

AllWatch

Ein winziger Prozess-Supervisor für Linux: Ziele im Vordergrund, Gesundheit über Kernel und TCP, Neustartrichtlinien pro Ziel

Handbuch

Version	0.1.0
Commit	`ae9cb8f`
Commit-Datum	24. November 2026
Lizenz	MIT

Offline-Handbuch, identisch mit den Guides des Repositorys in der angegebenen Version und dem angegebenen Commit. Der Text des Repositorys ist maßgeblich.

Inhaltsverzeichnis

Überblick **1**
- Installation auf Debian und Ubuntu . . . 1
- Installation auf Fedora, openSUSE Leap und Arch Linux . . . 1
- Aus dem Quelltext starten . . . 2
- Arbeitsweise . . . 2
- Konfiguration . . . 3
- Vertrag eines Ziels . . . 3
- Grenzen . . . 3
- Lizenz . . . 4
- Danksagung . . . 4

Den Debian-Dienst installieren und betreiben **5**
- Was das Paket installiert . . . 5
- Installieren . . . 5
- Pfade und Dateien . . . 5
- Ein Ziel deklarieren . . . 6
- Hängendes Ziel: restart oder alert . . . 6
- Dienstbefehle . . . 6
- Die Konfiguration prüfen, ohne ein Ziel zu starten . . . 6
- Ein Ziel stoppen, ohne den Dienst zu stoppen . . . 7
- Den Dienst unter einem anderen Benutzer betreiben . . . 7
- Aktualisieren . . . 7
- Entfernen und purgen . . . 7
- Weitere Familien: Fedora, openSUSE Leap und Arch Linux . . . 7
- Bekannte Grenzen des Pakets . . . 8

Fehlerbehebung **9**
- Zuerst den Dienstzustand lesen . . . 9
- Der Dienst startet nicht . . . 9
- Das Ziel wird nicht neu gestartet . . . 9
- Das Ziel gilt als blockiert, obwohl es läuft . . . 9
- `--diagnose` endet mit 0, aber das Ziel ist nicht betriebsbereit . . . 10
- Das Testziel antwortet nicht . . . 10
- Das Protokoll wächst unbegrenzt . . . 10
- Doppelte Ziele nach einem Supervisor-Neustart . . . 10
- Nichts im Protokoll . . . 10

Überblick

Das Programm gibt die --help-Zusammenfassung in der Sprache des Systems aus: standardmäßig Englisch, übersetzt sind fr, de, es und zh.

AllWatch ist ein winziger Prozess-Supervisor für Linux, geschrieben in einem einzigen Bash-Skript (etwa hundertsiebzig nutzbare Zeilen). Er startet jedes Ziel im Vordergrund, merkt sich dessen PID und beurteilt seinen Zustand nur anhand von Kernel und Netzwerk: Existiert der Prozess noch in /proc, nimmt sein TCP-Port noch Verbindungen an, und welchen Exit-Code hat er hinterlassen? Er startet das Ziel gemäß einer Richtlinie pro Ziel neu und gerät nicht in eine Endlosschleife bei einem Ziel, das sofort stirbt.

Das Ziel ist eine Blackbox. Es registriert sich nicht bei AllWatch, sendet ihm nichts und muss nicht wissen, dass AllWatch existiert.

Installation auf Debian und Ubuntu

Das Debian-Paket installiert den Supervisor als /usr/bin/allwatch, die systemd-Unit allwatch.service, die Beispielkonfiguration und die Handbuchseite in fünf Sprachen. Laden Sie das Paket und SHA256SUMS von der Download-Seite herunter und führen Sie aus:

```
sudo apt install ./allwatch_0.1.0-7_all.deb
```

Angekündigte Stufe: **Installation und Diagnose im Container validiert**, auf Debian 12 und 13 sowie auf Ubuntu 22.04 und 24.04. Das Debian-Paket wurde außerdem real erprobt: als Dienst auf einem Raspberry Pi 4 unter Debian 13 installiert, mit einer echten Anwendung, zwei Stunden und danach sechs Stunden lang (95 Ereignisse, 95 bestanden, keines fehlgeschlagen), dann entfernt und bereinigt. Der Dienst wird aktiviert installiert, mit einer Konfiguration ohne Ziel; tragen Sie Ihre Ziele in /etc/allwatch/allwatch.conf ein und führen Sie dann sudo systemctl restart allwatch aus.

Prüfen Sie die installierte Version und die Konfiguration, ohne ein Ziel zu starten:

```
allwatch --version
allwatch --help
allwatch --diagnose /etc/allwatch/allwatch.conf
```

--diagnose liest die Konfiguration, das Zustandsverzeichnis und den Journalpfad. Es startet und prüft kein Ziel; es beweist nicht, dass ein Ziel funktionsfähig ist, sondern dass der Supervisor seine Konfiguration lesen und seine Pfade nutzen kann.

Zum Entfernen:

```
sudo apt remove allwatch
sudo apt purge allwatch
```

apt remove stoppt den Dienst, lässt ihn aber aktiviert; apt purge entfernt zusätzlich die Konfiguration und den Dienstzustand.

Installation auf Fedora, openSUSE Leap und Arch Linux

Derselbe Tag baut auch native Pakete: ein .rpm für Fedora, ein .rpm für openSUSE Leap und ein .pkg.tar.zst für Arch Linux. Laden Sie die Datei Ihrer Familie und SHA256SUMS von der Download-Seite herunter und führen Sie die Zeile Ihrer Familie aus (Fedora, openSUSE Leap, Arch Linux, in dieser Reihenfolge):

```
sudo dnf install ./allwatch-0.1.0-1.fc44.noarch.rpm
sudo zypper install --allow-unsigned-rpm ./allwatch-0.1.0-1.leap160.noarch.rpm
sudo pacman -U ./allwatch-0.1.0-1-any.pkg.tar.zst
```

Angegebene Stufe: **Installation und Diagnose im Container validiert**, auf Fedora 43, 44 und 45, auf openSUSE Leap 16.0 und auf Arch Linux. Die Pakete werden nicht als auf einem physischen Server getestet dargestellt. Der Dienst wird installiert, aber nicht gestartet: Tragen Sie Ihre Ziele in /etc/allwatch/allwatch.conf ein und führen Sie dann sudo systemctl enable --now allwatch aus. Die Befehle allwatch --version, --help und --diagnose sind die oben gezeigten.

So entfernen Sie das Paket:

```
sudo dnf remove allwatch
sudo zypper remove allwatch
sudo pacman -R allwatch
```

Beim Entfernen bleiben der Systembenutzer allwatch und /var/lib/allwatch erhalten, ebenso Ihre Konfiguration, wenn Sie sie geändert haben (gesichert als allwatch.conf.rpmsave oder allwatch.conf.pacsave). So löschen Sie auch diese:

```
sudo userdel allwatch
sudo rm -rf /etc/allwatch /var/lib/allwatch
```

Aus dem Quelltext starten

Voraussetzungen: Linux (der Supervisor liest /proc), Bash 5 oder neuer sowie timeout und awk aus den üblichen Coreutils. Nur für das Testziel dummy-target.sh: setpriv (util-linux) und OpenBSD nc.

```
git clone https://github.com/RonanDavalan/AllWatch.git
cd AllWatch
./allwatch.sh --version
./allwatch.sh --help
./allwatch.sh allwatch.conf
```

Die Beispielkonfiguration startet dummy-target.sh auf Port 9001 unter der Kennung demo. In einem zweiten Terminal:

```
cat /tmp/allwatch-runtime/demo.pid
: > /dev/tcp/127.0.0.1/9001 && echo "Port 9001 antwortet"
```

Töten Sie das Ziel hart und beobachten Sie, wie es mit einer neuen PID zurückkommt (Abfrageintervall 5 Sekunden):

```
kill -9 "$(cat /tmp/allwatch-runtime/demo.pid)"
sleep 6
cat /tmp/allwatch-runtime/demo.pid
```

Lösen Sie einen Hänger aus (das Ziel bleibt am Leben, schließt aber seinen Port). AllWatch erkennt es, stoppt und startet es neu, was etwa fünfzehn Sekunden dauert:

```
kill -USR1 "$(cat /tmp/allwatch-runtime/demo.pid)"
sleep 20
grep ' demo ' ~/.local/state/allwatch/allwatch.log | tail -n 6
```

Beenden Sie den Supervisor mit Ctrl+C (oder SIGTERM); er stoppt seine Ziele und leert sein Zustandsverzeichnis:

```
ls -A /tmp/allwatch-runtime
```

Arbeitsweise

AllWatch ist der Elternprozess, das Ziel ist das Kind. Alle paar Sekunden prüft er jedes Ziel und ordnet es einem von vier Zuständen zu:

- **Läuft**: Der Prozess existiert und sein Port nimmt eine Verbindung an.
- **Hängt**: Der Prozess existiert, aber sein Port antwortet nicht. Mit ON_FREEZE=restart (Standard) sendet AllWatch SIGTERM, wartet 5 Sekunden, sendet bei Bedarf SIGKILL und behandelt das Ziel danach als fehlgeschlagen. Mit ON_FREEZE=alert sendet es kein Signal, schreibt einmal ALERTE GEL, lässt das Ziel überwacht und schreibt RETABLI, wenn der Port wieder antwortet.
- **Abgestürzt**: Der Prozess ist ohne geordnetes Ende verschwunden. AllWatch liest den Exit-Code und startet das Ziel gemäß seiner Richtlinie neu.

- **Beendet**: Das Ziel ist beendet und seine Richtlinie verlangt keinen Neustart, oder das Neustartlimit wurde erreicht.

Konfiguration

Die Konfiguration wird beim Start einmal zeilenweise gelesen. Leere Zeilen und Zeilen, die mit # beginnen, werden ignoriert. Allgemeine Parameter, ein KEY=value pro Zeile:

- POLL_SECONDS=5: Sekunden zwischen zwei Prüfungen.
- PROBE_TIMEOUT=3: Sekunden, die ein Port für die Probe hat.
- MAX_RESTARTS=5: erlaubte Neustarts im folgenden Fenster.
- RESTART_WINDOW=60: Länge dieses Fensters in Sekunden.
- ON_FREEZE=restart: Ein hängendes Ziel wird neutralisiert und neu gestartet; alert lässt es unberührt und schreibt ALERTE GEL. Jeder andere Wert wird beim Start mit Exit-Code 1 zurückgewiesen.

Ziele, eines pro Zeile:

```
id port policy command...
```

- id: ein kurzer Name für Zustandsdateien und Journal.
- port: der TCP-Port des Ziels (geprüft auf 127.0.0.1).
- policy: always, on-failure oder transient.
- command...: der Rest der Zeile, an Leerzeichen getrennt (ohne Anführungszeichen), im Vordergrund aus dem Startverzeichnis ausgeführt.

Richtlinien:

- always: das Ziel nach jedem Ende neu starten, auch nach Exit-Code 0, außer eine Datei <id>.stop liegt im Zustandsverzeichnis.
- on-failure: nach Absturz oder Hänger neu starten, nach Exit-Code 0 beendet lassen.
- transient: niemals neu starten.

Braucht ein Ziel MAX_RESTARTS Neustarts innerhalb von RESTART_WINDOW Sekunden, stellt AllWatch die Neustarts ein, schreibt ALERTE VERROU_CRITIQUE und führt die anderen Ziele weiter.

Das Zustandsverzeichnis ist /tmp/allwatch-runtime (überschreibbar mit ALLWATCH_RUNTIME); das Journal ist eine einfache Textdatei, eine Zeile pro Ereignis (date id event), unter ${XDG_STATE_HOME:-$HOME/.local/state}/allwatch/allwatch.log.

Vertrag eines Ziels

AllWatch passt sich an nichts an; ein Ziel, das den Vertrag bricht, ist im Fehler.

- **Vordergrund.** Das Ziel läuft im Vordergrund des von AllWatch gestarteten Prozesses. Es darf nicht dämonisieren oder doppelt forken.
- **Exit-Codes.** Ein sauberer, angeforderter Stopp endet mit Code 0. Jeder Fehler endet mit einem Code ungleich null.
- **Ein TCP-Port.** Jedes Ziel besitzt einen TCP-Port, offen und verbindungsbereit ab seiner Initialisierung. Ein lebendes Ziel, das im Probe-Zeitfenster keine Verbindung annimmt, gilt als hängend.
- **SIGTERM.** Das Ziel fängt SIGTERM, gibt seine Ressourcen frei und endet innerhalb von 5 Sekunden mit Code 0. Danach sendet AllWatch SIGKILL.

Fenstertitel, Anzeigeserver und die eigene Ausgabe des Ziels spielen für die Zustandsprüfung keine Rolle.

Grenzen

- Kein Hot-Reload der Konfiguration: sudo systemctl restart allwatch.
- Keine Prüfung über Fenstertitel, X11 oder Wayland.
- Keine Analyse von Anwendungsjournalen für Neustartentscheidungen.

- Keine Überwachung von Zielen, die sich im Hintergrund ablösen.
- --diagnose ist statisch: Es liest Konfiguration und Pfade, prüft kein Ziel und beweist nicht, dass ein Ziel funktionsfähig ist.
- --version, --help und --diagnose werden nur als erstes Argument erkannt.
- Der Befehl läuft nicht durch eine Shell: keine Anführungszeichen, Pipes, Umleitungen oder Variablenexpansion.
- Konfigurationswerte und Richtliniennamen werden nicht validiert.
- Das einfache Textjournal wird nicht rotiert.

Lizenz

MIT. Siehe LICENSE.

Danksagung

- **Ronan Davalan**: Architekt und Schiedsrichter. Er setzte Vision und Umfang, entschied über die Spezifikationen und validierte sie.
- **Claude Sonnet 5.5** (Anthropic): alleiniger Autor des Codes (Supervisor und Testziel), geschrieben in drei Sitzungen am 13. November 2026.
- **DeepSeek V4.1 Flash 1017**: Referenzdokumentation, sechsstündige Validierungskampagne und Kampagnenbericht.
- **Gemini 4 Hydrogène** (Google): Entwurfs- und Denkgespräche zum Supervisor, Idee und Entwurf des Symbols.

Siehe CONTRIBUTORS.md.

Den Debian-Dienst installieren und betreiben

Das Debian-Paket installiert AllWatch als systemd-Dienst. Der Dienst startet beim Booten, liest seine Konfiguration einmal und wartet, wenn kein Ziel deklariert ist. Dieses Kapitel nennt die genauen Pfade, Dateien und Befehle des Pakets.

Was das Paket installiert

```
/usr/bin/allwatch                          der Supervisor
/usr/bin/allwatch.sh                       Kompatibilitäts-Symlink
/usr/lib/systemd/system/allwatch.service   die systemd-Unit
/etc/allwatch/allwatch.conf                die Konfiguration
/usr/share/man/man1/allwatch.1.gz          Handbuchseite, Englisch
/usr/share/man/fr/man1/allwatch.1.gz       Handbuchseite, Französisch
/usr/share/man/de/man1/allwatch.1.gz       Handbuchseite, Deutsch
/usr/share/man/es/man1/allwatch.1.gz       Handbuchseite, Spanisch
/usr/share/man/zh_CN/man1/allwatch.1.gz    Handbuchseite, vereinfachtes Chinesisch
/usr/share/doc/allwatch/README.Debian.gz   Pakethinweise
/usr/share/doc/allwatch/examples/allwatch.conf.example
/usr/share/doc/allwatch/examples/allwatch.conf  Konfiguration des Quell-Repositorys
/usr/share/doc/allwatch/examples/dummy-target.sh
```

Das Paket legt den Systembenutzer und die Gruppe allwatch an (Home /var/lib/allwatch, Shell /usr/sbin/nologin). Der Dienst läuft unter dieser Kennung ab /var/lib/allwatch.

Installieren

```
sudo apt install ./allwatch_0.1.0-7_all.deb
```

Das Paket installiert das Programm und die Unit. Wenn systemd läuft, aktiviert es allwatch.service und startet den Dienst. Die ausgelieferte Konfiguration enthält kein aktives Ziel: Der Dienst startet und wartet.

Prüfen Sie Paketzustand und Programmversion:

```
dpkg -s allwatch | grep -E '^(Status|Version):'
allwatch --version
```

allwatch --version gibt allwatch 0.1.0 aus; die Paketversion ist 0.1.0-7.

Pfade und Dateien

- /etc/allwatch/allwatch.conf ist eine Conffile von root und für den Dienstbenutzer nicht beschreibbar. Sie ist die Vertrauensgrenze: Zielbefehle laufen mit den Rechten des Dienstbenutzers.
- /var/lib/allwatch enthält Zustand und Protokoll, Modus 0700, Eigentümer allwatch:allwatch, von systemd angelegt. Das einfache Textprotokoll ist /var/lib/allwatch/allwatch/allwatch.log.
- /run/allwatch ist das Laufzeitverzeichnis, Modus 0700, bei jedem Dienststart angelegt und beim Stoppen entfernt. Es enthält <id>.pid, <id>.fails und <id>.stop.
- /var/lib/allwatch und /run/allwatch überleben keine Purge; /run/allwatch überlebt keinen Stopp.

Ein Ziel deklarieren

Bearbeiten Sie /etc/allwatch/allwatch.conf als root und fügen Sie eine Zeile pro Ziel hinzu:

```
id port policy command...
```

Ein echtes Beispiel mit dem vom Paket installierten Testziel:

```
sudo apt install netcat-openbsd
sudo cp /usr/share/doc/allwatch/examples/allwatch.conf.example /etc/allwatch/allwatch.conf
sudo systemctl restart allwatch
```

Das Beispiel deklariert:

```
demo 9001 on-failure /usr/share/doc/allwatch/examples/dummy-target.sh 9001
```

Prüfen Sie den Port:

```
bash -c ': > /dev/tcp/127.0.0.1/9001' && echo "Port 9001 antwortet"
```

Der Dienstbenutzer muss den Befehl ausführen können. Verwenden Sie absolute Pfade: Der Befehl wird an Leerzeichen getrennt, ohne Anführungszeichen, und nicht über eine Shell gestartet. Für Argumente mit Leerzeichen oder Shell-Syntax schreiben Sie ein Wrapper-Skript und rufen es auf.

Hängendes Ziel: restart oder alert

ON_FREEZE entscheidet, was mit einem Ziel geschieht, das noch lebt, dessen TCP-Port aber nach der Gnadenfrist nicht mehr antwortet. ON_FREEZE=restart (der ausgelieferte Standard) sendet SIGTERM, wartet etwa fünf Sekunden, sendet bei Bedarf SIGKILL und startet das Ziel gemäß seiner Richtlinie neu. ON_FREEZE=alert sendet kein Signal: Es schreibt einmal ALERTE GEL, lässt das Ziel laufen und schreibt RETABLI, wenn der Port wieder antwortet. Ein Absturz folgt in beiden Fällen der Richtlinie des Ziels. Jeder andere Wert wird beim Start mit Exit-Code 1 zurückgewiesen.

Dienstbefehle

```
sudo systemctl status allwatch
sudo systemctl start allwatch
sudo systemctl stop allwatch
sudo systemctl restart allwatch
sudo systemctl enable allwatch
sudo systemctl disable allwatch
sudo journalctl -u allwatch -f
sudo tail -f /var/lib/allwatch/allwatch/allwatch.log
```

Die Konfiguration wird einmal beim Start gelesen; eine Änderung erfordert sudo systemctl restart allwatch. Es gibt kein Hot-Reload.

Die Konfiguration prüfen, ohne ein Ziel zu starten

Führen Sie --diagnose mit der Dienstumgebung und als Dienstbenutzer aus:

```
sudo -u allwatch env ALLWATCH_RUNTIME=/run/allwatch \
  XDG_STATE_HOME=/var/lib/allwatch \
  allwatch --diagnose /etc/allwatch/allwatch.conf
```

Es zeigt Version, Konfigurationspfad, Laufzeitpfad, Protokollpfad, das Verhalten bei Hängern und für jedes Ziel Kennung, Port und Richtlinie. Es endet mit 0, wenn die Konfiguration lesbar, ein vorhandenes Laufzeitverzeichnis dem aktuellen Benutzer gehörend und ein vorhandenes Protokollverzeichnis beschreibbar ist. Es endet mit 1 bei unlesbarer Konfiguration, ungültigem ON_FREEZE, nicht numerischem POLL_SECONDS, einem vorhandenen Laufzeitverzeichnis, das nicht dem aktuellen Benutzer gehört, oder einem vorhandenen Protokollverzeichnis, das nicht beschreibbar ist. Es startet kein Ziel und sondiert keinen Port.

Ein Ziel stoppen, ohne den Dienst zu stoppen

Legen Sie die Stoppdatei als Dienstbenutzer an:

```
sudo -u allwatch touch /run/allwatch/demo.stop
```

AllWatch entfernt die Datei, wenn das Ziel endet, und startet es nicht neu. Die Datei stoppt ein laufendes Ziel nicht von selbst; senden Sie SIGTERM an das Ziel oder lassen Sie es enden.

Den Dienst unter einem anderen Benutzer betreiben

Der Dienst läuft als allwatch. Wenn die Ziele root oder ein anderes Konto brauchen, legen Sie ein Drop-in an:

```
sudo systemctl edit allwatch
```

Fügen Sie hinzu:

```
[Service]
User=www-data
Group=www-data
```

Dann neu starten:

```
sudo systemctl restart allwatch
```

Die Zielbefehle erben diese Identität. Die Konfigurationsdatei bleibt Eigentum von root.

Aktualisieren

```
sudo apt install ./allwatch_0.1.0-7_all.deb
```

/etc/allwatch/allwatch.conf bleibt erhalten; wenn die Paketversion sie ändert, fragt dpkg, welche Version behalten werden soll. Der Dienst wird nach der Aktualisierung neu gestartet, wenn er aktiv war. /var/lib/allwatch und sein Protokoll bleiben.

Entfernen und purgen

```
sudo apt remove allwatch
sudo apt purge allwatch
```

apt remove stoppt den Dienst, entfernt Programm, Unit, Handbuchseiten und Paketdokumentation. Es behält /etc/allwatch/allwatch.conf, /var/lib/allwatch sowie Benutzer und Gruppe allwatch. Die Unit wird nicht deaktiviert; der Aktivierungszustand von deb-systemd-helper bleibt bis zur Purge.

apt purge entfernt zusätzlich die Conffile, /var/lib/allwatch, /etc/allwatch wenn leer, Benutzer und Gruppe allwatch wenn möglich sowie den Aktivierungszustand. Wenn ein Prozess das Konto noch nutzt, können userdel oder groupdel scheitern; prüfen Sie mit:

```
getent passwd allwatch
getent group allwatch
```

Weitere Familien: Fedora, openSUSE Leap und Arch Linux

Das .rpm für Fedora und openSUSE Leap und das .pkg.tar.zst für Arch Linux installieren dieselben Dateien an denselben Pfaden wie das Debian-Paket: /usr/bin/allwatch, /usr/lib/systemd/system/allwatch.service, /etc/allwatch/allwatch.conf, die Handbuchseiten und /usr/share/doc/allwatch. Installieren Sie die Datei Ihrer Familie (Fedora, openSUSE Leap, Arch Linux, in dieser Reihenfolge); dieselbe Zeile aktualisiert eine installierte Version:

```
sudo dnf install ./allwatch-0.1.0-1.fc44.noarch.rpm
sudo zypper install --allow-unsigned-rpm ./allwatch-0.1.0-1.leap160.noarch.rpm
sudo pacman -U ./allwatch-0.1.0-1-any.pkg.tar.zst
```

Unterschiede zum Debian-Paket: Der Dienst wird installiert, aber nicht gestartet; führen Sie sudo systemctl enable --now allwatch aus, sobald Ihre Ziele eingetragen sind. /etc/allwatch/allwatch.conf bleibt erhalten, wenn Sie sie geändert haben, und wird beim Entfernen als allwatch.conf.rpmsave (Fedora, openSUSE) oder allwatch.conf.pacsave (Arch) gesichert. Es gibt kein Purge: Das Entfernen behält den Benutzer allwatch und /var/lib/allwatch, und das Paket gibt die beiden Befehle aus, die sie löschen:

```
sudo userdel allwatch
sudo rm -rf /etc/allwatch /var/lib/allwatch
```

Bekannte Grenzen des Pakets

- Die Containervalidierung installiert das Paket, prüft --version, --help und --diagnose und entfernt es wieder. Sie startet systemd nicht; der Dienst ist installiert und dokumentiert, aber im Container nicht unter systemd validiert.
- --diagnose ist statisch: Es sondiert keine Ziele.
- /run/allwatch überlebt einen Dienststopp nicht. Nach einem Supervisor-Absturz können Ziele ohne PID-Datei überleben; der neu gestartete Supervisor kann Duplikate starten. Ein Drop-in mit RuntimeDirectoryPreserve=restart behält das Verzeichnis über einen Dienstneustart hinweg.

Fehlerbehebung

AllWatch hat keinen Statusbefehl. Der Zustand eines Ziels wird aus dem Prozess, dem TCP-Port und dem einfachen Textprotokoll gelesen. Dieses Kapitel nennt die genauen Befehle für die häufigen Störungen.

Zuerst den Dienstzustand lesen

```
sudo systemctl status allwatch
sudo journalctl -u allwatch -n 50 --no-pager
sudo tail -n 50 /var/lib/allwatch/allwatch/allwatch.log
```

Die Protokollzeilen sind Datum id Ereignis. Die Ereignisse sind DEMARRAGE, DEMARRE, ADOPTE, PLANTE, ALERTE GEL, RETABLI, NEUTRALISE SIGTERM, NEUTRALISE SIGKILL, CRASHE code=N, RELANCE, ALERTE VERROU_CRITIQUE, QUITTE und ARRET.

Der Dienst startet nicht

Das Programm schreibt diese Meldungen in der Sprache des Systems (LC_ALL, LC_MESSAGES oder LANG); Standard ist Englisch.

- allwatch: Konfiguration nicht lesbar: /etc/allwatch/allwatch.conf — die Datei fehlt oder ist für den Dienstbenutzer nicht lesbar. Prüfen Sie sudo -u allwatch test -r /etc/allwatch/allwatch.conf und sudo systemctl cat allwatch.
- allwatch: /run/allwatch gehört nicht dem aktuellen Benutzer — das Laufzeitverzeichnis gehört einem anderen Benutzer. Prüfen Sie ls -ld /run/allwatch /var/lib/allwatch; die Unit legt sie für allwatch an. Wenn Sie ein Drop-in mit einem anderen User= hinzugefügt haben, korrigieren Sie den Eigentümer oder entfernen Sie das Drop-in.
- status=203/EXEC — das Programm fehlt oder ist nicht ausführbar. Installieren Sie das Paket neu.
- status=1/FAILURE mit einem Konfigurationsfehler — führen Sie --diagnose als Dienstbenutzer aus (siehe unten).

Das Ziel wird nicht neu gestartet

- Richtlinie on-failure: Ein Ziel, das mit Code 0 endet, bleibt gestoppt. Das ist der Vertrag, kein Fehler. Verwenden Sie always, wenn es nach einem sauberen Ende neu starten soll.
- Richtlinie transient: wird nie neu gestartet.
- Stoppdatei vorhanden: ls -l /run/allwatch/demo.stop; entfernen Sie sie, wenn der Stopp nicht gewollt war. Die Datei stoppt ein laufendes Ziel nicht von selbst.
- ALERTE VERROU_CRITIQUE: Das Ziel hat MAX_RESTARTS in RESTART_WINDOW erreicht. Beheben Sie die Ursache und starten Sie den Dienst neu; die Sperre gilt bis zum Neustart des Supervisors.
- CRASHE code=N: Das Ziel endete mit Code N. Lesen Sie sein eigenes Protokoll; AllWatch meldet nur den Code.

Das Ziel gilt als blockiert, obwohl es läuft

- Die Sonde ist timeout 3 bash -c ': > /dev/tcp/127.0.0.1/9001'. Das Ziel muss auf 127.0.0.1 (oder 0.0.0.0) am deklarierten Port lauschen. Ein Ziel, das nur auf einer anderen Adresse lauscht, gilt als blockiert.

- Falscher Port in /etc/allwatch/allwatch.conf; prüfen Sie mit ss -ltnp | grep 9001.
- Das Ziel ist langsamer als PROBE_TIMEOUT=3; erhöhen Sie den Wert in der Konfiguration und starten Sie den Dienst neu.
- Das Ziel lebt, aber sein Listener ist geschlossen; genau das ist der Blockierzustand. Mit ON_FREEZE=restart (Standard) sendet AllWatch SIGTERM, wartet 5 Sekunden, sendet SIGKILL und startet das Ziel neu. Mit ON_FREEZE=alert schreibt es einmal ALERTE GEL und lässt das Ziel laufen; RETABLI wird geschrieben, wenn der Port wieder antwortet.

`--diagnose` endet mit 0, aber das Ziel ist nicht betriebsbereit

--diagnose ist statisch. Es liest die Konfiguration und die Pfade; es sondiert kein Ziel. Ein Rückgabecode 0 beweist nur, dass der Supervisor die Konfiguration lesen und seine Verzeichnisse nutzen kann.

Das Testziel antwortet nicht

- netcat-openbsd fehlt: sudo apt install netcat-openbsd.
- Der Befehl verwendet einen relativen Pfad. Der Dienst läuft ab /var/lib/allwatch; verwenden Sie den absoluten Pfad /usr/share/doc/allwatch/examples/dummy-target.sh.
- Der Port ist bereits belegt: ss -ltnp | grep 9001.
- Der Dienstbenutzer kann die Datei nicht ausführen: sudo -u allwatch test -x /usr/share/doc/allwatch/examples/dummy-target.sh.

Das Protokoll wächst unbegrenzt

AllWatch rotiert das Protokoll nicht. Verwenden Sie logrotate oder einen systemd-Timer; die Datei ist /var/lib/allwatch/allwatch/allwatch.log.

Doppelte Ziele nach einem Supervisor-Neustart

/run/allwatch wird beim Dienststopp entfernt. Ein Ziel kann einen Supervisor-Absturz ohne seine PID-Datei überleben; der neue Supervisor startet ein Duplikat. Fügen Sie ein Drop-in hinzu:

```
sudo systemctl edit allwatch
```

```
[Service]
RuntimeDirectoryPreserve=restart
```

Dann neu starten. Das Laufzeitverzeichnis bleibt über einen Dienstneustart erhalten, und der neue Supervisor kann die überlebende PID übernehmen.

Nichts im Protokoll

Der Dienst läuft als allwatch und schreibt nach /var/lib/allwatch/allwatch/allwatch.log; prüfen Sie die Dienstumgebung mit sudo systemctl show allwatch -p Environment. Wenn das Verzeichnis nicht beschreibbar ist, bleibt das Protokoll leer; --diagnose meldet es.