



■ INTERNES HANDBUCH · DUG-IT NETMONITOR

Das komplette Handbuch zu NetMonitor

Wie das Drei-Skript-System aufgebaut ist, wie die Installation beim Kunden abläuft, wie die Auswertung mit Charts und Eventlog-Korrelation funktioniert — und wie NetMonitor genau wie WinPrep über dl.dug-it.de beim Kunden zum Einsatz kommt.

Skripte `NetMonitor.ps1`, `Monitor-Worker.ps1`, `Analyze-NetMonitor.ps1` Version 2.9

Stand 2026-09-23 Umfang 3 Skripte, 9 Kapitel

01 Überblick & Architektur	02 Installation (Discovery-Assistent)
03 Bedienung & Hauptmenü	04 Laufzeit-Artefakte
05 Auswertung mit dem Analyzer	06 Eventlog-Korrelation
07 Deinstallation	08 Komfortabler Einsatz beim Kunden
09 Anhang: Ordner, Troubleshooting & Release	

01 Überblick & Architektur

Was NetMonitor macht, woher es kommt und wie die drei Skripte zusammenspielen.

NetMonitor ist ein kompaktes PowerShell-Monitoring-Paket von DUG-iT zur Diagnose sporadisch hängender Netzwerk-Anwendungen auf Windows-Clients — entstanden aus einem realen Fall: Solutio/Charly-Freezes in einer Zahnarztpraxis, deren Ursache eine defekte Netzwerkleitung war, nachgewiesen über burstweise Receive-Discards. Zielgruppe sind MSP-Techniker: Das Tool wird beim Kunden installiert, sammelt über Tage automatisch Daten, und

der Techniker wertet sie später mit dem Analyzer aus — als Konsolen-Report vor Ort oder als HTML-Report für die Kundendokumentation.

Die drei Skripte

NetMonitor.ps1	Orchestrator: interaktives Menü, Discovery-Assistent für die Ersteinrichtung, install/start/stop/status/log/config/uninstall.
Monitor-Worker.ps1	Läuft dauerhaft im Hintergrund (sichtbares, nicht schließbares Fenster): misst SMB-Latenz, TCP-Connect-Zeiten, Ping, Retransmits und Link-Änderungen, schreibt bei Schwellwert-Überschreitung sofort einen Diagnose-Snapshot.
Analyze-NetMonitor.ps1	Reiner Offline-Auswerter, läuft beim Techniker, nicht beim Kunden: liest die gesammelten Dateien, zeigt Konsolen-Report oder erzeugt HTML-Report mit Zeitreihen-Charts, optional Vergleich zweier Systeme und Eventlog-Korrelation.

Datenfluss

Der Discovery-Assistent (Kapitel 02) erzeugt eine `config.json` mit der zu überwachenden Anwendung, den TCP-Zielen und Schwellwerten. `Monitor-Worker.ps1` liest sie beim Start und schreibt fortlaufend drei Datenquellen: PerfMon-Rohdaten per `logman`, eine eigene 30-Sekunden-Messreihe für TCP/Ping/Retransmits und ein Ereignis-Log mit Diagnose-Snapshots. `Analyze-NetMonitor.ps1` liest all das aus einem Ordner ein (Kapitel 05) — typischerweise, nachdem der Techniker die Dateien vom Kunden-PC kopiert hat.

- i** Genau wie [WinPrep](#) läuft NetMonitor sowohl direkt lokal als auch über den gemeinsamen DUG-iT-Tool-Launcher (`dl.dug-it.de/wp.ps1` , Kapitel 08) — gleiches Deployment-Prinzip, gleicher Server, gleiche Optik.

02 Installation (Discovery-Assistent)

Menüpunkt 6 — ein geführter Assistent, der die zu überwachende Anwendung anhand ihrer aktiven Netzwerkverbindungen selbst findet, statt Prozessnamen von Hand einzutragen.

1

Prozess wählen: analysiert alle Prozesse mit aktiven TCP-Verbindungen (Systemprozesse wie `svchost` / `lsass` ausgeblendet) und zeigt sie mit Verbindungszahl und Zielen zur

Auswahl.

- 2 **Verwandte Prozesse:** schlägt Prozesse mit ähnlichem Namensstamm vor (z. B. `SolutioHelper` neben `Solutio`) und fragt, ob sie mit überwacht werden sollen.
Standard: ja (Enter).
- 3 **TCP-Ziele wählen:** listet alle Verbindungsziele der Anwendung, markiert LAN- getrennt von WAN-Zielen und empfiehlt, nur LAN-Ziele zu tracken (WAN-Latenz schwankt naturgemäß).
- 4 **SMB-Latenz:** falls aktive SMB-Verbindungen erkannt werden, optional mit überwachen.
- 5 **Schwellwerte:** TCP-Latenz (Standard 300 ms) und, falls SMB aktiv, SMB-Latenz (Standard 500 ms).
- 6 **Anzeigename:** für Reports und Fensterbanner — Standard ist der erkannte Prozessname.

Danach richtet `Install-Monitor` automatisch ein: `config.json` speichern, Autostart (Run-Key), PerfMon-Collector (SMB/NIC/TCP/CPU/RAM, DE/EN-Counter-Namen automatisch aufgelöst), Watchdog-Scheduled-Task (startet den Worker alle 5 Minuten neu, falls er nicht läuft), PerfMon-Heal-Scheduled-Task (siehe Callout unten) und drei Desktop-Buttons für den angemeldeten Benutzer.

! Die Installation braucht **Administratorrechte** (Registry unter `HKLM` , PerfMon-Collector, geplanter Task). Der Worker selbst läuft danach bewusst nur als angemeldeter Benutzer, nicht im SYSTEM-Kontext — der Scheduled Task für den Watchdog wird deshalb gezielt mit `/RL LIMITED` für den ermittelten interaktiven Benutzer angelegt.

i Ab Version 2.6: `logman` -Operationen auf dem als SYSTEM angelegten Collector scheitern **ohne Adminrechte grundsätzlich** mit "Zugriff verweigert" — das gilt auch für die reine Statusabfrage, nicht nur für den Start (echter Kundenfall: ein nach einem Reboot gestorbener Collector konnte vom unprivilegierten Worker nie selbst repariert werden). Seit V2.6 löst `Install-Monitor` das zweigleisig: ein zusätzlicher Scheduled Task `NetMonitor_PerfMonHeal` läuft als **SYSTEM** und hält den Collector alle 5 Minuten am Laufen (übersteht jeden Reboot, unabhängig vom angemeldeten Benutzer), und der angemeldete Benutzer wird automatisch der eingebauten Gruppe "Performance Log Users" hinzugefügt, damit auch der Worker selbst den Collector-Status lesen kann — wirkt erst nach dem nächsten Ab-/Anmelden. **Ab V2.9:** ein blosser Start allein reichte bei einem echten Kunden über 8 Stunden hinweg nicht aus (Verdacht:

der Sicherheitsdeskriptor der unter dem Administrator-Kontext angelegten Collector-Definition räumt SYSTEM keine ausreichenden Rechte ein — unabhängig von normalen Dateisystem-Rechten). Der Heal-Task ruft deshalb `PerfMonHeal.ps1` auf (generiert wie `Watchdog.ps1`): erst ein leichter Start, und falls das nicht reicht, wird die Collector-Definition komplett neu angelegt — diesmal von SYSTEM selbst erstellt. Jeder Versuch wird als `PERFMON-HEAL / PERFMON-HEAL-FEHLER` nach `hang-snapshots.log` protokolliert, damit sich der Zustand auch ohne Zugriff auf den Kunden-PC nachvollziehen lässt.

03 Bedienung & Hauptmenü

Nach der Ersteinrichtung reicht das Hauptmenü für den Alltag — genau wie bei WinPrep bleibt es nach einer Aktion offen, bis `Q` gedrückt wird.

```
● NetMonitor.ps1 - Hauptmenü

+-----+
| NetMonitor V2.9                                     |
+-----+

-- Betrieb --
  1) Starten
  2) Stoppen
  3) Status aktualisieren

-- Diagnose --
  4) Log anzeigen
  5) Konfiguration anzeigen

-- Einrichtung --
  6) Installieren / Neu konfigurieren
  7) Deinstallieren

  Q) Beenden
```

TASTE	AKTION	WIRKUNG
1	Starten	Startet den Worker (falls nicht aktiv) und den PerfMon-Collector, aktiviert den Watchdog-Task.

TASTE	AKTION	WIRKUNG
2	Stoppen	Deaktiviert zuerst den Watchdog (sonst startet er sofort neu), beendet dann Worker und PerfMon.
3	Status aktualisieren	Der Status steht ohnehin immer oben im Menü — dieser Punkt lädt ihn einfach neu.
4	Log anzeigen	Letzte 80 Zeilen aus <code>hang-snapshots_<PC>.log</code> .
5	Konfiguration anzeigen	Rohinhalt der <code>config.json</code> .
6	Installieren / Neu konfigurieren	Startet den Discovery-Assistenten erneut (Kapitel 02) — überschreibt die bestehende Konfiguration.
7	Deinstallieren	Siehe Kapitel 07.
Q	Beenden	Ohne etwas auszuführen.

Für Automatisierung/RMM steht jeder Menüpunkt auch direkt als Parameter zur Verfügung, ohne durchs Menü zu müssen: `.\NetMonitor.ps1 start , stop , status , log , config , install , uninstall`.

Freeze melden (Desktop-Button)

Zusätzlich zu `NetMonitor-Start.cmd` / `NetMonitor-Stop.cmd` legt die Installation einen dritten Desktop-Button an: `NetMonitor-FreezeMelden.cmd`. Der Anwender klickt ihn, sobald er selbst einen Hänger bemerkt — das legt eine Trigger-Datei ab, die der Worker in seiner Hauptschleife erkennt und sofort einen Snapshot mit der Kennzeichnung `MANUELL GEMELDET` erstellt, unabhängig von Schwellwerten oder dem sonst geltenden 60-Sekunden-Cooldown zwischen automatischen Snapshots.

04 Laufzeit-Artefakte

Alle Dateien, die NetMonitor auf dem Kunden-PC anlegt — diese kopiert der Techniker für die Auswertung (Kapitel 05).

Ab Version 2.4 liegen alle gesammelten Messdaten in einem eigenen Unterordner `Messdaten\`, getrennt von Skripten und Setup-Dateien im Installationsordner. Das hält den Installationsordner übersichtlich und sorgt dafür, dass `Uninstall-Monitor` die Messdaten nicht versehentlich mit löschen kann. `Analyze-NetMonitor.ps1` erkennt den Unterordner automatisch, fällt aber bei älteren, flach abgelegten Ordnern (vor V2.4) weiterhin darauf zurück — der Techniker muss beim Auswählen des Ordners nichts beachten.

DATEI	INHALT
<code>Messdaten\netmon-perflog_<PC>*.csv</code>	PerfMon-Rohdaten (<code>logman</code> , 5-Sekunden-Intervall, cp1252-kodiert, US-Datumsformat).
<code>Messdaten\tcp-latenz_<PC>.csv</code>	Eigene 30-Sekunden-Messreihe: Zeit, TCP-Ziele, Ping, Retransmits.
<code>Messdaten\hang-snapshots_<PC>.log</code>	Diagnose-Snapshots, App-Start/Stopp-Ereignisse, Link-Änderungen, Neustart-Korrelation (siehe unten).
<code>Messdaten\sysinfo_<PC>.json</code>	OS/Hardware/NIC-Details, bei jedem Worker-Start aktualisiert.
<code>heartbeat.txt</code>	Lebenszeichen des Workers, alle 3 Sekunden überschrieben. Bleibt im Installationsordner (operativ, keine Diagnosedaten).
<code>freeze-trigger.txt</code>	Kurzlebig: vom Freeze-Melden-Button angelegt, vom Worker sofort verarbeitet und wieder gelöscht. Bleibt ebenfalls im Installationsordner.
<code>config.json</code>	Vom Discovery-Assistenten erzeugte Konfiguration (Kapitel 02).
<code>Watchdog.ps1</code> / <code>PerfMonHeal.ps1</code> / <code>counters.txt</code>	Bei Installation generierte Hilfsdateien für die beiden Scheduled Tasks bzw. PerfMon-Collector.

i Neustart-Korrelation: Wird die überwachte App nach einem **kurzen** Neustart (Lücke ≤ 10 Minuten zwischen Beenden und Neustart — typisches Absturz-/Hänger-Muster, nicht das normale Feierabend-Beenden) wieder gestartet, schreibt der Worker automatisch die bereits erfassten Netzwerkwerte der letzten 5 Minuten vor dem Beenden zusätzlich in `hang-snapshots_<PC>.log` — so lässt sich ein Zusammenhang zwischen Netzwerkproblemen und App-Neustarts erkennen.

i Ab Version 2.8: jeder Diagnose-Snapshot enthält zusätzlich einen systemweiten Schnappschuss aller gerade wartenden/scheiternden TCP-Verbindungsversuche (`SYN_SENT`) mit Zielport und Prozess — nicht nur auf die überwachte App beschränkt. Echter Kundenfall: ein **lokaler** Begleitdienst der Praxissoftware auf `127.0.0.1:8085` fehlte, die App versuchte dauerhaft erfolglos dorthin zu verbinden. Solche Versuche sind extrem kurzlebig und per Hand kaum einzufangen — im Moment eines Hangs (wenn die App vermutlich selbst gerade blockiert) stehen die Chancen am besten, es automatisch mitzuprotokollieren.

i Alle pro-PC benannten Dateien können gefahrlos in einen gemeinsamen Ordner kopiert werden, wenn mehrere Systeme verglichen werden sollen (Kapitel 05) — der PC-Name im Dateinamen verhindert Überschreiben.

05 Auswertung mit dem Analyzer

`Analyze-NetMonitor.ps1` läuft beim Techniker, nicht beim Kunden — interaktiver Assistent oder direkt mit Parametern.

● Windows PowerShell

```
PS> .\Analyze-NetMonitor.ps1
# Interaktiver Assistent: Ordner, optionaler Vergleich, Zeitfilter, HTML-Report, Eve

PS> .\Analyze-NetMonitor.ps1 -Path C:\Kunde-A -CompareWith C:\Kunde-B -Html
# Direkt zwei Systeme vergleichen und HTML-Report erzeugen
```

Der Konsolen-Report zeigt Paketverluste (mit Bursts und Korrelation zu App-Neustarts), TCP-Retransmits, SMB-Latenz, TCP-Connect-Latenz mit Timeouts, App-Ereignisse und Link-Änderungen — farblich nach den Referenzwerten unten bewertet. Mit `-Html` entsteht zusätzlich ein eigenständiger HTML-Report mit Systeminformationen, denselben Messwerten

inklusive Erläuterung, und bei zwei Systemen einer Gegenüberstellung mit automatischem Fazit.

Referenzwerte

MESSWERT	GRÜN	ORANGE	ROT
Paketverluste / Stunde	< 1	1–3	> 3
Discard-Burst (Einzelereignis)	+1	+2	≥ +3
NIC-Fehler	0	–	≥ 1
SMB Read/Write Latenz	< 20 ms	20–100 ms	> 100 ms
TCP-Connect (LAN)	< 100 ms	100–300 ms	> 300 ms / Timeout
Ping (LAN)	< 5 ms	5–20 ms	> 20 ms / Verlust
Link-Änderungen im Betrieb	0	–	≥ 1

Zeitreihen-Charts im HTML-Report

Der HTML-Report enthält reine Inline-SVG-Diagramme (keine externen Libraries) für Paketverluste kumulativ, Ping-Laufzeit und TCP-Connect-Latenz je Ziel. Bei Einzelauswertung eine Linie pro Metrik, bei Vergleich zweier Systeme ein gemeinsames Diagramm mit Legende statt zwei getrennter — System A in DUG-iT-Rot, System B in Grau, dieselbe Farbsprache wie das Logo.

Zeitraum eingrenzen & direkte Parameter

PARAMETER	BEDEUTUNG
-Path	Ordner mit den Messdaten eines Systems.
-CompareWith	Zweiter Ordner für den Direktvergleich.
-From / -To	Zeitfilter, z. B. "2026-07-16 15:00" .

PARAMETER	BEDEUTUNG
-Html	Zusätzlich HTML-Report erzeugen.
-EventLog	Eventlog-Korrelation aktivieren, siehe Kapitel 06.

06 Eventlog-Korrelation

Optionaler Analyzer-Schritt: gleicht Windows-eigene Application-Hang-Events zeitlich mit Paketverlusten und App-Neustarts ab.

Windows protokolliert selbst, wenn eine Anwendung nicht mehr auf Nachrichten reagiert (Provider `Application Hang`, Event-ID 1002, Application-Log). Mit `-EventLog local` fragt der Analyzer das lokale System ab, mit `-EventLog C:\export.evtx` eine zuvor exportierte Logdatei (z. B. per `wevtutil epl Application export.evtx` vom Kunden-PC mitgenommen). Treffer innerhalb von ± 5 Minuten zu einem Discard-Burst oder einem App-Neustart werden als Korrelation ausgewiesen — in Konsole und HTML.

Windows PowerShell

```
PS> .\Analyze-NetMonitor.ps1 -Path C:\Kunde-A -EventLog local -Html
```

i Der Provider-Name `Application Hang` ist — anders als PerfMon-Counter-Namen — nicht lokalisiert und auf deutschen wie englischen Windows-Systemen identisch. Nur die *Meldungs-Anzeige* ist lokalisiert, der Prozessname wird deshalb bevorzugt aus den rohen Event-Properties gelesen, nicht aus dem übersetzten Text.

07 Deinstallation

Menüpunkt 7 — entfernt die Einrichtung vollständig, lässt aber bewusst die gesammelten Diagnose-Daten stehen.

Entfernt Watchdog-Task, PerfMon-Heal-Task (ab V2.6), die bei der Installation vergebene Mitgliedschaft in "Performance Log Users", Autostart-Eintrag, PerfMon-Collector (inkl. Freigabe eventuell gesperrter CSV-Dateien) und alle drei Desktop-Buttons. Anschließend

werden nur die reinen Setup-/Betriebsdateien gelöscht: `config.json`, `counters.txt`, `Watchdog.ps1`, `PerfMonHeal.ps1`, `heartbeat.txt`, `freeze-trigger.txt`.

! Der Unterordner `Messdaten\` mit allen gesammelten Diagnose-Daten bleibt **bewusst erhalten** — der Techniker braucht sie ja meist erst *nach* der Deinstallation für die Auswertung. Erst danach von Hand löschen, falls gewünscht.

08 Komfortabler Einsatz beim Kunden

Genau dieselbe Infrastruktur wie WinPrep — derselbe Server, derselbe Launcher, dieselbe Handhabung.

NetMonitor liegt als eigenes GitHub-Repo (`github.com/crolj/netmonitor`) und wird wie WinPrep automatisch nach **dl.dug-it.de** veröffentlicht — spätestens 10 Minuten nach jedem `git push`, unter `/netmonitor/`.

Der Ein-Zeiler

PowerShell **als Administrator** öffnen, dann:

```
● Windows PowerShell (als Administrator)
```

```
PS> iex(irm https://dl.dug-it.de/wp.ps1)
```

Das lädt den generischen DUG-iT-Tool-Launcher (`DUGiT_Tools_Launcher.ps1`) — zeigt bei mehreren Tools (aktuell WinPrep/NetMonitor) eine nummerierte Auswahl. Direkt zu NetMonitor, ohne Auswahlschritt:

```
● Windows PowerShell (als Administrator)
```

```
PS> iex(irm https://dl.dug-it.de/netmonitor/DUGiT_NetMonitor_Bootstrap.ps1)
```

`DUGiT_NetMonitor_Bootstrap.ps1` lädt bei jedem Aufruf alle drei Skripte frisch nach `C:\DUGITInstall\NetMonitor` und startet `NetMonitor.ps1` — alle drei, nicht nur eines, da Installation/Start ohne `Monitor-Worker.ps1` im selben Ordner nicht funktionieren.

! `https://` davor ist Pflicht: ohne Schema geht `Invoke-RestMethod` erst auf Port 80, Caddy leitet per 308 auf https weiter — Windows PowerShell 5.1 folgt 308-Redirects nicht zuverlässig. Gleicher Grund, gleiche Lösung wie im [WinPrep-Handbuch](#), Kapitel 09.

Mit Parametern (z. B. für RMM)

● Windows PowerShell (als Administrator)

```
PS> iwr https://dl.dug-it.de/wp.ps1 -OutFile boot.ps1
PS> .\boot.ps1 -Tool netmonitor status
```

Handbuch unterwegs

- dl.dug-it.de/netmonitor/DUGiT_NetMonitor_Handbuch.html
- dl.dug-it.de/netmonitor/DUGiT_NetMonitor_Handbuch.pdf

09 Anhang: Ordner, Troubleshooting & Release

Nachschlagen statt lesen — Ordnerstruktur, häufige Stolpersteine, und wie ein neues Release entsteht.

Ordner im Repo

DATEI/ORDNER	ZWECK
NetMonitor.ps1	Hauptskript (Kapitel 01–04, 07).
Monitor-Worker.ps1	Hintergrundprozess (Kapitel 01, 03).
Analyze-NetMonitor.ps1	Auswertung (Kapitel 05–06).
DUGiT_NetMonitor_Bootstrap.ps1	Lädt & startet immer die aktuelle Version (Kapitel 08).
VERSION	Einzeilige Versionsnummer, Quelle für <code>build.ps1</code> und den Live-Deploy.

build.ps1	Erstellt ein versioniertes Release-Zip, siehe unten.
tests/	Pester-5-Tests für die Analyzer-Parser gegen synthetische Fixtures.

Häufige Stolpersteine

- **Kein UAC-Fenster, Konsole schließt sofort** — meist Execution-Policy oder Domänen-GPO. `Get-ExecutionPolicy -List` prüfen (v. a. `MachinePolicy / UserPolicy`).
- **Fehler beim Download** — `Test-NetConnection dl.dug-it.de -Port 443` prüft, ob der Kunden-PC den Server erreicht (Firewall/Proxy).
- **PerfMon-Collector startet nicht neu** — seit V2.6 hält der Scheduled Task `NetMonitor_PerfMonHeal` (läuft als SYSTEM) den Collector automatisch am Laufen, auch nach einem Reboot. Falls trotzdem leer: als Administrator `logman query NetMonitor` prüfen — "Zugriff verweigert" ohne Adminrechte ist normal und kein Fehler, auch bei laufendem Collector.
- **Fehler "Der Verzeichnisname ist ungültig" bei logman/schtasks** — seit V2.7 behoben: das Skript setzt jetzt ganz am Anfang selbst ein gültiges Arbeitsverzeichnis. Ursache war ein ungültiges Windows-Arbeitsverzeichnis (z.B. ein UNC-Pfad) beim Start, mit dem `CreateProcess` für native Programme grundsätzlich nicht klarkommt — typischerweise bei Fernausführung über ein RMM-Tool.
- **Analyzer zeigt falsche/verzehnfachte Werte** — sollte seit Version 2.0 nicht mehr vorkommen: die PerfMon-Zahlenwerte werden jetzt kulturunabhängig (`InvariantCulture`) geparkt, unabhängig von der Regionaleinstellung des Technikerrechners.

Release-Workflow

Für die interne Auslieferung als Zip (z. B. für ein Kundenpaket ohne Internetzugang): Version in der `VERSION` -Datei erhöhen, dann `.\build.ps1` ausführen — erzeugt `dist\NetMonitor-vX.Y.zip` mit den drei Skripten (Versionszeile automatisch gestempelt), `README.md` und optional (`-IncludeClaudeMd`) den internen Projekthinweisen. Für den laufenden Betrieb über `dl.dug-it.de` (Kapitel 08) reicht dagegen ein einfacher `git push` — der Deploy-Timer auf dem Server zieht die neue Version automatisch, ganz ohne Zip.

DUG-iT NetMonitor · internes Handbuch

Messdaten bleiben auf dem Kunden-PC, bis der Techniker sie abholt