

USV anbinden mit den Network UPS Tools

# Der Letzte macht das Licht aus

von Dr. Christian Knemann

Mit einer unterbrechungsfreien Stromversorgung schützen Sie angeschlossene Geräte vor Überspannung und Stromausfall. Sollte der Strom längere Zeit ausbleiben, sorgen die "Network UPS Tools" dafür, dass gleich mehrere Systeme ordnungsgemäß herunterfahren, bevor die Batterien erschöpft sind. IT-Administrator stellt das Open-Source-Projekt vor.

**E**ine unterbrechungsfreie Stromversorgung (USV) – im Englischen als Uninterruptible Power Supply (UPS) bezeichnet – gehört in großen Rechenzentren zur Grundausstattung. Doch auch in Außenstellen und im Home Office haben kleinere Modelle ihre Daseinsberechtigung, helfen sie Ihnen im Ernstfall doch, PCs und NAS noch so lange mit Strom zu versorgen, dass Sie Ihre Systeme sauber herunterfahren können, ohne Datenverlust zu riskieren.

Damit das auch in einem unbeaufsichtigten Moment gelingt, empfiehlt es sich, den Vorgang zu automatisieren. Allerdings verfügen USV-Modelle in entsprechender Größenordnung meist lediglich über eine USB-Schnittstelle, so dass sie im Notfall nur einen angeschlossenen Client über das drohende Unheil informieren können. Hier helfen die Network UPS Tools (NUT) weiter [1].

## Modulare Client-Server-Architektur

Die Open-Source-Software bringt Treiber für USV-Modelle von über 140 Herstellern mit und setzt auf einen modularen Aufbau mit mehreren Daemons. Den Kern bildet der "UPS Information Server" (upsd), der auf einem als Master bezeich-

neten System läuft. Sowohl der Master selbst als auch mehrere Slaves genannte Clients führen zusätzlich den "UPS monitor and shutdown controller" (upsmon) aus. Letzterer überwacht standardmäßig alle fünf Sekunden den Status des Masters und initiiert einen Shutdown, sobald die Batterie der angeschlossenen USV einen kritischen Wert erreicht.

Sollen Clients bereits früher herunterfahren oder weitere Aktionen ausführen, hilft der optionale "Timer helper for scheduling events from upsmon" (upssched). Zu guter Letzt erfasst der "UPS status logger" (upslog) periodisch den Zustand der USV und schreibt konfigurierbare Werte in eine Datei.

Optional sorgt eine Anbindung an das Trio Node-RED, InfluxDB und Grafana für ein erweitertes Logging mitsamt grafischer Auswertung. Doch darauf werden wir später zurückkommen. Wenden wir uns nun zunächst dem grundlegenden Setup zu.

## Raspberry Pi als Master

NUT ist unter der Haube in den Betriebssystemen diverser etablierter NAS-Hersteller aktiv [2]. Doch auch falls Sie kein solches System Ihr Eigen nennen,

ist NUT schnell konfiguriert. Für unseren Workshop verwenden wir im Folgenden den Kleinstcomputer Raspberry Pi sowie exemplarisch eine USV des Herstellers APC. Wir nehmen hierbei an, dass das Betriebssystem "Raspbian Buster Lite" [3] bereits installiert und mit Updates versorgt ist.

Schließen Sie die USV per USB an den Rasi an und verschaffen Sie sich anschließend per *lsusb* einen Überblick über den USB-Bus. In der Ausgabe sollten Sie die USV mitsamt ihrer ID wiederfinden. Da in unserem Beispiel die USV das einzige angeschlossene Gerät ist, fällt die Rückgabe kurz aus. Die ID, in unserem Beispiel "051d:0002", besteht aus der Vendor-ID vor dem Doppelpunkt und der Product-ID dahinter. Merken Sie sich beide Werte zur späteren Verwendung. Mit dem Kommando *ls /dev/usb* finden Sie den Namen der zugehörigen Gerätedatei */dev/usb/lcd/lcd* heraus, die als Parameter für den nächsten Befehl dient:

```
udevadm info --attribute-walk  
--name=/dev/usb/lcd/lcd | egrep  
'manufacturer|product|serial'
```

Dieser verrät Ihnen zusätzlich zu Vendor- und Product-ID auch die Serien-

Quelle: jagcz - 123RF

nummer der USV. Nun installieren Sie NUT mitsamt allen Abhängigkeiten:

```
sudo apt-get install nut
```

Der NUT-Monitor-Service wird anschließend vergeblich versuchen zu starten, da der NUT-Server noch nicht konfiguriert ist.

### USV per USB ansteuern

Vergewissern Sie sich zunächst anhand der Hardware Compatibility List [4], welcher Treiber für Ihre USV der passende ist, und fügen Sie der Konfigurationsdatei "/etc/nut/ups.conf" eine Sektion für Ihre USV hinzu. Der String innerhalb der eckigen Klammern ist dabei ein frei wählbarer Name:

```
[UPS01]
  driver = usbhid-ups
  port = auto
  desc = "Back-UPS CS 650"
  vendorid = "051d"
  productid = "0002"
```

Falls mehrere USV mit identischen Vendor- und Product-IDs am NUT-Server angeschlossen sein sollten, verwenden Sie in der Konfiguration alternativ die Seriennummer:

```
[...]
  desc = "Back-UPS CS650"
  serial = <Ihre-Seriennummer>
```

Nun starten Sie den USV-Treiber mittels `sudo upsdrvctl start`

Der Befehl sollte darauf antworten mit:

```
Network UPS Tools - UPS driver controller 2.7.4
Network UPS Tools - Generic HID driver 0.41 (2.7.4)
USB communication driver 0.33
Using subdriver: APC HID 0.96
```

### Startschwierigkeiten umschiffen

Falls der Befehl stattdessen den Startversuch mit einer Fehlermeldung der Form "Can't claim USB device could not detach kernel driver from interface 0: Operation not permitted Driver failed

to start (exit status=1)" quittiert, deutet dies auf fehlende Zugriffsrechte hin. Sie benötigen dann eine udev-Regel, die der Gruppe "nut" Zugriff auf das USB-Gerät erteilt. Erzeugen Sie dazu die Datei "/etc/udev/rules.d/90-nut-ups.rules" mit folgendem Inhalt:

```
ACTION=="add", SUBSYSTEM=="usb",
  ATTR{idVendor}=="051d", ATTR{id-Product}=="0002", MODE="0660",
  GROUP="nut"
```

Lesen Sie das udev-Regelwerk neu ein:

```
sudo udevadm control --reload
```

```
sudo udevadm trigger
```

Anschließend sollte der Start des Treibers per `sudo upsdrvctl start` gelingen.

### Netzwerk und Benutzer konfigurieren

Weiter geht es dann mit upsd. Ändern Sie in der Datei "/etc/nut/nut.conf" die letzte Zeile zu "MODE=netserver". In der Datei "/etc/nut/upsd.conf" fügen Sie am Ende die beiden folgenden Zeilen ein:

```
LISTEN 127.0.0.1 3493
LISTEN <IP-des-Raspi> 3493
```

Dies sorgt dafür, dass der Server unter der lokalen Loopback-Adresse und seiner IP-Adresse im Netzwerk über den Port TCP/3493 erreichbar ist. Auf die für den produktiven Einsatz empfehlenswerte Absicherung der Kommunikation mittels

SSL und TCP-Wrappers verzichten wir in unserem Testaufbau. Hierbei hilft die NUT-Dokumentation unter [5]. Nun fehlen noch Benutzerkonten, die der NUT-Monitor für den Verbindungsaufbau benötigt. Fügen Sie am Ende der Datei "/etc/nut/upsd.users" noch die folgenden Sektionen hinzu:

```
[localuser]
  password = <Passwort 1>
  upsmon master

[remoteuser]
  password = <Passwort 2>
  upsmon slave
```

Die Strings in den eckigen Klammern sind wiederum frei wählbar. Sie verfügen damit über zwei User. Den einen werden wir im Folgenden lokal auf dem Raspi verwenden, der als Master agiert. Weitere Clients verwenden den zweiten User zum Zugriff über das Netzwerk als Slave. Konfigurieren Sie abschließend upsd für den automatischen Start und setzen Sie den Dienst in Gang:

```
sudo systemctl enable nut-server.service

sudo systemctl start nut-server.service
```

### USV per Shell auslesen

Nun können Sie sich bereits mit dem NUT-Client davon überzeugen, ob der Server funktioniert: `upsc UPS01@localhost`. Das Kommando liefert eine Liste sämtli-

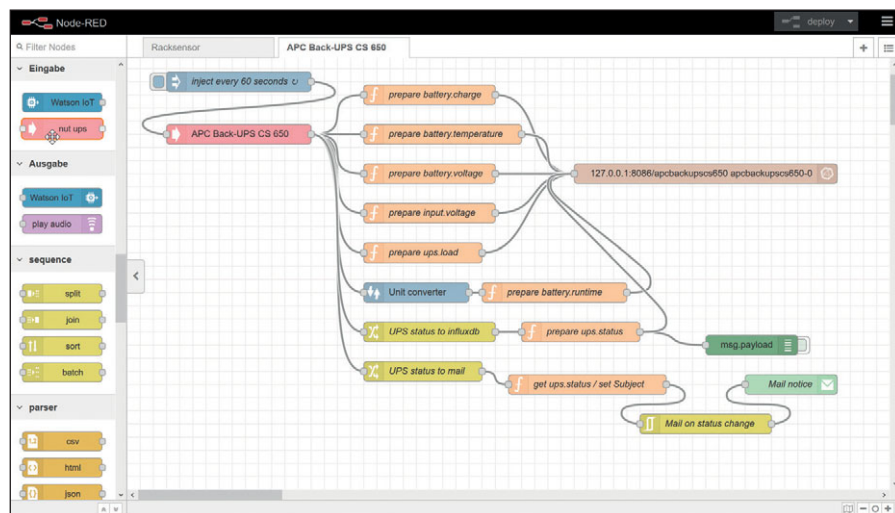


Bild 1: Der Node-RED-Flow bereitet Messwerte der USV auf und schreibt sie in eine Datenbank.

cher Parameter Ihrer USV zurück. Nützlich sind hier etwa die Ladung der Batterie (battery.charge), deren Restlaufzeit in Sekunden (battery.runtime) und der Status (ups.status). Im Regelbetrieb weist Letzterer den Wert "OL", also online, aus.

Sobald der Strom ausfällt, wechselt der Status zu "OB DISCHRG" (on battery, discharging). Kehrt der Strom zurück, weist der Status "OL CHRG" (online, charging) aus, bis er schließlich wieder zu "OL" wechselt, sobald die Batterie vollgeladen ist. Doch falls die Stromversorgung nicht rechtzeitig wiederhergestellt werden sollte, geht der Status zu "OB LB" (on battery, low battery) über. Und dies ist dann für upsmon auf den Slaves sowie anschließend auch auf dem Master das Signal, die Systeme herunterzufahren.

### upsmon auf Master und Slave

Ergänzen Sie zur Inbetriebnahme von upsmon in der "/etc/nut/upsmon.conf" die Zeile

```
MONITOR UPS01@localhost 1 localuser
  <Passwort 1> master
```

und starten Sie den Daemon mit

```
sudo systemctl enable
nut-monitor.service
sudo systemctl start
nut-monitor.service
```

Auf einem Slave unter Linux installieren Sie die Software ebenso:

```
sudo apt-get install nut
```

Die Konfiguration erfordert aber deutlich weniger Handgriffe. Ändern Sie in der Datei "/etc/nut/nut.conf" die letzte Zeile zu `MODE=netclient`. In "/etc/nut/upsmon.conf" sorgt die Zeile

```
MONITOR UPS01@<IP-des-Raspi> 1
  remoteuser <Passwort 2> slave
```

dafür, dass upsmon über das Netzwerk mit upsd kommuniziert. Setzen Sie nun auch hier den Daemon in Gang:

```
sudo systemctl enable nut-
monitor.service
```



Bild 2: Grafana visualisiert den Betriebszustand der USV übersichtlich in einem Dashboard.

```
sudo systemctl start nut-
monitor.service
```

### Windows-Client nur mit Workaround

Das NUT-Projekt hat die Software auch auf Windows portiert und stellt ein passendes MSI-Paket zur Installation bereit [6]. Die Version verharrt allerdings leider seit Jahren im Beta-Stadium und lässt sich nur mit einem Workaround für einen bekannten Fehler zur Arbeit überreden [7].

Installieren Sie das MSI-Paket und deaktivieren Sie die Option "Install libUSB driver", da wir das Windows-System nur als Slave konfigurieren wollen. NUT registriert sich daraufhin als Dienst für den automatischen Start, startet aber nicht sofort. Laden Sie nun ein ZIP-Archiv herunter, das fehlende Bibliotheken enthält [8].

Kopieren Sie die beiden Dateien libeay32.dll sowie sslay32.dll aus dem Archiv nach "C:\Program Files (x86)\NUT\sbin". Weiterhin kopieren Sie die Datei libgcc\_s\_dw2-1.dll aus "C:\Program Files (x86)\NUT\bin" ebenfalls nach ""C:\Program Files (x86)\NUT\sbin". Damit ist NUT startklar.

Die weitere Konfiguration erledigen Sie ähnlich der Linux-Variante unter "C:\Program Files (x86)\NUT\etc". Hier benennen Sie die "nut.conf.sample" zu "nut.conf" um und passen die letzte Zeile an:

```
MODE=netclient
```

Weiterhin erzeugen Sie aus der Datei "upsmon.conf.sample" die "upsmon.conf". Hier müssen Sie neben dem Ziel für den Shutdown-Kommando für Linux gegen die passende Variante für Windows austauschen:

```
MONITOR UPS01@<IP-Adresse des Raspi>
  1 remoteuser <Passwort 2> slave
(...)
SHUTDOWNCMD
  "C:\\WINDOWS\\system32\\shutdown.
  exe -s -t 0"
```

Abschließend starten Sie über die Computerverwaltung den Dienst "Network UPS Tools", der sich daraufhin im Ereignisprotokoll "Anwendung" als funktionsfähig meldet.

### Den Ernstfall proben

Damit verfügen wir nun über einen Master mit zwei Slaves und die Stunde der Wahrheit ist gekommen. Trennen Sie die USV vom Stromnetz und warten Sie darauf, dass NUT die Shutdown-Prozedur einleitet [9]. Die beginnt, sobald die USV zum Status "OB LB" wechselt. Wann dies der Fall ist, hängt vom Modell der USV ab. Das hier verwendete Gerät von APC wartet, bis der Parameter "battery.charge.low", also zehn Prozent Restkapazität der Batterie, erreicht ist.

Die Slaves erhalten dann vom Master die Anweisung herunterzufahren, was sie nach einer mittels des Parameters FINALDELAY in der upsmon.conf definierten Frist auch tun werden. Der Standardwert für diese Frist ist fünf Sekunden. Der Master wartet zusätzlich die im Parameter HOSTSYNC konfigurierte Zeitspanne, typischerweise weitere 15 Sekunden, bevor auch er herunterfährt und als letzte Amtshandlung die USV selbst abschaltet.

Durch Ändern der Parameter FINALDELAY sowie HOSTSYNC passen Sie das Ganze nach Ihren Wünschen an. Noch deutlich flexibler erweist sich der Einsatz von upsched, falls Sie nicht bis zum Status "OB LB" warten oder zusätzliche Kommandos skripten möchten [10].

### Grafisches Monitoring

Bis hierher erwies sich die Arbeit mit NUT als sehr textlastig und auch die integrierte Logging-Funktion mittels upslog wendet dies leider nicht zum Besseren, da NUT seine Status-Informationen von Haus aus lediglich in Textdateien schreibt. Wer Monitoring und Alarmierung komfortabler gestalten möchte, bindet NUT an die Workflow-Engine Node-RED an, schreibt die Messwerte der USV von dort in die Zeitreihen-Datenbank InfluxDB und visualisiert das Ganze mittels Grafana.

Nebenbei verschickt Node-RED auch noch E-Mails, sobald sich der Online-Status der USV verändert. Im Folgenden nehmen wir an, dass die drei Open-Source-Produkte bereits grundsätzlich eingerichtet sind. Wie dies gelingt, hatten wir in unserem Workshop zum Internet der Dinge erläutert [11].

### Workflow-Engine anbinden

Nun wollen wir Node-RED für die Anbindung von NUT ertüchtigen. Installieren Sie dazu das Node-RED-Modul für NUT und auch den Einheiten-Rechner sowie die E-Mail-Erweiterung, falls noch nicht vorhanden:

```
cd /home/pi/.node-red/node_modules
npm install node-red-contrib-nut-ups
```

```
npm install node-red-contrib-unit-converter
npm install node-red-node-email
npm audit fix
```

Daraufhin starten Sie Node-RED neu:

```
sudo systemctl restart nodered.service
```

Nun legen Sie eine leere Datenbank an, die später die Messwerte der USV aufnehmen wird:

```
sudo influx -execute 'CREATE DATABASE apcbackupscs650'
```

### Einen Flow aufbauen

So gerüstet, bauen Sie nun im Frontend von Node-RED einen Flow auf, der Werte der USV abfragt und in die Datenbank befördert. Eine fertige Vorlage für einen Flow sowie ein passendes Grafana-Dashboard zum Import finden Sie unter [12]. Der Flow verwendet einen Knoten vom Typ "inject" als Zeitgeber, der periodisch – in diesem Beispiel minütlich – einen Eingabeknoten vom Typ "nut ups" triggert (Bild 1). Letzterer stellt dann sämtliche Werte bereit, die Sie auch per upsc in der Linux-Shell abfragen können.

Ein Knoten vom Typ "Funktion\ function" sorgt jeweils dafür, den empfangenen Wert für die Datenbank aufzubereiten. Im Falle der Restlaufzeit wandelt zuvor noch die Funktion "unit converter" Sekunden in Minuten um. Den Status der USV wertet der Flow gleich zweimal aus. Zum einen landet dieser übersetzt auf einen Zahlenwert in der Datenbank, zum anderen reagiert Node-RED auf jede Änderung und verschickt eine E-Mail an die im Knoten mit dem Namen "Mail notice" konfigurierte Ziel-Adresse.


### Mit Grafana visualisieren

Vorausgesetzt, dass auch in Grafana eine Verbindung zur Datenbank mit den Messwerten der USV konfiguriert ist, importieren Sie das Dashboard und gelangen so zu einer grafischen Auswertung, mit der Sie den Zustand Ihrer USV stets im Blick haben (Bild 2). Das Dashboard

wandelt im ersten Panel oben links den Zahlenwert des USV-Status zurück in Text mitsamt passender farblicher Darstellung. Alle weiteren Werte sind als Panel vom Typ "Gauge" konfiguriert.

### Fazit

Die Server mit USVs vor Stromausfällen abzusichern, ist eine sinnvolle Sache. Mit dem einmaligen Installieren entsprechender Systeme ist es allerdings nicht getan. Vielmehr wollen diese auch überwacht sowie für Ihre Umgebung passend konfiguriert werden. Und für den Fall der Fälle müssen Sie das Prozedere vorab eindeutig festlegen.

Mit den Network UPS Tools binden Sie mehrere Slaves an Ihre Notstromversorgung an und sorgen so dafür, dass Ihre Systeme im Ernstfall geordnet schlafen gehen. Dabei behalten Sie per Node-RED und Grafana grafisch aufbereitet stets den Zustand Ihrer USV im Blick. Das Einrichten geht dabei leicht von der Hand. (dr) 

### Link-Codes

- [1] **Network UPS Tools**  
k7z01
- [2] **NUT-Integrationen bei NAS-Herstellern**  
k7z02
- [3] **Raspbian Buster Lite**  
hbp51
- [4] **Hardware Compatibility List**  
k7z04
- [5] **NUT-Dokumentation**  
k7z05
- [6] **MSI-Paket**  
k7z06
- [7] **Workaround für Windows-Client**  
k7z07
- [8] **Archiv mit fehlenden Bibliotheken**  
k7z08
- [9] **Shutdown-Prozedur**  
k7z09
- [10] **Verwenden von upsched**  
k7z00
- [11] **Workshop: Bestandteile und Aufbau eines IoT, IT-Administrator 09/2019**  
k7z0a
- [12] **Flow-Vorlage und passendes Grafana-Dashboard**  
k7z0b