort: Neue Instanzen liegen gemeinsam unter Dokumente/NetMon/Secure Browser; NetMon legt den Ordner beim ersten Zugriff automatisch an. Ein abweichender XDG-Dokumentenordner wird beruecksichtigt.

- Privatsphaeregewinn: Cookies, Website-Speicher, Verlauf, Logins und Einstellungen bleiben vom normalen Browser und von anderen Secure-Browser-Instanzen getrennt. Das reduziert dienstuebergreifende Wiedererkennung und verhindert, dass mehrere sensible Webanwendungen ungewollt denselben lokalen Browserzustand teilen.
- Hardening: Je nach Sicherheitsprofil koennen Telemetrie, Studies, Pocket, Sync, WebRTC, Formular-/Suchvorschlaege und spekulative Netzwerkzugriffe deaktiviert werden. Strenger Tracking-Schutz, HTTPS-Only und blockierte Drittanbieter-Cookies verringern weitere Datenabfluesse.
- Website-Filter: Im Whitelist-Modus werden standardmaessig alle Ziele blockiert und nur konfigurierte Domains erlaubt. Notwendige Login-, Zahlungs- oder CDN-Domains muessen bewusst ergaenzt werden.
- Cookie-Entscheidung: Dauerhafte Cookies und Logins bieten Komfort, erhalten aber wiedererkennbare Kennungen. Ohne Cookie-Persistenz werden Sitzungen und lokale Spuren beim Beenden konsequenter entfernt.
- Sicherer Einstieg: Zuerst Webseite, Domains und Sicherheitsprofil festlegen, danach den Dry-Run pruefen und erst anschliessend mit Anlegen installieren.
- Grenzen: Der Secure Browser macht nicht anonym. Webseiten sehen weiterhin die oeffentliche IP, koennen Fingerprinting einsetzen und erkennen angemeldete Konten. VPN oder Tor, Browser-Updates und eine enge Whitelist bleiben eigenstaendige Schutzbausteine.
- Tour: Unter Setup -> Tour steht Secure Browser als eigene deutsche oder englische Tour bereit. Die Volle Tour enthaelt diesen Abschnitt automatisch.
- Deinstallation: Der NetMon-Deinstaller fragt separat Installierte Secure Browser ebenfalls deinstallieren? [y/N]. Bei N bleiben Browser, Profile und Starter auch dann erhalten, wenn andere NetMon-Dokumente entfernt werden. Bei Y entfernt der Deinstaller die dedizierte Browser-Ablage, erkannte Legacy-Container und deren NetMon-Menueeintraege.

Scanfunktionen

- Zweck: Download-Scan, ClamAV, Lynis, RKHunter, Startup Guard, Repository Guard und Roots Baseline Guard sind die wichtigsten Scan-/Pruefpfade.
- Beschreibung: Mehrere Bereiche enthalten Baseline-, Vergleichs-, Export- und Trustlist-Logik.
- Screenshot-Platzhalter: Screenshot-Platzhalter: Scan-/Audit-Ansicht.
- Typische Anwendung: Typische Nutzung: Baselines erstellen, Veraenderungen vergleichen, Findings exportieren.

ClamAV-Scans

- Schnellscan-Pfade: Schnellscan-Pfade oeffnet einen Dialog fuer die Standardordner Downloads, Desktop, Dokumente und Documents. Zusaetzliche eigene Ordner lassen sich hinzufuegen und wieder entfernen.
- Speicherung: Die Auswahl bleibt in den NetMon-Einstellungen erhalten. Standard aktiviert wieder alle vorhandenen Standardordner und entfernt die eigenen Eintraege aus der Auswahl.
- Ausschluesse: Beim Schnellscan werden Cache, Papierkorb, Steam-Ordner und Snap-Daten unter dem Benutzerverzeichnis ausgelassen, sofern diese Ordner vorhanden sind.
- Scanarten: Der ClamAV-Bereich bietet Schnellscan, frei gewaehlten Ordnerscan und Systemscan. Der Schnellscan ist nur so klein und schnell wie die dort ausgewaehlten Ordner.
- Fortschritt: Die Laufzeit beginnt mit dem eigentlichen Scan und wird waehrenddessen weitergezaehlt. Die verlaessliche Zahl gescannter Dateien kommt aus der abschliessenden ClamAV-Zusammenfassung; bei

ruhiger --infected-Ausgabe kann sie waehrend des Laufs noch bei 0 stehen.

- Abbruch: Scan abbrechen beendet den laufenden ClamAV-Prozess kontrolliert. Ein Schnellscan hat zusaetzlich ein Zeitlimit von 15 Minuten.
- Signaturen: Ein Update ohne root-Rechte speichert ClamAV-Signaturen unter XDG_DATA_HOME/netmon/clamav-db beziehungsweise ~/.local/share/netmon/clamav-db. NetMon verwendet nur ein eigenes, nicht fuer andere Nutzer zugaengliches Verzeichnis mit vorhandenen .cvd- oder .cld-Dateien.
- Funde: Erkannte Dateien werden zuerst als Befunde gespeichert. Quarantaene und Wiederherstellung bleiben getrennte, vom Benutzer ausgeloeste Aktionen.

Laufenden ClamAV-Scan im Terminal pruefen

```
ps -o pid,etime,%cpu,cmd -C clamscan
```

ELAPSED zeigt die bisherige Laufzeit und %CPU die aktuelle Prozessauslastung. Die vollstaendige, nicht abgeschnittene Kommandozeile inklusive Scanpfaden zeigt:

```
ps -efww | grep '[c]lamscan'
```

Eine hohe CPU-Auslastung bedeutet normalerweise, dass ClamAV noch arbeitet. clamscan --infected liefert jedoch keine verlaessliche Live-Dateinummer; die endgueltige Anzahl steht erst nach Abschluss in der Scan-Zusammenfassung.

RKHunter-Auswertung und Referenzupdate

- Klassifizierung: NetMon ordnet relevante RKHunter- und dpkg -V-Meldungen als erwartete Systemaenderung, pruefenswert oder kritisch ein.
- Ausgabe: Die Auswertung zeigt Kurzbewertung, relevante Auffaelligkeiten, Risiko, konkrete Handlungsempfehlungen und typische Fehlalarme.
- Entwicklerumgebungen: VS Code, Codex, Dev-Container und Docker werden nur erwaehnt, wenn eine tatsaechliche Warnzeile oder ein weiteres Risikosignal vorliegt.
- Referenzupdate: Der Button Vertrauenswuerdige Referenz uebernehmen bleibt deaktiviert, bis ein vollstaendiger Scan niedriges Risiko meldet und dpkg -V nur erwartete Aenderungen zeigt.
- Bestaetigung: sudo rkhunter --propupd wird ausschliesslich nach einer ausdruecklichen Bestaetigung ausgefuehrt. NetMon startet diesen Befehl niemals automatisch.
- Nach dem Update: Ein erfolgreiches Referenzupdate entzieht die Freigabe sofort. Vor einem weiteren Update ist ein neuer vollstaendiger Scan erforderlich.

Logs

- Zweck: Der Log-Tab sammelt App-Ausgaben und kann an andere Bereiche gekoppelte Log-Links enthalten.
- Beschreibung: Beispiel: Nach show_firewall_status_clean() wird ein Link Zur Firewall springen angehaengt.
- Screenshot-Platzhalter: Screenshot-Platzhalter: Log-Tab mit Sprunglink.
- Typische Anwendung: Typische Nutzung: Statusmeldungen lesen und aus dem Log direkt in relevante Tabs springen.

Theme Studio

- Zweck: Der Tools-Untertab Theme Studio dient zum Anpassen des Pop-Testprofils.
- Beschreibung: Es gibt Funktionen fuer Vorschau, Anwenden, Speichern, Laden und Ruecksetzen.

- Screenshot-Platzhalter: Screenshot-Platzhalter: Theme Studio.
- Typische Anwendung: Typische Nutzung: Farben testen und Profile in den Dokumente-Ordner schreiben.

Linux-Hilfe

- Zweck: Die Linux-Hilfe zeigt CSV-/JSON-basierte Kommandoeintraege mit Kategorien, Risiken und Details.
- Beschreibung: Die obere Eintragsliste bietet per Rechtsklick normale Chad-Prompts und den Eintrag Als Prompt mit PDF. Dieser PDF-Prompt ist fuer externe Erklaerung und PDF-Erstellung gedacht; NetMon erzeugt diese PDF nicht lokal.
- PDF-Bibliothek: Die untere Bibliothek importiert heruntergeladene PDFs, zeigt Metadaten und Vorschau, oeffnet PDFs per Doppelklick oder Button und erlaubt per Rechtsklick das Loeschen der PDF inklusive JSON-Metadaten.
- Screenshot-Platzhalter: Screenshot-Platzhalter: Linux-Hilfe.
- Typische Anwendung: Typische Nutzung: Befehl nachschlagen, PDF-Prompt kopieren, externe PDF herunterladen und ueber PDF importieren in NetMon anzeigen.

Fehlerbehebung

Problem	Ursache im Code	Loesung / Workaround
GUI startet nicht	tkinter fehlt oder DISPLAY ist nicht gesetzt (netmon/main.py)	python3-tk installieren oder --headless verwenden; bei Remote-Umgebungen X11/Wayland weiterreichen.
NetMon startet nicht erneut	Instanz-Lock aktiv (acquire_netmon_instance_lock)	Laufende Instanz beenden oder alten Prozess pruefen.
Onlineanalyse funktioniert nicht	OPENAI_API_KEY fehlt (AiOnlineService)	API-Key als Umgebungsvariable setzen.
Linux-Hilfe erzeugt keine PDF direkt	Der aktuelle Workflow ist bewusst auf Prompt-Kopie, externen PDF-Build und Import ausgelegt	Oben Als Prompt mit PDF nutzen, PDF extern erstellen/herunterladen und unten ueber PDF importieren laden.
projectM laesst sich nicht starten	projectM ist optional und wird gesondert erkannt	projectM/Flatpak-Pfade pruefen; im Radio-UI den Installationshinweis nutzen.
Packet Capture / Wireshark fehlt	Wireshark-/tshark-Launcher nicht installiert	Wireshark installieren; der Packet-Capture-Bereich enthaelt einen Installationshinweis.
DNS-/Firewall-Aktionen scheitern	Viele Systemaktionen benoetigen sudo -n oder eine frische sudo-Session	Sudo-Session erneuern und mit passender Linux-Umgebung arbeiten.
Secure Browser erreicht eine benoetigte Seite nicht	Der Website-Filter erlaubt nur die konfigurierte Domain-Liste	Dry-Run und Whitelist pruefen; nur die konkret benoetigten Login-, Zahlungs- oder CDN-Domains ergaenzen.
ClamAV-Schnellscan dauert unerwartet lange	Zu viele oder grosse eigene Scanpfade, Archive, viele kleine Dateien oder langsame Datentraeger	Unter Schnellscan-Pfade die Auswahl verkleinern; Prozessstand mit ps -o pid,etime,%cpu,cmd -C clamscan pruefen.
ClamAV zeigt waehrend des Scans 0 Dateien	clamscan --infected liefert die verwertbare Dateisumme erst in der Abschlussausgabe	Laufzeit und Prozess im Terminal beobachten; die endgueltige Dateizahl nach Scanende ablesen.

Bekannte Probleme

Ursache	Beobachtung	Loesung
Linux Hilfe fuehrt riskante Befehle nicht direkt aus	Der Bereich ist als Referenz-, Prompt- und Bibliotheksfunktion ausgelegt	Befehle pruefen, Prompts extern nutzen und PDFs ueber die Bibliothek verwalten.
Sprachpaket-Hinweis nennt spaetere Vollumsetzung	Sprachunterstuetzung ist vorhanden, aber der Hinweistext ist noch konservativ	Sprachdateien in netmon/lang/ pruefen und bei Bedarf erweitern.
Lizenz-/Dokumentations-/Video-Links enthalten Platzhaltertexte in app.py	Nicht alle Hilfe-/Info-Links sind produktiv angebunden	Externe Hilfe nur verwenden, wenn konkrete Inhalte hinterlegt sind.

FAQ

1. Wie starte ich NetMon ohne GUI?

python -m netmon --headless fuehrt den Headless-Snapshot aus.

1. Wie pruefe ich nur, ob die GUI einmal startet?

python -m netmon --headless-smoke erstellt die GUI einmal und beendet sie wieder.

1. Welche Themes kann ich direkt beim Start waehlen?

--theme akzeptiert laut netmon/main.py die Werte dark, light, beige und pop.

1. Wo speichert NetMon seine Konfiguration?

Standardmaessig unter ~/.config/netmon bzw. XDG_CONFIG_HOME/netmon.

1. Wo speichert NetMon Laufzeit- und Statusdateien?

Standardmaessig unter ~/.local/state/netmon bzw. XDG_STATE_HOME/netmon.

1. Warum startet die GUI nicht?

Im Startpfad werden fehlendes tkinter, fehlendes DISPLAY, Tk-Fehler und weitere Bootstrap-Fehler explizit behandelt. NetMon schreibt dabei ein Startprotokoll nach ~/Documents/NetMon/diagnostics bzw. ~/Dokumente/NetMon/diagnostics und erzeugt zusaetzlich strukturierte Diagnoseberichte als netmon-*.json.

1. Wie sende ich Fehlerprotokolle oder Diagnoseberichte an den Support?

Im Bereich Einstellungen -> Support / Diagnose kann der Benutzer ein Supportpaket erzeugen. Dieses ZIP landet unter ~/Documents/NetMon/support bzw. ~/Dokumente/NetMon/support und enthaelt das aktuelle Start-/Crash-Protokoll sowie den neuesten Diagnosebericht. Zusaetzlich gibt es im About-/Bugreport-Bereich ein Formular, dessen Angaben als Nutzermeldung in das Supportpaket aufgenommen werden.

Vor dem Export zeigt NetMon im Support-Bereich ausserdem einen sichtbaren Datenschutzhinweis an: Diagnosepakete dienen nur der Fehleranalyse, koennen aber je nach Fall Hostnamen, Benutzernamen, Dateipfade oder IP-Adressen enthalten und sollten vor dem Versand kurz geprueft werden.

1. Darf NetMon mehrfach parallel laufen?

Nein. netmon/main.py verwendet einen Instanz-Lock und beendet weitere Starts mit einer Fehlermeldung.

1. Wie aktiviere ich DNS Guard?

Im DNS-Tab gibt es den Schalter DNS Guard aktivieren; Statusvariablen und Logmeldungen werden im Code gesetzt.

1. Was macht DNS Guard?

Er ueberwacht die aktiven DNS-Server, bewertet Abweichungen und kann je nach Zustand eine Auto-Korrektur anstossen.

1. Wie erkenne ich, ob DoT aktiv ist?

Die App fuehrt DoT-Statusvariablen wie dot_status_var, dot_server_var und dot_last_test_var.

1. Wo finde ich den Radio-Bereich?

Als Radio-Tab im verschachtelten Tools-Notebook.

1. Kann ich den Radio-Player losloesen?

Ja. Der Code in netmon/netmon_radio_core.py und netmon/features/radio_controller.py verwaltet Host-/Handover-/Attach-Dateien fuer eingebetteten und losgeloesten Betrieb. Das losgeloeste Standalone-Radio laeuft dabei aktuell ausschliesslich im Darkmode.

1. Wo liegt die Radio-Datenbank?

Im State-Verzeichnis, standardmaessig als radio_db.json und radio_db_meta.json.

1. Wie speichere ich Theme-Studio-Profile?

Das Theme Studio bietet Theme speichern und Theme laden; Profile landen unter ~/Documents/NetMon/ThemeStudio oder ~/Dokumente/NetMon/ThemeStudio.

1. Wirkt Theme Studio auf alle Themes?

Nein. Laut UI-Hinweis wirkt Theme Studio nur auf das Testprofil Pop.

1. Wie funktioniert Chad-Onlineanalyse?

Der Online-Service liest OPENAI_API_KEY und sendet Requests an /v1/responses.

1. Was passiert ohne API-Key?

Der Online-Service antwortet mit Kein API-Key gesetzt und fuehrt keine Onlineanalyse aus.

1. Wo finde ich System- und Security-Funktionen?

Im Hauptnotebook gibt es die Bereiche System, Tools und Security mit jeweils eigenem Unter-Notebook.

1. Gibt es einen Packet-Capture-Export nach Wireshark?

Ja. Packet-Capture-Controller und UI enthalten Wireshark-/tshark-Erkennung und einen Button In Wireshark oeffnen.

1. Wo sehe ich Logs?

Im Tab Log; dort existiert auch der Link aus dem Firewall-Status direkt in den Firewall-Bereich.

1. Wie erkennt NetMon eine neue Version?

Im About-Dialog kann NetMon die aktuelle Website-Version ueber netmonroots.com/latest-version.json pruefen. Wenn online eine neuere Version gemeldet wird, kann NetMon direkt auf die Downloadseite weiterleiten.

Unklar / Nicht dokumentiert

- Eine formale Lizenzdatei wurde im Repository nicht gefunden.
- Ein Maintainer-/Kontaktabschnitt mit belastbarer Kontaktadresse wurde nicht gefunden.

- Fuer einige Platzhalterbereiche (z. B. Dokumentations- oder Video-Links im UI) ist kein produktiver Inhalt im Repository hinterlegt.
- Fuer mehrere Systemaktionen ist die Zielumgebung implizit Linux mit passenden Tools; eine distributionsuebergreifende Installationsmatrix ist im Code nicht dokumentiert.

CONFIGURATION

Quelle: docs/CONFIGURATION.md

Konfiguration NetMon

Speicherorte

- Konfigurationsverzeichnis: XDG_CONFIG_HOME/netmon oder ~/.config/netmon (netmon/core/state.py).
- Status-/State-Verzeichnis: XDG_STATE_HOME/netmon oder ~/.local/state/netmon (netmon/core/state.py).
- Dokumentenverzeichnis: ~/Documents/NetMon oder ~/Dokumente/NetMon (netmon/core/state.py).
- Diagnoseordner: ~/Documents/NetMon/diagnostics oder ~/Dokumente/NetMon/diagnostics (netmon/core/diagnostics.py).
- Supportordner: ~/Documents/NetMon/support oder ~/Dokumente/NetMon/support (netmon/core/support_bundle.py).
- Theme-Studio-Profile: ~/Documents/NetMon/ThemeStudio oder ~/Dokumente/NetMon/ThemeStudio (app.py).
- Linux-Hilfe-PDF-Bibliothek: ~/Documents/NetMon/LinuxHelp oder ~/Dokumente/NetMon/LinuxHelp mit pdfs/ und meta/ (netmon/services/linux_help_library_service.py).
- ClamAV-Benutzersignaturen: XDG_DATA_HOME/netmon/clamav-db oder ~/.local/share/netmon/clamav-db (netmon/features/clamav_controller.py).

Konfigurationsdateien

Datei	Standardort	Zweck
config.json	Config	App-Einstellungen (self.netmon_config_path)
ai_config.json	Config	Chad-/AI-bezogene Einstellungen (self.ai_config_path)
linux_help_commands.json	Config	Linux-Hilfe-Daten
netmon_hdd_slots.json	Config	HDD-Slot-Konfiguration
netmon_link_slots.json	Config	Link-Slot-Konfiguration
netmon_vpn_slots.json	Config	VPN-Slots
netmon_primary_vpn_slots.json	Config	Primaere VPN-Slots
netmon_privacy.json	Config	Privacy-/DNS-/VPN-Einstellungen
radio_stations.json	Config	Radio-Stationen
radio_favorites.json	Config	Radio-Favoriten
radio_settings.json	Config	Radio-Einstellungen

Die Schnellscan-Auswahl liegt in config.json:

- clamav_quickscan_targets: aktivierte Standardordner (downloads, desktop, documents_de, documents_en).
- clamav_quickscan_custom_paths: zusaetzliche, vom Benutzer ausgewaehlte absolute Ordnerpfade.

- Sind noch keine Werte gespeichert, verwendet NetMon die vorhandenen Standardordner. Im ClamAV-Tab koennen die Pfade ueber Schnellscan-Pfade geaendert und auf den Standard zurueckgesetzt werden.

State- und Laufzeitdateien

Datei / Ordner	Standardort	Zweck
radio_db.json	State	Radio-Datenbank
radio_db_meta.json	State	Radio-DB-Metadaten
radio_host_state.json	State	Host-Status fuer eingebetteten Radio-Betrieb
radio_attach_request.json	State	Attach-Anforderung
radio_standalone_attach_request.json	State	Standalone-Attach-Anforderung
radio_standalone_show_request.json	State	Show-Anforderung fuer Standalone-Radio
radio_detached_state.json	State	Status fuer losgeloesten Player
radio_handover_request.json	State	Handover an externen Player
radio_handover_ack.json	State	Handover-Bestaetigung
netmon_security_baseline.json	State	Security-Baseline
startup_guard_baseline.json	State	Startup-Guard-Baseline
startup_guard_trustlist.json	State	Startup-Guard-Trustlist
repository_guard_baseline.json	State	Repository-Guard-Baseline
repository_guard_trustlist.json	State	Repository-Guard-Trustlist
process_guard_trustlist.json	State	Process-Guard-Trustlist
roots_baseline_guard_baseline.json	State	Roots-Baseline-Guard-Baseline
roots_baseline_guard_whitelist.json	State	Roots-Baseline-Guard-Whitelist
netmon_clamav_findings.json	State	Persistierte ClamAV-Funde
captures/	State	Packet-Capture-Dateien
logs/speedtest_history.jsonl	State	Speedtest-Historie
logs/	State	Laufzeit- und Feature-Logs, z. B. speedtest_history.jsonl
WireGuard/	Documents	NetMon-Importordner fuer lokale WireGuard-Profile
LinuxHelp/pdfs/	Documents	Importierte Linux-Hilfe-PDFs
LinuxHelp/meta/	Documents	JSON-Metadaten zu importierten Linux-Hilfe-PDFs

Diagnose- und Supportdateien

Datei / Ordner	Standardort	Zweck
startup_error.log	Documents/diagnostics	Startfehlerprotokoll mit Bootstrap- und GUI-Startfehlern
crash.log	Documents/diagnostics	Sammellog fuer Laufzeitabstuerze

Datei / Ordner	Standardort	Zweck
netmon-*.json	Documents/diagnostics	Strukturierte Diagnoseberichte fuer Start- und Laufzeitfehler
netmon-support-*.zip	Documents/support	Supportpaket mit Manifest, Diagnosebericht und Fehlerprotokollen

Standardwerte

- App-Version laut netmon/config/constants.py: 1.1.19
- Theme-Startwert: light als bevorzugtes Theme in App.__init__.
- Das losgeloeste Standalone-Radio erzwingt aktuell dark als Start- und Laufzeittheme; der eingebettete Radio-Tab folgt weiter dem NetMon-/Radio-Theme.
- Chad-Startmodell: gpt-4.1-mini.
- DNS Guard startet mit False / inaktiv, mehrere Statusvariablen initialisieren auf Platzhaltertexte.
- Fehler- und Supportdiagnose ist standardmaessig aktiv; Supportpakete werden bei Bedarf manuell aus der UI erzeugt.
- Der About-Dialog kann online https://netmonroots.com/latest-version.json pruefen und bei einer neueren Website-Version auf die Downloadseite verweisen.

Backup / Restore

- Das Repository enthaelt Shell-Skripte fuer Backups und Restore unter scripts/, z. B. backup_dev_lab.sh, restore_dev_lab_backup.sh und restore_selected_dev_lab_backup.sh.
- Zusaetzlich existiert bereits BACKUP_RESTORE.md.
- Fuer Feature-spezifische JSON-Dateien erfolgt die Persistenz ueber safe_write_json() oder lokale JSON-Schreiber.

Unklar / Nicht dokumentiert

- Es gibt keine zentrale schema-basierte Dokumentation aller JSON-Dateien.
- Defaultwerte vieler einzelner Settings liegen verteilt in App.__init__, _default_settings_payload() und Feature-Modulen; nicht jedes Setting ist separat kommentiert.

Hinweise zu VPN-/WireGuard-Profilen

- NetMon liest WireGuard-Profile nicht nur aus Slot-JSON-Dateien, sondern auch aus:
- /etc/wireguard/*.conf
- Documents/NetMon/WireGuard bzw. Dokumente/NetMon/WireGuard
- echten NetworkManager-wireguard-Connections
- Seit dem aktuellen Stand werden dabei keine allgemeinen NetworkManager-vpn-Connections mehr als WireGuard-Profile in den NetMon-Profilpool aufgenommen.

SECURITY

Quelle: docs/SECURITY.md

Security-Hinweise NetMon

Netzwerkzugriffe

Im Code wurden folgende Arten von Netzwerkzugriffen gefunden:

- DNS-/Resolver-Pruefungen ueber resolvectl, nmcli und systemnahe Linux-Kommandos im DNSControlService.
- Netzwerk- und Portpruefungen ueber Socket-/Tooling-Pfade (port_analysis.py, Trace, MTR, Nmap, iftop, Packet Capture).
- Onlinezugriffe fuer Chad/OpenAI ueber AiOnlineService (https://api.openai.com/v1/responses).
- Radio-HTTP-Zugriffe in netmon_radio_app.py auf Radio-Browser-Endpunkte und Logo-/DB-Downloads.
- Repository-Guard-Pruefungen fuer APT-/Flatpak-Quellen.

DNS-Aenderungen

- DNSControlService enthaelt Reparatur- und Policy-Funktionen fuer DNS-/DoT-/NetworkManager-Konfigurationen.
- Der Service nutzt u. a. sudo -n nmcli, sudo -n mkdir, sudo -n install, sudo -n rm und sudo -n systemctl restart systemd-resolved.
- DNS Guard kann Abweichungen erkennen und je nach Pfad Auto-Korrekturen anstossen.

Root-Rechte / sudo

- Der generische Command-Layer prueft sudo -n true und kann ueber renew_sudo_timestamp() eine frische sudo-Session anfordern.
- UFW-/Firewall-, WireGuard-, DoT-, Packet-Capture-, RKHunter- und Cleanup-Funktionen enthalten sudo-nahe Systemkommandos.
- Der Cleanup Assistant markiert eigene run_sudo_command-Pfade als bestaetigungspflichtig und nutzt einen separaten ensure_sudo_session-Flow.

RKHunter-Referenzdatenbank

rkhunter --propupd kann manipulierte Dateieigenschaften als neue Referenz akzeptieren und ist deshalb besonders geschuetzt:

- Der Befehl wird niemals automatisch ausgefuehrt.
- Ein vollstaendiger RKHunter-Scan und eine erfolgreiche dpkg -V-Pruefung sind erforderlich.
- Rootkits, Backdoors, verdaechtige Laufzeit- oder Netzwerkfunde, manipulierte Systemprogramme, gelbe Prueffaelle und Scanfehler blockieren die Freigabe.
- Das berechnete Gesamtrisiko muss Niedrig sein.
- Der Benutzer muss den Vorgang ausdruecklich bestaetigen und eine sudo-Sitzung freigeben.

- Das Sicherheits-Gate wird vor dem Dialog, nach der Bestaetigung und nach der sudo-Freigabe erneut geprueft.
- Jeder neue, fehlgeschlagene oder blockierte Scan sowie jedes Referenzupdate entzieht eine vorherige Freigabe.

ClamAV-Signaturdatenbank

- Ein benutzerbezogenes freshclam-Update schreibt Signaturen nach XDG_DATA_HOME/netmon/clamav-db oder ersatzweise ~/.local/share/netmon/clamav-db, nicht in einen vorhersagbaren gemeinsamen /tmp-Ordner.
- NetMon erstellt den Ordner ohne Symlink-Folgen und mit Benutzerrechten. Vor clamscan --database werden Eigentuemerschaft, Verzeichnistyp und fehlende Gruppen-/Fremdrechte geprueft.
- Als benutzerbezogene Datenbank werden nur Verzeichnisse mit vorhandenen .cvd- oder .cld-Signaturen akzeptiert. Andernfalls wird auf eine gueltige Systemdatenbank unter /var/lib/clamav beziehungsweise auf die ClamAV-Standardsuche zurueckgefallen.
- Schnellscan-Pfade werden vom Benutzer ausgewaehlt und als normalisierte absolute Pfade gespeichert. Nicht mehr vorhandene Ordner werden beim Scan uebersprungen.
- Quarantaene- und Wiederherstellungspfade werden getrennt validiert; ein Fund wird nicht allein durch den Scan automatisch verschoben oder wiederhergestellt.

Logging

- Startfehler enthalten argv, cwd, DISPLAY und optional Stacktraces in ~/Documents/NetMon/diagnostics/startup_error.log bzw. ~/Dokumente/NetMon/diagnostics/startup_error.log.
- Laufzeitfehler koennen zusaetzlich strukturierte Diagnoseberichte als netmon-*.json im Diagnoseordner erzeugen.
- Ueber Support / Diagnose in den Einstellungen oder ueber den About-/Bugreport-Bereich kann der Benutzer ein Supportpaket als ZIP fuer den Versand an den Support erzeugen.
- Im Support-Bereich weist NetMon vor dem Versand sichtbar darauf hin, dass Diagnosepakete systembezogene Daten wie Hostnamen, Benutzernamen, Dateipfade oder IP-Adressen enthalten koennen.
- Der Log-Tab sammelt Laufzeitmeldungen; aus einzelnen Meldungen kann in andere Tabs gesprungen werden.
- Mehrere Features persistieren strukturierte JSON-/JSONL-Dateien im Config-/State-Verzeichnis.

Datenschutz

- AiAnalysisService redigiert inhaltlich mehrere sensible Muster (Bearer-Token, API Keys, E-Mail-Adressen, Home-Pfade, Passphrasen), bevor Inhalte in Prompts fliessen.
- AiOnlineService verwendet OPENAI_API_KEY aus der Umgebung und sendet Inhalte an die OpenAI-Responses-API.
- Diagnose- und Supportberichte verwenden Redaction-Helfer fuer typische sensible Werte, ersetzen aber keine manuelle Sichtpruefung vor dem Versand.
- Der sichtbare Datenschutzhinweis im Support-Tab fordert deshalb ausdruecklich zur Pruefung vor dem Versand auf und beschreibt die beabsichtigte Nutzung nur fuer Fehleranalyse und Supportbearbeitung.

- Radio- und Netzwerkfunktionen koennen externe Dienste kontaktieren; die konkreten Endpunkte sind im Code sichtbar.

Secure Browser

- Der Secure Browser trennt Webdienste in eigene Firefox-Profile. Cookies, Website-Speicher, Verlauf, Logins und Einstellungen werden dadurch nicht automatisch mit dem normalen Browser oder anderen Secure-Browser-Instanzen geteilt.
- Das generierte Hardening reduziert unter anderem Telemetrie, Sync-, WebRTC-, Vorschlags-, Prefetch- und Drittanbieter-Cookie-Pfade. Das konkrete Verhalten haengt vom gewaehlten Sicherheitsprofil ab.
- Der optionale Firefox-Website-Filter arbeitet nach dem Prinzip blockieren, ausser explizit erlaubt. Eine breite Whitelist oder deaktivierter Filter reduziert die Schutzwirkung.
- Cookie- und Passwortmanager-Persistenz sind bewusste Komfort-/Datenschutzentscheidungen und sollten pro Webdienst festgelegt werden.
- Die Funktion ist keine Anonymisierung: Oeffentliche IP, Browser-/Geraete-Fingerprinting und eine Anmeldung am Benutzerkonto bleiben fuer den Webdienst erkennbar.
- Vor dem Anlegen sollte der Dry-Run geprueft werden. Browser-Updates, enge Domain-Ausnahmen und bei Bedarf ein separater VPN-/Tor-Pfad bleiben notwendig.

Risiken

Bereich	Risiko	Beleg im Code
Systemkommandos	Viele Aktionen veraendern Netzwerk-, DNS- oder Firewall-Zustand auf dem Host	sudo -n-Kommandos in netmon_network_service.py, dns_control_service.py, firewall_controller.py
Onlineanalyse	Inhalte werden an einen externen API-Dienst gesendet	AiOnlineService
Start-Error-Logging	Startlog schreibt Umgebungsdaten und Stacktraces in Datei	netmon/main.py
Packet Capture	Rohdaten aus dem Netzwerk koennen lokal gespeichert und in Wireshark geoeffnet werden	Packet-Capture-Tab/Controller
Detached Radio	Handover-/Status-Dateien im State-Verzeichnis steuern Prozesswechsel	netmon_radio_core.py

Schutzmassnahmen

- Crash- und Locking-Mechanismen im Startpfad.
- Atomische JSON-Schreibfunktion safe_write_json() fuer viele Persistenzpfade.
- Redaction-Logik in AiAnalysisService.
- Sudo-Session-Pruefung vor kritischen Systemaenderungen.
- Trustlist-/Baseline-Konzepte fuer mehrere Security-Guards.
- Eigentums-, Rechte- und Symlink-Pruefungen fuer benutzerbezogene ClamAV-Signaturen und Quarantaenepfade.

Benutzerhinweise

- Kritische Systemfunktionen nur auf einem Linux-System mit bekannter Konfiguration und gueltiger sudo-Session ausfuehren.
- Onlineanalyse nur mit bewusst gesetztem API-Key und nach Pruefung des zu sendenden Inhalts nutzen.
- Packet-Capture- und Wireshark-Funktionen mit Bedacht einsetzen, da Rohdaten gespeichert werden.
- Config-/State-Verzeichnisse regelmaessig sichern, wenn Baselines, Trustlists oder Radio-Daten wichtig sind.

Unklar / Nicht dokumentiert

- Es existiert keine zentrale Datenschutz- oder Datenaufbewahrungsrichtlinie im Repository.
- Die exakten Zugriffsmodelle fuer alle externen Radio-/Logo-Endpunkte sind nicht separat dokumentiert.

Audit-Einordnung K1 bis K9

Die in scripts/build_security_audit_pdf.py aufgefuehrten Punkte K1 bis K9 sind im Audit selbst nicht als bestaetigte Sicherheitsluecken bewertet, sondern als ausgeschlossene Kandidaten mit niedriger Konfidenz.

ID	Thema	Audit-Status	Einordnung	Aktueller Repo-Status
K1	Git-Subcommand-Passthrough	ausgeschlossen	Lokaler Entwicklerpfad, kein LLM- oder Netzwerk-Input in run_command()	zusaetzlich gehaertet: Git-Subcommand-Allowlist in agent/runner.py
K2	WireGuard-Endpoint in UFW-Killswitch	ausgeschlossen	Profil liegt root-owned; Audit sah keinen realistischen Remote-Pfad	zusaetzlich gehaertet: Endpoint-Host wird als feste IP validiert
K3	UFW-Quelle any im Firewall-Formular	ausgeschlossen	beabsichtigte Admin-Funktion mit Warnung, Vorschau und Bestaetigung	unveraendert, weiter als kontextuell unkritisch bewertet
K4	Vorhersagbarer ClamAV-Temp-Pfad	ausgeschlossen	Audit sah keinen sinnvollen Code-Execution-Pfad ueber signierte ClamAV-Daten	zusaetzlich gehaertet: symlink-sichere Verzeichniserstellung
K5	Dateipfad in image_to_data_url()	ausgeschlossen	bewusster Upload-Fluss, Bildtypen-Restriktion, Dateidialog	zusaetzlich gehaertet: Pfad auf Benutzer-Home begrenzt
K6	WireGuard-Import via nmcli	ausgeschlossen	nmcli/NetworkManager fuehrt wg-quick-Hooks nicht aus	unveraendert, weiter als unkritisch bewertet
K7	WireGuard-Import via wg-quick	ausgeschlossen	manueller Admin-Import, kein automatischer wg-quick up-Pfad	unveraendert, weiter als unkritisch bewertet
K8	Shell-Snippet-Generierung im Repository-Guard	ausgeschlossen	Snippet wird nur angezeigt bzw. kopiert, nicht automatisch ausgefuehrt	zusaetzlich gehaertet: shlex.quote() fuer Shell-Werte

ID	Thema	Audit-Status	Einordnung	Aktueller Repo-Status
K9	Symlink-Bypass im Roots-Baseline-Guard	ausgeschlossen	Audit bewertete den praktischen Mehrwert des Bypasses als gering	zusaetzlich gehaertet: os.path.realpath() vor Guard-Match

Kurz gesagt: K1 bis K9 sind keine bestaetigten Sicherheitsbefunde aus dem Audit. Teile davon wurden spaeter trotzdem als Defense-in-Depth-Massnahmen technisch gehaertet.

Bestaetigte Findings nach dem Audit

Radio-Station-URLs aus externem API

- Bereich: netmon/features/radio_controller.py
- Risiko: Low Severity
- Einordnung: bestaetigtes Finding

Radio-Stationen aus externen Quellen wie radio-browser.info wurden zuvor ohne Schema-Validierung weiterverarbeitet. Dadurch konnten praepartierte URLs bis an mpv und an urllib.request.urlopen() gelangen.

Der aktuelle Codestand haertet diesen Pfad zentral in normalize_station_entry():

- erlaubt sind nur http, https, rtsp und rtmp
- url und url_resolved werden beide normalisiert
- Eintraege mit unerlaubtem Schema werden verworfen

Ziel der Aenderung ist, unnoetige SSRF- oder lokale Protokollpfade aus externen Stationslisten gar nicht erst in nachgelagerte Playback- oder Metadaten-Funktionen zu uebernehmen.

rkhunter WEB_CMD und PATH-Hijacking

- Bereich: netmon/core/rkhunter_manager.py
- Risiko: ehemals High Severity
- Einordnung: bestaetigtes Finding, geschlossen

Der Rueckgabewert von shutil.which() durfte nicht ungeprueft in WEB_CMD=... fuer rkhunter.conf uebernommen werden, weil rkhunter --update diesen Pfad spaeter mit Root-Rechten ausfuehrt.

Der aktuelle Codestand begrenzt _preferred_web_cmd() auf vertrauenswuerdige Pfade:

- erlaubt sind nur Treffer unter /usr/bin/, /bin/ und /usr/local/bin/
- user-kontrollierte Treffer wie ~/.local/bin/wget werden verworfen
- falls nur untrusted Kandidaten gefunden werden, bleibt WEB_CMD leer

Ziel der Aenderung ist, PATH-Hijacking und moegliche Privilege Escalation ueber manipulierte Benutzer-Binaries zu verhindern.

Agent-CLI und Repo-Pfad-Confinement

- Bereich: agent/runner.py
- Risiko: ehemals Medium Severity
- Einordnung: bestaetigtes Finding, geschlossen

Der lokale Agent-Runner pruefte urspruenglich nur den Basisbefehl (black, ruff, pytest, ls), aber keine uebergebenen Pfadargumente. Dadurch konnten absolute Pfade ausserhalb des Repositories an Schreib- oder Lese-Tools weitergereicht werden.

Der aktuelle Codestand haertet run_command() zusaetzlich:

- Pfadargumente fuer black, ruff, pytest und ls werden kanonisiert
- erlaubt sind nur Ziele innerhalb von REPO_ROOT
- auch pytest-Node-IDs wie tests/test_x.py::test_name werden korrekt auf

den Dateipfad reduziert und geprueft

Ziel der Aenderung ist, Dateikorruption und Information Disclosure ausserhalb des Repositories ueber Prompt-Injection oder untrusted CLI-Eingaben zu verhindern.

Fun-Launcher und Spielpfad-Begrenzung

- Bereich: netmon/features/fun_controller.py
- Risiko: ehemals Medium Severity
- Einordnung: bestaetigtes Finding, geschlossen

Der Fun-Launcher startet Python-Dateien aus dem Code-the-Classics-Bereich ueber subprocess.Popen(["python3", path]). Der aktuelle Codestand begrenzt diesen Pfad jetzt zusaetzlich:

- gestartet werden nur .py-Dateien innerhalb von fun_games_dir
- der Pfad wird per realpath() gegen den Spieleordner geprueft
- vorhandene NetMon-Managed-Path-Checks werden genutzt, wenn verfuegbar

Ziel der Aenderung ist, das Starten beliebiger Python-Dateien ausserhalb des verwalteten Spielebereichs defensiv zu verhindern.

ARCHITECTURE

Quelle: docs/ARCHITECTURE.md

Architektur NetMon

Systemarchitektur

NetMon ist eine monolithische Tk-/ttk-Desktop-Anwendung mit einer grossen zentralen App-Klasse und ausgelagerter Fachlogik in Controllern, Services, Core-Helfern und UI-Tab-Builders.

Architekturueberblick

```
flowchart TD
    A[python -m netmon] --> B[netmon/__main__.py]
    B --> C[netmon/main.py]
    C --> D{CLI-Modus?}
    D -- headless --> E[run_headless_snapshot]
    D -- headless-smoke --> F[run_headless_smoke]
    D -- GUI --> G[Startup-Checks
tkinter / DISPLAY / Lock / Splash]
    G --> H[App in netmon/app.py]
    H --> I[Main Notebook]
    I --> J[System Notebook]
    I --> K[Tools Notebook]
    I --> L[Security Notebook]
    H --> M[Controller in netmon/features]
    H --> N[Services in netmon/services]
    H --> O[Core-Helfer in netmon/core]
    H --> P[Radio Core / Detached Radio]
```

GUI-Architektur

```
flowchart LR
    App[App]
    App --> MainNB[Hauptnotebook self.nb]
    MainNB --> Log[Log]
    MainNB --> Trace[Trace]
    MainNB --> Conn[Verbindungen]
    MainNB --> DNS[DNS]
    MainNB --> Chad[Chad]
    MainNB --> Download[Download Scan]
    MainNB --> Settings[Settings]
    MainNB --> SystemMain[System]
    MainNB --> ToolsMain[Tools]
    MainNB --> SecurityMain[Security]
    SystemMain --> SystemNB[system_nb]
    ToolsMain --> ToolsNB[tools_nb]
    SecurityMain --> SecurityNB[security_nb]
```

Backend-Architektur

```
flowchart TD
    App --> Features[features/*_controller.py]
    Features --> Services[services/*.py]
    Features --> Core[core/*.py]
    App --> Tabs[ui/tabs/*.py]
    Tabs --> App
    App --> RadioCore[netmon_radio_core.py]
    App --> SecurityParser[netmon_security_parser.py]
    App --> SecurityStore[netmon_security_store.py]
    Services --> State[state.py]
    Services --> Utils[utils.py]
```

Konfigurations- und Laufzeitdaten

```
flowchart LR
    Config[~/.config/netmon] --> AppCfg[config.json / ai_config.json]
    Config --> Slots[HDD/Link/VPN/Privacy/Radio JSON]
    Config --> LinuxHelp[linux_help_commands.json]
    Config --> RadioCfg[radio_stations.json / radio_favorites.json / radio_settings.json]
    Data[XDG_DATA_HOME/netmon] --> ClamDb[clamav-db / Benutzersignaturen]
    State[~/.local/state/netmon] --> Sec[security / startup / repository / roots baseline Dateien]
    State --> RadioState[radio_db.json / handover / attach / detached state]
    State --> Logs[startup_error.log / logs/speedtest_history.jsonl]
    State --> Capture[captures/]
```

GUI

- Tk-Basis ueber tkinter/ttk.
- Hauptnavigation ueber drei Ebenen: Haupttabs, verschachtelte System-/Tools-/Security-Notebooks und bereichsspezifische Widgets.
- Theme-System in app.py mit Light, Dark, Beige und Pop sowie Theme Studio fuer das Pop-Profil.

Backend

- Controller-Module in netmon/features/ kapseln UI-nahe Aktionen und Dateipfade.
- Services in netmon/services/ kapseln Parser, Policies, Guards oder Onlinezugriffe.
- Core-Helfer in netmon/core/ kapseln Basiskommandos, Locking, Crash-Handling, Pfad- und Formatierungslogik.
- clamav_controller.py verwaltet Scanarten, Schnellscan-Konfiguration, Fortschrittsanzeige, Signaturauswahl, Befunde und Quarantaene.
- security_audit_controller.py orchestriert RKHunter-Scans und Referenzupdates; rkhunter_analysis_service.py klassifiziert die Ausgabe und liefert die Gate-Evidenz.

Netzwerkmodule

- NetmonNetworkService: sudo-nahe Adapter fuer WireGuard und UFW.
- DNSControlService: Resolver-/VPN-/DoT-/DNSSEC-Evaluation und Reparaturen.
- port_analysis.py, speedtest.py, Trace-/MTR-/Nmap-/iftop-/htop-Tabs: klassische Netzwerkanalyse.
- Packet-Capture-Controller: tcpdump/Wireshark/tshark-nahe Integration.

Logging

- Startfehler werden in eine dedizierte State-Datei geschrieben.
- Laufende Aktionen landen im Log-Tab.
- Mehrere Bereiche schreiben JSON- oder JSONL-Dateien fuer Persistenz, Baselines, History oder Handover.

Konfiguration

- XDG-basierte Config-/State-Verzeichnisse ueber netmon/core/state.py.
- Dateinamen zentral in netmon/config/constants.py.
- Viele Features bekommen ihre Pfade zentral in App.__init__.

Moduldiagramm nach Codebereichen

```
flowchart TD
    subgraph Core
        C1[command.py]
        C2[state.py]
        C3[utils.py]
        C4[lock.py]
        C5[crash.py]
    end
    subgraph Services
        S1[dns_control_service.py]
        S2[ai_online_service.py]
        S3[repository_guard_service.py]
        S4[startup_guard_service.py]
        S5[path_guard_service.py]
        S6[process_guard_service.py]
        S7[rkhunter_analysis_service.py]
    end
    subgraph Features
        F1[dns_controller.py]
        F2[firewall_controller.py]
        F3[vpn_controller.py]
        F4[radio_controller.py]
        F5[security_monitoring_controller.py]
        F6[security_audit_controller.py]
        F7[repository_guard_controller.py]
        F8[clamav_controller.py]
    end
    subgraph UI
        U1[ui/tabs/*.py]
        U2[ui/widgets/*.py]
    end
    App[app.py] --> Core
    App --> Services
    App --> Features
    App --> UI
    Features --> Services
    Features --> Core
    UI --> App
```

Unklar / Nicht dokumentiert

- Es existiert keine separate Architektur-Dokumentation im Quellcode; die hier gezeigte Architektur ist aus Importen, Buildern und Aufrufpfaden abgeleitet.
- Einzelne UI-Texte beschreiben noch Platzhalter oder spaetere Integrationsschritte; sie spiegeln nicht immer den aktuellen Implementierungsstand 1:1 wider.

DEB PACKAGING PLAN

Quelle: docs/DEB_PACKAGING_PLAN.md

Debian Packaging Plan

Diese Datei beschreibt die Soll-Struktur fuer netmon als .deb auf Basis des aktuellen Repository-Stands.

Aktueller dokumentierter Paketstand: 1.1.18.

Ziel

Die .deb soll die startfaehige NetMon-Anwendung enthalten, inklusive aller zur Laufzeit benoetigten Python-Module aus dem Repo, der Desktop-Integration und der statischen App-Daten. Entwicklungsdateien, Tests, Backups und Build-Artefakte gehoeren nicht ins Paket.

Vorgeschlagene Paketstruktur

Debian-Metadaten

- DEBIAN/control
- DEBIAN/postinst
- DEBIAN/prerm

Starter und Desktop-Integration

- /usr/bin/netmon
- /usr/bin/netmon-uninstall
- /usr/share/applications/netmon.desktop
- /usr/share/applications/netmon-uninstall.desktop
- /usr/share/icons/hicolor/256x256/apps/netmon.png

Applikationscode

Basisziel: /usr/lib/netmon/netmon/

Direkt unter /usr/lib/netmon/netmon/:

- __init__.py
- __main__.py
- main.py
- app.py
- netmon_ops.py
- netmon_network_service.py
- netmon_state_coordinator.py
- netmon_security_parser.py
- netmon_security_store.py
- netmon_projectm_sidecar.py

- netmon_radio_app.py
- netmon_radio_core.py
- linux-befehle.csv

Unterverzeichnisse, die komplett mitgenommen werden sollen:

- /usr/lib/netmon/netmon/config/
- /usr/lib/netmon/netmon/core/
- /usr/lib/netmon/netmon/features/
- /usr/lib/netmon/netmon/lang/
- /usr/lib/netmon/netmon/modules/
- /usr/lib/netmon/netmon/services/
- /usr/lib/netmon/netmon/ui/

Statische Daten und Assets

Mitliefern

- /usr/share/netmon/assets/log-pics/
- Quelle: tools/Log-Pics/
- Zweck: Seed-Bilder fuer Log-/Overlay-Funktionen
- features/Code-the-Classics.zip
- Zielpfad: /usr/lib/netmon/netmon/features/Code-the-Classics.zip
- Soll offiziell Teil des Pakets sein
- projectM-Presets
- Sollen offiziell mit ausgeliefert werden
- Quellen:
- tools/MD2 Presets Edited for projectM 4.x/
- optional zusaetzlich ausgewaehlte Inhalte aus tools/mdpresets/, falls wir diese kuratieren wollen
- Aktueller Paketpfad: /usr/lib/netmon/tools/
- Hintergrund: Der Laufzeitcode sucht die gebuendelten Preset-Quellen aktuell repo-relativ unter tools/

Optional mitliefern

- derzeit nichts weiter

Nicht in die .deb

- tests/
- backups/
- dist/
- .venv/

- venv/
- __pycache__/
- .pytest_cache/
- .ruff_cache/
- Repo-interne Hilfsskripte ausser dem eigentlichen Paket-Build
- alternative Icons und Dev-Artefakte ohne Laufzeitbezug

Abhaengigkeiten

Die bisherige control-Datei im Build-Skript ist zu schmal und teilweise veraltet. Sie basiert noch auf einem kleineren Altpaket.

Depends

Diese Abhaengigkeiten sind fuer einen normalen GUI-Start plausibel und sollten als erste Zielmenge gelten:

- python3
- python3-tk
- python3-psutil
- python3-pil
- python3-pil.imagetk
- udisks2
- cryptsetup

Recommends

Diese Tools werden laut Code aktiv genutzt, sind aber eher Feature-bezogen als fuer den nackten Start zwingend:

- network-manager
- systemd-resolved
- dnsutils
- wireguard-tools
- ufw
- traceroute
- mtr
- nmap
- iproute2
- iftop
- htop
- lynis
- rkhunter

- clamav
- clamav-freshclam
- speedtest-cli
- mpv
- whois

Suggests

Diese Komponenten sind optional oder eher Zusatzfunktionen:

- projectm
- projectm-pulseaudio
- wireshark
- tshark
- python3-pypdf2
- python3-docx
- python3-pygame
- python3-pgzero

Hinweise zur Python-Laufzeit

- requirements.txt nennt aktuell psutil und PySide6.
- Der aktuelle Startpfad python -m netmon laeuft aber ueber netmon/main.py und netmon/app.py, die auf tkinter setzen.
- Pillow wird im Startpfad fuer Bildfunktionen verwendet.
- requests ist derzeit nicht hart erforderlich, weil der Online-Service auf urllib zurueckfallen kann.
- PySide6 wirkt nach heutigem Stand nicht wie eine Kernabhaengigkeit des Debian-GUI-Starts von netmon.

Konsequenz fuer den Builder

Der aktuelle Builder in scripts/build_netmon_deb.sh sollte von einer Datei-fuer-Datei-Auswahl auf eine bewusst gepflegte Paketliste umgestellt werden.

Stand 1.1.18:

- scripts/build_netmon_deb.sh liefert bereits eine explizite Datei-/Ordnerliste.
- udisks2 und cryptsetup sind fuer den installierten LUKS-/Mount-Pfad als Depends hinterlegt.
- Die AppStream-Datei packaging/netmon.metainfo.xml wird mit ausgeliefert, damit Software-Center die Lizenz korrekt erkennen.
- Privater Laufzeitcode liegt Debian-konform unter /usr/lib/netmon; gemeinsame Assets liegen unter /usr/share/netmon.
- Paketdokumentation, Copyright und komprimierter Changelog liegen unter /usr/share/doc/netmon.

- Manpages, md5sums, normalisierte Dateirechte und ein auf 256 x 256 skaliertes Hicolor-Icon werden beim Build erzeugt.

Empfohlene Richtung:

1. Vollstaendige Laufzeit-Unterpakete config, core, features, lang, modules, services, ui uebernehmen.
2. Die benoetigten Top-Level-Dateien aus netmon/ explizit kopieren.
3. Assets getrennt unter /usr/share/netmon/assets/ ablegen.
4. Depends, Recommends und Suggests aus der aktuellen Feature-Registry in netmon/app.py ableiten.
5. Tests und Repo-Ballast explizit ausschliessen, statt implizit wegzulassen.

Festgelegte Entscheidungen

- Code-the-Classics.zip wird standardmaessig im Paket enthalten sein.
- projectM-Presets werden offiziell mit ausgeliefert.
- Konfigurations-Defaults bleiben direkt im Paket unter netmon/config/.
- Es wird nur netmon als Produktpaket gepflegt.
- netmon.png ist das Hauptprogramm-Icon.
- netmon-radio.png ist das Radio-spezifische Icon.

Deinstaller-Stand

Der aktuelle Paketstand liefert netmon-uninstall als eigenen Launcher und CLI-Weg mit aus.

Der Deinstaller fragt optional per y/n ab:

- Paket entfernen
- Launcher-/Menue-Reste entfernen
- Benutzerdaten und Dokumente entfernen
- installierte Secure Browser separat entfernen oder standardmaessig behalten
- NetMon-VPN-/WireGuard-Reste entfernen
- alle Profile unter /etc/wireguard auf Testsystemen entfernen
- NetMon-NetworkManager-WireGuard-Connections entfernen
- DNS-/DoT-/NetworkManager-Reste zuruecksetzen
- UFW zuruecksetzen
- optionale Test-Abhaengigkeiten purgen

Hinweis:

- Die aggressive Option fuer /etc/wireguard ist bewusst nur als Testsystem-Option gedacht.
- Die normale NetMon-VPN-Bereinigung versucht weiterhin, NetMon-bezogene Reste gezielt statt pauschal zu entfernen.

netmon-dark

netmon-dark sollte nicht als separates Debian-Paket weitergefuehrt werden.

Begruendung:

- Der Dark Mode ist bereits im Hauptprogramm eingebaut.
- In netmon/app.py sind LIGHT_THEME, DARK_THEME, BEIGE_THEME und POP_THEME direkt Teil von THEMES.
- netmon/main.py unterstuetzt bereits --theme.
- scripts/build_netmon_dark_deb.sh startet nur einen separaten Test-Launcher python3 -m netmon.dark_test.

Konsequenz:

- netmon-dark ist als historischer Test-/Migrationspfad zu behandeln.
- Fuer die Debian-Auslieferung sollte nur netmon gebaut werden.
- Falls gewuenscht, koennen wir den separaten netmon-dark-Builder spaeter entfernen oder archivieren.

CHANGELOG

Quelle: docs/CHANGELOG.md

Changelog

Automatisch aus der Git-Historie des Repositories erzeugt.

2026-07-18 - Version 1.1.19

- DNS-Startup-Fallback und Startup-Reparatur fuer verwaiste Killswitch-Zustaende robuster gemacht.
- Nach einem erfolgreichen VPN-Profilwechsel synchronisiert NetMon den UFW-Sollzustand jetzt auch ohne aktiven Killswitch wieder mit den referenzierten VPN-Profilen.
- Das losgeloeste Standalone-Radio startet nur noch im Darkmode; die Theme-Umschaltung im externen Radio wurde entfernt.
- Der Speedtest-Bereich ordnet Server-ID, Speichern, Server-IDs abrufen, Speedtest starten und Speedtest-Verlauf klar vertikal an.
- Die Linux-Hilfe besitzt jetzt einen PDF-Prompt-Workflow: Eintraege koennen per Rechtsklick als normaler Prompt oder als Als Prompt mit PDF kopiert werden.
- PDF-Dokumentationen werden nicht mehr lokal erzeugt, sondern extern erstellt/heruntergeladen und anschliessend in der unteren Linux-Hilfe-Bibliothek importiert.
- Die Linux-Hilfe-Bibliothek kann importierte PDFs anzeigen, per Doppelklick oeffnen, den gespeicherten Prompt kopieren, den Dokumentenordner oeffnen und PDFs inklusive Metadaten per Rechtsklick loeschen.

2026-07-17 - Version 1.1.18

- Der Speedtest-Bereich kann verfuegbare Server-IDs abrufen und bis zu 20 Treffer mit Anbieter, Ort, Land und Entfernung im NetMon-Log anzeigen.
- Die Serverabfrage unterstuetzt sowohl das offizielle Ookla-CLI als auch speedtest-cli, laeuft im Hintergrund und veraendert die gespeicherte Server-ID nicht automatisch.
- Der ClamAV-Schnellscan besitzt frei konfigurierbare Standard- und eigene Scanordner; die Auswahl wird dauerhaft in den NetMon-Einstellungen gespeichert.
- Die ClamAV-Anzeige zaehlt die Laufzeit ab Scanstart weiter und stellt Abbruch, Abschlussstatus und technische Fehlerausgaben klarer dar.
- Schnellscan-Ausschluesse fuer Cache, Papierkorb, Steam und Snap vermeiden unnoetig grosse Standardscans; ohne gueltige Zielordner startet kein stiller Home-Komplettscan.
- Benutzerbezogene ClamAV-Signaturen liegen XDG-konform unter ~/.local/share/netmon/clamav-db beziehungsweise XDG_DATA_HOME/netmon/clamav-db; Eigentuemerschaft und restriktive Rechte werden vor der Nutzung geprueft.
- Neue und geaenderte ClamAV-Oberflaechentexte sind ueber deutsche und englische Sprachschluessel angebunden.
- RKHunter-Warnungen werden als erwartete Systemaenderung, pruefenswert oder kritisch klassifiziert; bekannte Entwicklerumgebungen und typische Konfigurationsaenderungen werden nicht mehr pauschal als Sicherheitsproblem dargestellt.

- Die Analyseausgabe enthaelt immer Kurzbewertung, relevante Auffaelligkeiten, Risiko, konkrete Handlungsempfehlungen und die Einordnung typischer Fehlalarme.
- Ein vollstaendiger RKHunter-Scan wird um dpkg -V als zusaetzliche Vertrauensevidenz ergaenzt.
- rkhunter --propupd wird nur bei niedrigem Risiko, erfolgreicher Paketpruefung und ohne kritische oder ungeklaerte Befunde angeboten.
- Das Referenzupdate erfordert immer eine ausdrueckliche Bestaetigung und eine erneute Gate-Pruefung; es wird niemals automatisch ausgefuehrt.

2026-07-11 - Version 1.1.17

- Rechter Link-Slot-Bereich von 15 auf 20 frei beschreibbare interne oder externe Links erweitert; die fuenf neuen Buttons werden in Originalgroesse unten angehaengt und verschieben VPN-Slots sowie Security-Systemcheck entsprechend nach unten.
- Bestehende Konfigurationen mit 15 Link-Slots bleiben erhalten und werden beim Laden automatisch um die leeren Slots 16 bis 20 ergaenzt.
- Rechter VPN-Slot-Bereich von 5 auf 10 Slots erweitert; die neuen Slots 6 bis 10 werden in Originalgroesse unten angehaengt und verschieben nachfolgende Bereiche entsprechend nach unten.
- Bestehende VPN-Slot-Konfigurationen bleiben erhalten und werden beim Laden automatisch um die leeren Slots 6 bis 10 ergaenzt.
- Aktualisierter Paketbuild mit unveraenderter Versionsnummer 1.1.17 fuer die nachgezogenen VPN-, Status- und Tour-Korrekturen.
- VPN-Profile lassen sich im VPN-Tab wieder direkt aus der markierten Tabellenzeile loeschen, auch wenn das Profilfeld im Formular leer ist.
- Beim Loeschen eines WireGuard-Profils werden zugehoerige normale VPN-Slots, primaere VPN-Slots und die Privacy-VPN-Auswahl konsistent bereinigt.
- VPN-Slots starten vorhandene Profile auch dann, wenn die zwischengespeicherte Profilliste noch nicht aktualisiert wurde.
- Die VPN-Ampel aktualisiert nach einem erfolgreichen Profilwechsel die oeffentliche IP und das Exit-Land automatisch; der normale 60-Sekunden-Cache blockiert diesen Wechsel-Refresh nicht mehr.
- Der Secure-Browser-Tab ist als eigene deutsche und englische Tour im Setup verfuegbar und wird automatisch in die volle Tour aufgenommen.
- Die Secure-Browser-Tour erklaert Profiltrennung, Browser-Hardening, Domain-Whitelist, Cookie-Persistenz, Dry-Run und die Grenzen gegenueber IP-Sichtbarkeit, Fingerprinting und Kontoerkennung.
- Neue Secure-Browser-Instanzen werden XDG-konform unter Dokumente/NetMon/Secure Browser angelegt; der Zielordner entsteht bei Bedarf automatisch und harte benutzerspezifische Pfade wurden entfernt.
- Die nach abgeschlossener Browser-Migration nicht mehr benoetigte Legacy-Integration von ~/Programme wurde aus Inventur, Verwaltung, Quellensuche und Deinstaller entfernt; Secure Browser verwenden ausschliesslich Dokumente/NetMon/Secure Browser.
- Der Deinstaller fragt Secure Browser separat mit der sicheren Vorgabe [y/N] ab und schuetzt sie bei N auch vor der allgemeinen NetMon-Dokumentenbereinigung.
- Secure-Browser-Menueeintraege zeigen nach Erstellung, Einbindung oder Aktualisierung nur noch den eigentlichen Namen wie GitHub oder Sparkasse Ulm; das stoerende Praefix NetMon Secure Browser bleibt

auf technische Dateinamen und interne Kennungen beschraenkt.

- Die Secure-Browser-Aktualisierung behaelt vorhandene lokale PNG-/ICO-Symbole bei fehlgeschlagenem Favicon-Download und bevorzugt kompatible Raster-Icons vor SVG-Dateien; dadurch bleiben unter anderem Chad- und cPanel-Menueicons erhalten.
- Der mittlere Radio-/Uhrbereich im Kopf erhaelt eine groessere feste Hoehenreserve, damit die LCD-Uhr auch mit dem aktuellen Theme vollstaendig oberhalb der Haupttabs sichtbar bleibt.
- Mitgelieferte Log-Pics werden bei der Paketinstallation unter Documents/NetMon/Log-Pics beziehungsweise Dokumente/NetMon/Log-Pics vorbereitet.
- Debian-Paketlayout mit Lintian geprueft und auf /usr/lib/netmon, /usr/share/netmon sowie /usr/share/doc/netmon standardisiert.
- Paket um Debian-Copyright, komprimierten Changelog, Manpages und md5sums ergaenzt; Dateirechte, Icon-Groesse und Paketbeschreibung korrigiert.
- Deutsche und englische Sprachzuordnungen systematisch bereinigt; deutsche Presets zeigen in Firewall, PATH Guard, Lynis, Repository Guard und weiteren Security-Bereichen keine englischen Standardtexte mehr an.
- Security-Untertabs neu und kompakt benannt: Devices, Firewall, Path, Prozesse/Processes, Repos, Roots, Autostart, Rootkithunter und Capture.
- PATH Guard aus dem bisherigen Device-/Guard-Bereich in einen eigenen Security-Tab mit eigener Detailansicht verschoben; Ampel-, Menue-, Hilfe- und Tour-Navigation entsprechend angepasst.
- PATH-Guard-Berichte, Risiken, Schweregrade, Hinweise und Shell-Profilbefunde vollstaendig sprachabhaengig formatiert.
- Firewall-Bewertungen und Listener-Hinweise im deutschen Preset uebersetzt, englische Ausgaben bleiben ueber die englischen Sprachschluessel verfuegbar.
- Lynis-Status, Scanbezeichnungen, Abschlussmeldungen, Hinweise und Ergebnisplatzhalter fuer beide Sprachen konsistent zugeordnet.
- Repository-Guard-Oberflaeche, Berichte, Detailansichten, Filter, Vertrauensstatus und Befundtexte sprachabhaengig vereinheitlicht.
- D-Scan um Pfad kopieren erweitert; der Temp-Ordner-Pfad kann direkt in die Zwischenablage uebernommen werden. Die zugehoerigen Aktionen sind in zwei funktionale Button-Reihen gegliedert.
- Laufzeit- und Netzwerk-Debug-Logs verwenden jetzt vollstaendige Zeitstempel mit Datum, damit Eintraege ueber Tages- und Stundenwechsel chronologisch eindeutig bleiben.
- Sprach-, Controller-, Routing- und UI-Tests fuer die korrigierten Ausgaben und neuen Bedienwege erweitert.
- Paket-, App-, Dokumentations- und Downloadstand auf 1.1.17 angehoben.

2026-07-07 - Version 1.1.16

- Release-Artefakte fuer den aktuellen Stand erneut gebaut.
- Website- und Download-Verweise des Security-Audits auf Runde 6 aktualisiert.
- Site-Bundle erzeugt jetzt den aktuellen Audit-Alias security-audit-r6.html und liefert netmon_security_audit_2026_r6.pdf mit aus.

2026-07-05 - Version 1.1.16

- Debian-Paketversion auf 1.1.16 angehoben.
- Release-Skripte, Doku-Build und Website-Downloadverweise auf 1.1.16 aktualisiert.
- Secure-Browser-Integration und Sprachschluessel in den aktuellen Paket- und Dokumentationsstand uebernommen.

2026-06-29 - Version 1.1.15

- Debian-Paketversion auf 1.1.15 angehoben, um den stabilen Recovery-/Backup-Stand neu auszurollen.
- Website-, Download- und Build-Referenzen auf 1.1.15 umgestellt.
- Produktdoku und Downloadseiten verweisen jetzt auf das aktuelle Sicherheits-Audit Runde 4 fuer 1.1.15.
- Defense-in-Depth nachgezogen: WireGuard-Import warnt jetzt vor Hook-Direktiven wie PostUp/PreDown.

2026-06-28 - Version 1.1.14

- rkhunter-WEB_CMD auf trusted Binary-Pfade begrenzt; PATH-Hijacking ueber user-kontrollierte wget/curl-Treffer wird verworfen.
- Agent-Runner begrenzt black-, ruff-, pytest- und ls-Pfadargumente jetzt auf das Repository.
- Fun-Launcher startet nur noch validierte Python-Launcher innerhalb des verwalteten Spieleordners.
- Radio-Station-URLs aus externen APIs auf erlaubte Stream-Schemas begrenzt (http, https, rtsp, rtmp).
- Unerlaubte Stations-URLs werden bereits bei der Normalisierung verworfen und nicht mehr an mpv oder ICY-Metadatenabrufe weitergereicht.
- Website-Branding auf NetMonRoots vereinheitlicht und Header mit Banner-Icon erweitert.
- Download-, SEO- und Release-Verweise auf Paketversion 1.1.14 angehoben.
- Website-Build erwartet jetzt netmon_1.1.14_all.deb und erzeugt passende Release-Hashes.
- Aktuelles / Updates als zentrale Statusseiten fuer Freigaben, Testlabor-Hinweise und bekannte Themen eingefuehrt.
- About-Dialog prueft online gegen latest-version.json und verweist bei neueren Website-Releases auf die Downloadseite.
- LUKS-Mount-Pfad fuer Dev-Lab und installierte Pakete angeglichen; Paketabhaengigkeiten fuer udisks2 und cryptsetup geschaerft.
- Support-Bereich um sichtbaren Datenschutzhinweis vor dem Versand von Diagnosepaketen erweitert.

2026-06-21 - Version 1.1.13

- Speedtest um konfigurierbare Server-ID erweitert.
- Speedtest zeigt jetzt zusätzlich Servername und Server-ID im Ergebnisbereich an.
- Speedtest-History und Debug-Ausgaben enthalten jetzt auch den verwendeten Server.
- WireGuard-Endpoint-Verarbeitung auf feste IP-Adressen verschärft.
- Hostnamen in WireGuard-Endpoints werden für Killswitch-/UFW-Handshake-Regeln nicht mehr stillschweigend übernommen.
- NetMon zeigt jetzt einen Hinweis, wenn ein WireGuard-Profil Hostnamen statt fester Endpoint-IP verwendet.

- ClamAV-Quarantäne- und Temp-Verzeichnisse gegen Symlink-Missbrauch gehärtet.
- AI-Bildimport (image_to_data_url) auf erlaubte Bildtypen und Pfade im Benutzer-Homeverzeichnis begrenzt.
- Deinstaller entschärft: Paket- und optionale Abhängigkeitsentfernung nutzt jetzt apt remove statt apt purge.
- Deinstaller-Texte und Rückmeldungen an die entschärfte Remove-Strategie angepasst.

2026-06-14 - Version 1.1.10

- Debian-Paketierung bis 1.1.10 konsolidiert.
- Paket enthaelt jetzt den aktuellen Deinstaller netmon-uninstall inklusive Desktop-Eintrag.
- Deinstaller bereinigt optional:
- NetMon-Paket
- Launcher-/Menue-Reste
- Benutzerdaten, Dokumente und Caches
- NetMon-VPN-/WireGuard-Reste
- referenzierte NetworkManager-WireGuard-Connections
- optional alle Profile unter /etc/wireguard fuer Testsysteme
- DNS-/DoT-/NetworkManager-Overrides
- optionale Test-Abhaengigkeiten
- Setup-Tab erweitert um Systemwartungsbuttons fuer:
- apt update
- apt upgrade -y
- apt autoremove -y
- apt clean
- Menue-Refresh
- Menue- und Icon-Cache-Refresh
- Security-Haertung nachgezogen:
- WireGuard-Profilpfade gegen Path-Traversal abgesichert
- Root-Datei-Helper in app.py und rkhunter_manager.py auf Allowlists begrenzt
- Repo-Pfadpruefung fuer Agent-Dateilesezugriffe
- agent/runner.py-Allowlist fuer lokale Shell-Kommandos verengt
- Roots-Baseline-Guard-Root-Reads auf erlaubte Pfade begrenzt
- Startup-Guard robuster gemacht:
- Worker-Ausnahmen blockieren den Guard nicht mehr still
- Fehlerpfade setzen Status und UI sauber zurueck
- DNSSEC-Ausgabe bereinigt:

• konfigurierter Modus und echter Validierungstest werden getrennt dargestellt

• VPN-Profilliste korrigiert:

• NetMon mischt fuer WireGuard-Profile nicht mehr beliebige NM-vpn-Connections ein,

sondern nur noch echte NM-wireguard-Connections.

• 2026-03-07 5ceb443e Initial commit

• 2026-03-07 4429bb26 Set up dev container workspace

• 2026-03-07 f7b0619d Format code with black

• 2026-03-07 82b539a0 Fix pytest import path and black target version

• 2026-03-07 6656618b Add makefile and VS Code settings

• 2026-03-07 27a62cf4 Agent core: tools + runner + netmon integration

• 2026-03-08 725a74ba netmon: fix layout-related indentation and traceroute helper structure

• 2026-05-02 e334324a refactor(netmon): extract phase 1 startup and core helpers

• 2026-05-02 e9239f8f refactor(netmon): extract phase 2 ui widget helpers

• 2026-05-02 85a56da6 refactor(netmon): continue phase 4 controller extraction