Raspberry Pi als APC-USV-Server für Home Assistant

Diese Anleitung nutzt **apcupsd mit Modbus** für eine APC Smart-UPS SMT1500I und einen Raspberry Pi (getestet: Pi 1 Model B+, Raspberry Pi OS Legacy Lite 32 Bit). Home Assistant liest die USV über das Netzwerk aus. Ein Watchdog behebt das bei USB-Modbus beobachtete Einfrieren der Messwerte. Mittelfristig ist der Wechsel auf **serielles RS-232-Modbus** vorgesehen.

Aufbau: APC-USV – USB (später RS-232) – Raspberry Pi/apcupsd – TCP 3551 – Home Assistant.

1. Raspberry Pi und USV vorbereiten

1. Raspberry Pi OS Lite passend zum Pi installieren, Netzwerk/SSH einrichten und dem Pi eine feste IP bzw. DHCP-Reservierung geben. Pi und Netzwerkverbindung für die gewünschte USV-Überwachung ebenfalls notstromversorgen.
2. USV per USB an den Pi anschließen. An der USV unter **Advanced / Configuration** die Option **Modbus = Enabled** setzen.
3. Pakete installieren:

```
sudo apt update
sudo apt install -y apcupsd cron
```

2. apcupsd konfigurieren

```
sudo nano /etc/apcupsd/apcupsd.conf
```

Diese Einträge setzen (andere gleichnamige aktive Einträge nicht doppelt stehen lassen):

```
UPSCABLE usb
UPSTYPE modbus
DEVICE
POLLTIME 30

NETSERVER on
NISIP 0.0.0.0
NISPORT 3551

ONBATTERYDELAY 6
BATTERYLEVEL 10
MINUTES 5
TIMEOUT 0
KILLDELAY 0
```

Warum: `DEVICE` bleibt bei USB-Modbus leer; `POLLTIME 30` fragt die USV ungefähr alle 30 s ab.

`NISIP 0.0.0.0` erlaubt den Zugriff über das LAN (Port 3551 nur für vertrauenswürdige Geräte freigeben). Die Batteriegrenzen betreffen die lokale apcupsd-Abschaltung des Pi; sie fahren nicht automatisch andere Server herunter.

Bei älteren Debian-/Pi-OS-Installationen ggf. in `/etc/default/apcupsd` zusätzlich `ISCONFIGURED=yes` setzen.

```
sudo systemctl enable --now apcupsd
sudo systemctl restart apcupsd
apcaccess status | grep -E '^(DATE|STATUS|LINEV|LOADPCT|END APC)'
```

Der angezeigte DATE-Zeitpunkt muss aktuell sein, STATUS normalerweise `ONLINE`.

3. Home Assistant anbinden

Einstellungen → Geräte & Dienste → Integration hinzufügen → APC UPS Daemon. IP-Adresse des Pi und Port **3551** eintragen. Die HA-Integration fragt standardmäßig **alle 60 s** ab; `POLLTIME 30` auf dem Pi bedeutet also **keine** 30-s-Aktualisierung in HA. Vor Abschaltautomationen stets die Aktualität der Werte berücksichtigen.

Optional: Ein Leistungssensor lässt sich aus LOADPCT und der **Wirkleistungs-Nennleistung** der jeweiligen USV abschätzen, z. B. bei 1.000 W Nennleistung: `Last in % * 10 = ungefähr W`. Das ist kein echter Wirkleistungs- oder Energiezähler; bei fehlenden Werten **nicht** automatisch 0 W annehmen.

4. Dienst bei Prozessfehler automatisch neu starten

```
sudo systemctl edit apcupsd
```

```
[Service]
Restart=on-failure
RestartSec=10s
```

```
sudo systemctl daemon-reload
sudo systemctl restart apcupsd
```

Grenze: `Restart=on-failure` hilft nur bei Prozessfehlern. Ein laufender Daemon kann trotzdem **eingefrorene Modbus-Daten** liefern.

5. USB-Modbus-Freeze: automatischer Watchdog

Beobachteter Fehler: `apcaccess` liefert weiter einen neuen Statusbericht (`END APC`), aber DATE bleibt stundenlang stehen. DATE ist der Zeitpunkt der letzten Datenaktualisierung von der USV; END APC nur der Berichtzeitpunkt. Ein manueller `systemctl restart apcupsd` hat die Daten jeweils wiederhergestellt.

Nicht die Netzspannung vergleichen! LINEV darf im Normalbetrieb unverändert bleiben. Der Watchdog prüft deshalb jede Minute, ob DATE mehr als **120 s** alt ist, bestätigt den Fehler nach 10 s und startet apcupsd neu. Neustarts sind auf **höchstens einen Versuch pro 15 Minuten** begrenzt; LOWBATT und SHUTTING DOWN werden geschützt. Fehler und Erfolge landen im Journal.

Skript mit **korrekter Bash-Startzeile** anlegen (den gesamten Block ausführen):

```
sudo tee /usr/local/sbin/apcupsd-watchdog.sh >/dev/null <<'SCRIPT'
#!/usr/bin/env bash
set -u

MAX_AGE=120
COOLDOWN=900
STATE=/run/apcupsd-watchdog.last-restart
MODE="${1:-}"

if [[ -n "$MODE" && "$MODE" != "--check" ]]; then
    echo "Aufruf: $0 [--check]" >&2
    exit 2
fi

exec 9>/run/apcupsd-watchdog.lock
if [[ "$MODE" == "--check" ]]; then
    flock -w 5 9 || { echo "BELEGT: Watchdog laeuft bereits."; exit 2; }
else
    flock -n 9 || exit 0
fi

probe() {
    local report date_text stamp now
    reason=""; age=""; ups_status=""; health="fault"
    report=$(timeout 10s /usr/sbin/apcaccess status 2>/dev/null) ||
report=""
    date_text=$(printf '%s\n' "$report" | awk '/^DATE[[:space:]]*:/
{sub(/^[^:]*:[[:space:]]*/, ""); print; exit}')
    ups_status=$(printf '%s\n' "$report" | awk '/^STATUS[[:space:]]*:/
{sub(/^[^:]*:[[:space:]]*/, ""); print; exit}')

    if [[ "$ups_status" == *"SHUTTING DOWN"* || "$ups_status" == *"LOWBATT"*
]]; then
        health="protected"; return 0
    fi
    if [[ "$ups_status" == *"COMMLOST"* ]]; then
        reason="STATUS=COMMLOST"; return 1
    fi
    if [[ -z "$date_text" || -z "$ups_status" ]]; then
        reason="DATE oder STATUS fehlt"; return 1
    fi
    stamp=$(date -d "$date_text" +%s 2>/dev/null) || stamp=""
    if [[ ! "$stamp" =~ ^[0-9]+$ ]]; then
        reason="DATE nicht auswertbar"; return 1
```

```
    fi
    now=$(date +%s)
    age=$((now - stamp))
    if ((age < -60)); then
        health="clock"; reason="DATE liegt in der Zukunft: Systemzeit
pruefen"; return 2
    fi
    if ((age > MAX_AGE)); then
        reason="DATE ist ${age}s alt (Grenzwert ${MAX_AGE}s)"; return 1
    fi
    health="ok"; return 0
}

probe
if [[ "$MODE" == "--check" ]]; then
    case "$health" in
        ok)        echo "OK: DATE ist ${age}s alt, STATUS=$ups_status"; exit
0 ;;
        protected) echo "GESCHUETZT: STATUS=$ups_status, kein Neustart";
exit 0 ;;
        clock)     echo "WARNUNG: $reason"; exit 2 ;;
        *)         echo "FEHLER: $reason (Testmodus: kein Neustart)"; exit 1
;;
    esac
fi

case "$health" in ok|protected|clock) exit 0 ;; esac
sleep 10
probe
case "$health" in ok|protected|clock) exit 0 ;; esac

last=0
if [[ -s "$STATE" ]]; then
    read -r last < "$STATE"
    [[ "$last" =~ ^[0-9]+$ ]] || last=0
fi
now=$(date +%s)
((now - last >= COOLDOWN)) || exit 0
printf '%s\n' "$now" > "$STATE"

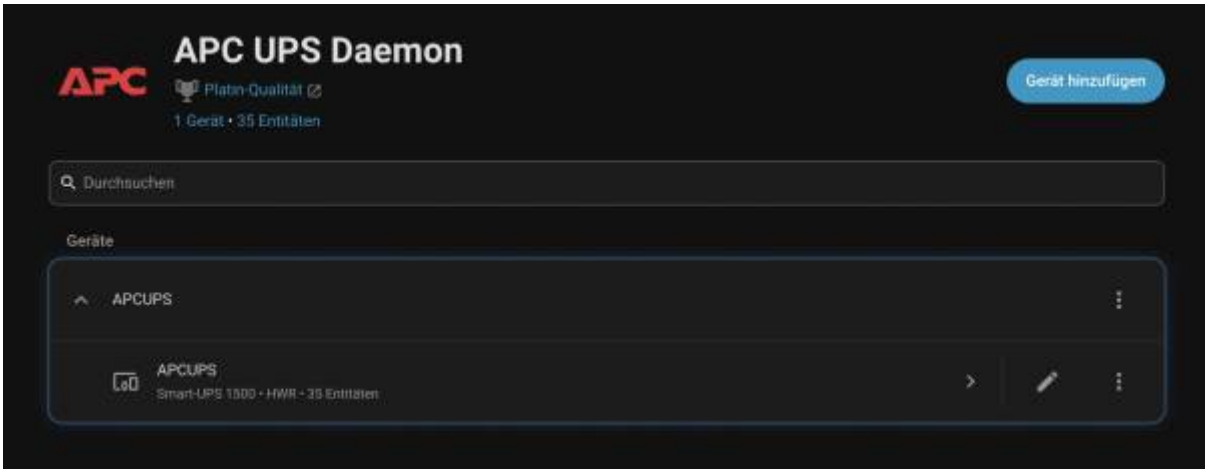
logger -p daemon.warning -t apcupsd-watchdog \
    "USV-Datenfehler bestaetigt: $reason. Starte apcupsd neu."
if ! timeout 30s systemctl restart apcupsd; then
    logger -p daemon.err -t apcupsd-watchdog \
        "Neustart von apcupsd fehlgeschlagen!"
    exit 1
fi
sleep 35
probe
if [[ "$health" == "ok" ]]; then
```

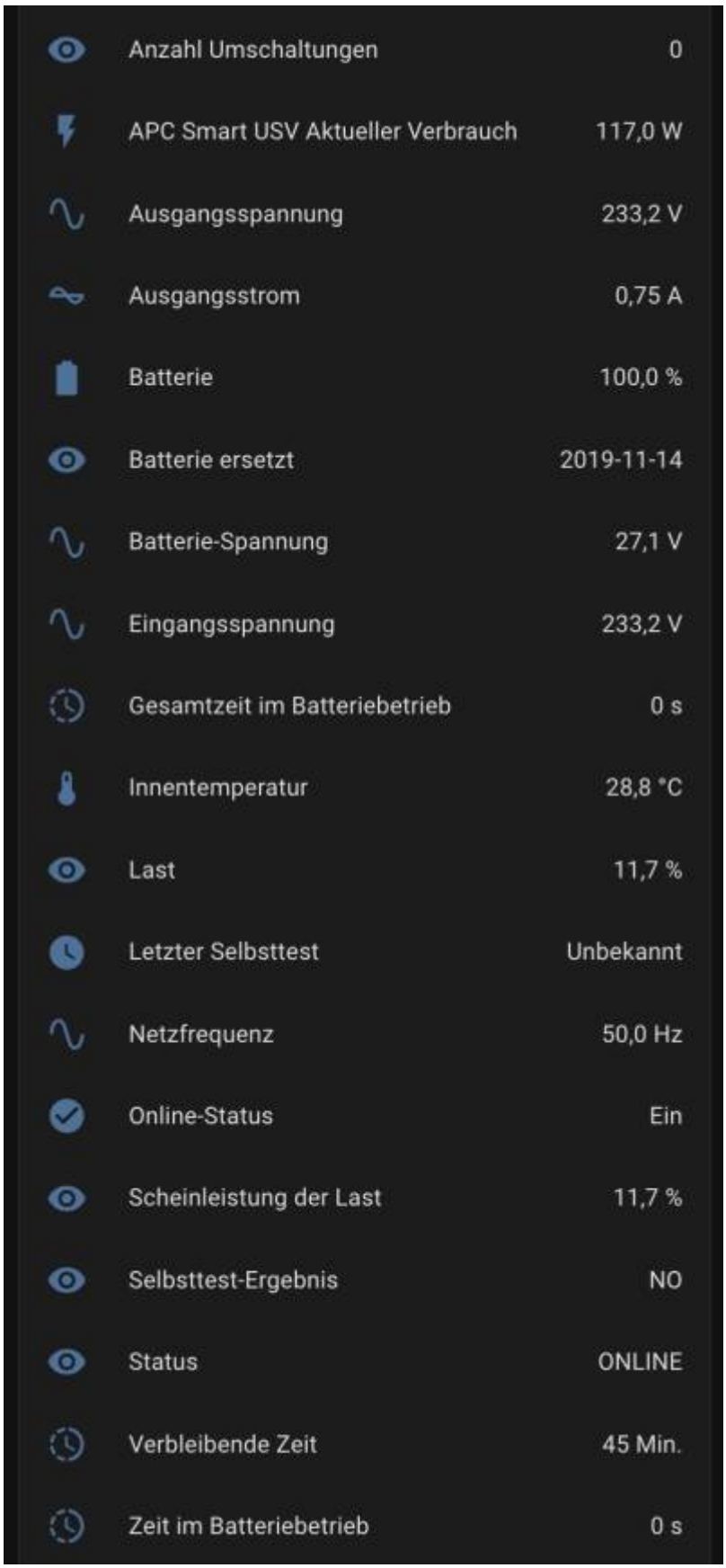
```
    logger -t apcupsd-watchdog \
        "Wiederhergestellt: DATE ${age}s alt, STATUS=$ups_status."
else
    logger -p daemon.err -t apcupsd-watchdog \
        "Nach Neustart noch keine aktuellen Daten ($reason;
STATUS=$ups_status)."
    exit 1
fi
SCRIPT
sudo chmod 750 /usr/local/sbin/apcupsd-watchdog.sh
sudo bash -n /usr/local/sbin/apcupsd-watchdog.sh
sudo /usr/local/sbin/apcupsd-watchdog.sh --check
```

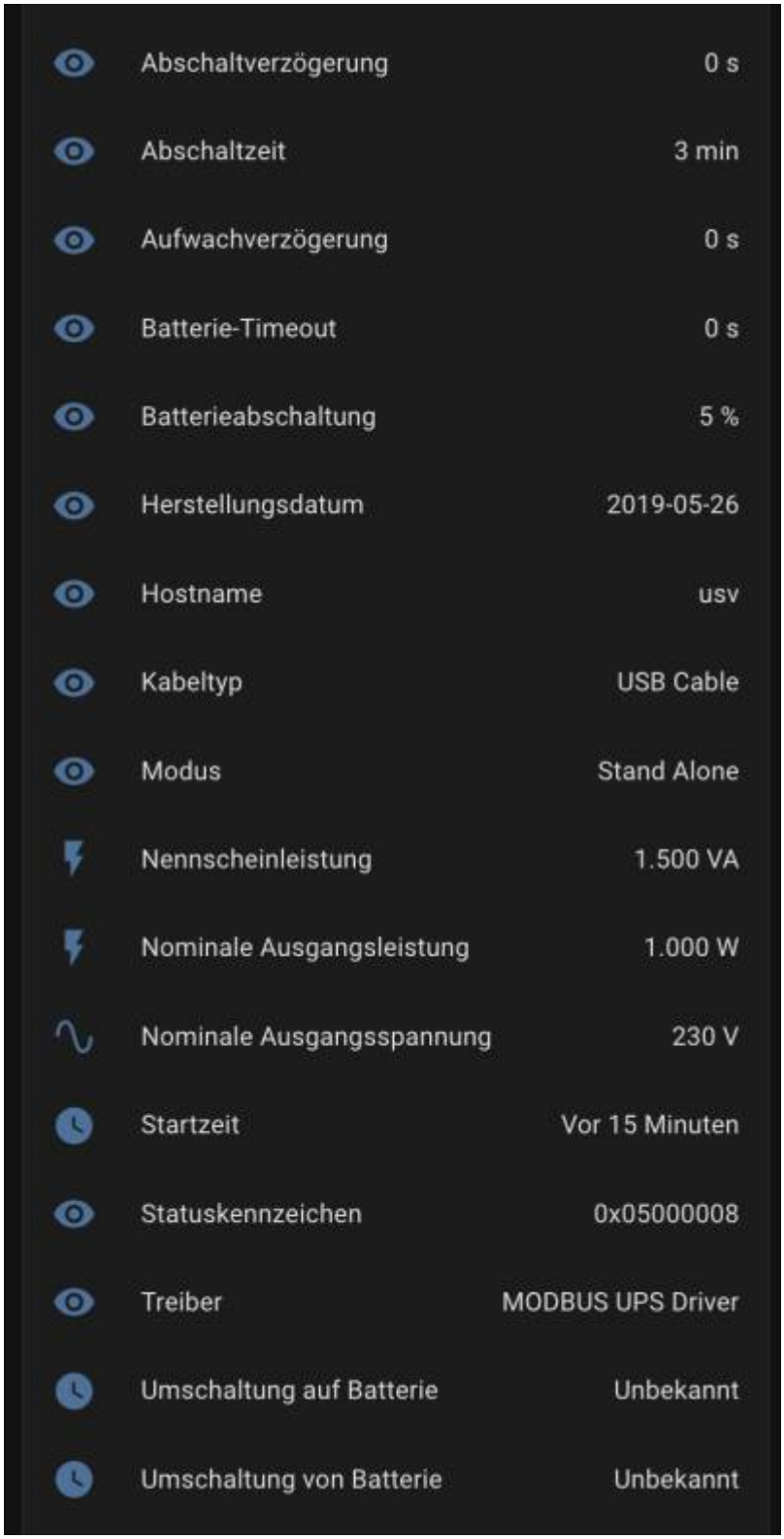
Wichtig: Das Skript muss mit `#!/usr/bin/env bash` **in der ersten Zeile** beginnen. '' grep -E '^(DATE|STATUS|END APC)' </code> ''NETSERVER'', ''NISPORT'', HA-Integration und Watchdog können **unverändert** bleiben. Die Dokumentation beschreibt den späteren Umbau; er ist noch nicht vor Ort getestet. ===== 7. Schnelle Fehlersuche ===== <code bash> apcaccess status | grep -E '^(DATE|STATUS|END APC)' sudo systemctl status apcupsd --no-pager sudo journalctl -u apcupsd -n 40 --no-pager sudo journalctl -t apcupsd-watchdog -n 30 --no-pager lsusb </code> * ''DATE'' alt, ''END APC'' aktuell: **Modbus-Daten eingefroren**, Watchdog sollte eingreifen. * ''DATE'' aktuell, HA-Werte alt: HA-Abfrage oder Netzwerk prüfen. * ''throttled=0x0'' (Pi) und USB ''power/control=on'' waren beim Test unauffällig; USB-Autosuspend war nicht aktiv. * Ein USB-Reset **zeitgleich mit apcupsd-Neustart** ist allein kein Beweis für einen Hardwaredefekt. ===== Quellen / Hinweise ===== * [[https://www.home-assistant.io/integrations/apcupsd/|Home Assistant: APC UPS Daemon (60-s-Polling)

- apcaccess: DATE / END APC
- Bekannter Freeze bei SMT1500I mit USB-Modbus
- Schneider Electric: Warnung vor falschen seriellen Kabeln

Bilder




Anzahl Umschaltungen 0
APC Smart USV Aktueller Verbrauch 117,0 W
Ausgangsspannung 233,2 V
Ausgangsstrom 0,75 A
Batterie 100,0 %
Batterie ersetzt 2019-11-14
Batterie-Spannung 27,1 V
Eingangsspannung 233,2 V
Gesamtzeit im Batteriebetrieb 0 s
Innentemperatur 28,8 °C
Last 11,7 %
Letzter Selbsttest Unbekannt
Netzfrequenz 50,0 Hz
Online-Status Ein
Scheinleistung der Last 11,7 %
Selbsttest-Ergebnis NO
Status ONLINE
Verbleibende Zeit 45 Min.
Zeit im Batteriebetrieb 0 s



From:
https://wiki.mahlen.eu/ - **Smart-Home Wiki**

Permanent link:
https://wiki.mahlen.eu/doku.php?id=home_assistant:home_assistant_rezepte:apcusv_einbinden&rev=1793951605

Last update: **06.11.2026**