Deutsch: Integration von Victron Energy mit Node-RED und Amazon Alexa

Dieses Dokument beschreibt die Integration eines Victron-Systems (Venus OS / Cerbo GX) mit Amazon Alexa über einen Cloudflare-Tunnel und Node-RED.

Zugriffsvoraussetzung: Um diese Schritte auszuführen, müssen Sie den Vollzugriff auf Ihrem Victron-Gerät aktivieren:

1. Gehen Sie zu **Settings > General > Access Level**.
2. Ändern Sie die Ebene auf **Superuser**. Der Superuser-Status wird erreicht, indem Sie die Benutzertaste einige Sekunden lang gedrückt halten.
3. Aktivieren Sie im selben Menü **SSH on LAN**. Ohne dies lässt das Terminal keine Verbindung zu; aktualisieren Sie auf ein eigenes Passwort für den Terminal-Zugriff.

Speicherhinweis: Die gesamte Software (cloudflared), Skripte und Logs müssen auf einer **SD-Speicherkarte** im FAT32-Format liegen, die normalerweise unter dem Pfad /media/IHR-SD-KARTE/ eingebunden ist. Befehle werden über ein SSH-Terminal ausgeführt. Alle Befehle und Programmierungen können kopiert und eingefügt werden; Sie müssen lediglich bestimmte Parameter in den Node-RED-Knoten entsprechend Ihrer Geräte und Batterien ändern.

1. Cloudflare-Tunnel-Konfiguration

1. Gehen Sie im **Cloudflare Dashboard** zu **Zero Trust > Networks > Tunnels**.
2. Erstellen Sie einen neuen Tunnel (z. B. "victron").
3. Im Reiter **Public Hostname**:
   - **Subdomain:** victron
   - **Domain:** Ihre registrierte Domain.
   - **Service:** http://localhost:1880
4. Kopieren Sie das bereitgestellte **Token** für das Start-Skript.

2. Konfiguration am Gerät (SSH-Terminal)

Erstellen Sie das Steuerungs-Skript auf der SD-Karte, um eine Sättigung des Systems durch doppelte Prozesse zu vermeiden.

- **Datei:** /media/mmcblk0p1/start_tunnel.sh
- **Befehl zum Erstellen der Datei:**

```
printf '#!/bin/sh

if ps | grep -v grep | grep "cloudflared tunnel" > /dev/null; then
    exit 0
fi
echo "nameserver 1.1.1.1" > /etc/resolv.conf
echo "nameserver 8.8.8.8" >> /etc/resolv.conf
```

```
for i in 1 2 3 4 5 6; do
    if ping -c 1 1.1.1.1 > /dev/null 2>&1; then
        break
    fi
    sleep 5
done

nohup /run/media/mmcblk0p1/cloudflared tunnel --protocol http2 --heartbeat-
interval 10s --heartbeat-count 3 run --token <TU_TOKEN_AQUI> > /run/media/
mmcblk0p1/tunnel.log 2>&1 &
' > /run/media/mmcblk0p1/start_tunnel.sh
```

- **Ausführungsberechtigungen:** `chmod +x /run/media/mmcblk0p1/start_tunnel.sh`

Automatisierung nach Neustart Um den Tunnel „unzerstörbar“ zu machen, verwenden wir Crontab. Das Skript ist jetzt intelligent: Wenn der Tunnel in Ordnung ist, passiert nichts; wenn er unterbrochen wurde, wird er neu gestartet. Führen Sie crontab -e aus und fügen Sie am Ende diese Zeilen hinzu:

```
@reboot /bin/sh /media/mmcblk0p1/start_tunnel.sh

** * * * /bin/sh /media/mmcblk0p1/start_tunnel.sh
```

3. Amazon Alexa Konfiguration (Developer Console)

Rufen Sie **developer.amazon.com** auf, um den Skill zu konfigurieren: **A. Skill erstellen**

- **Create Skill > Name:** Victron Casa.
- **Default Language:** Wählen Sie Ihre bevorzugte Sprache (z. B. German (DE)).
- **Model:** Wählen Sie **Custom**. **Hosting:** Wählen Sie **Provision your own**.

B. Interaction Model (Intents) Konfigurieren Sie die Aktivierungsphrasen:

- BateriaIntent: "Wie ist der Batteriestatus?", "Batterie", "Ladestand".
- SolarIntent: "Wie viel produzieren die Paneele?", "Solarproduktion", "Solarleistung".
- **Wichtig:** Klicken Sie nach dem Speichern auf "Build Model".

C. Endpoint

- **Endpoint > HTTPS auswählen.**
- **Default Region:** https://ihre-domain.de/api/alexa (Cloudflare URL).
- **SSL Certificate:** Wählen Sie: "My endpoint is a sub-domain that has a wildcard certificate...".

4. Logik in Node-RED

Importieren Sie das im Handbuch enthaltene JSON. Sie können die Antworttexte von Alexa im Knoten „Generate Smart Response“ anpassen.


```
[{"id":"3240ada185f2413b","type":"tab","label":"Alexa Victron"},
{"id":"f27a94492211a6c9","type":"function","z":"3240ada185f2413b","name":"Genera
te Smart Response","func":"const soc = flow.get('soc') || \"not available\";
\nconst pv = flow.get('pv') || 0;\nconst intentName = (msg.payload.request &&
msg.payload.request.intent) ? msg.payload.request.intent.name :
\"LaunchRequest\";\nlet speak = \"\";\nswitch (intentName) {\n    case
\"BateriaIntent\":\n        speak = `The battery charge level is ${soc}
percent.`;\n        break;\n    case \"SolarIntent\":\n        speak = `The
solar panels are generating ${pv} watts at this moment.`;\n        break;\n
default:\n        speak = `System ready. The battery is at ${soc} percent and
solar production is ${pv} watts.`;\n}\nmsg.payload = { version: '1.0', response:
{ outputSpeech: { type: 'PlainText', text: speak }, shouldEndSession: true } };
\nreturn msg;","outputs":1,"x":450,"y":320,"wires":[["6baa9a22615b9c3f"]]},
{"id":"d8eacd0659e971f8","type":"http in","z":"3240ada185f2413b","name":"Alexa
Endpoint","url":"/api/alexa","method":"post","x":160,"y":300,"wires":
[["f27a94492211a6c9"]]},{"id":"6baa9a22615b9c3f","type":"http
response","z":"3240ada185f2413b","name":"HTTP
200","statusCode":"200","x":680,"y":320,"wires":[]},
{"id":"a77968dacbf0c65e","type":"function","z":"3240ada185f2413b","name":"Save
SOC","func":"flow.set('soc', msg.payload);
\nnode.status({fill:'blue',shape:'dot',text:`SOC ${msg.payload}%`});\nreturn
null;","outputs":1,"x":540,"y":140,"wires":[[]]},
{"id":"8c6a9162d6321944","type":"function","z":"3240ada185f2413b","name":"Save
PV","func":"flow.set('pv', msg.payload);
\nnode.status({fill:'green',shape:'dot',text:`PV ${msg.payload} W`});\nreturn
null;","outputs":1,"x":530,"y":220,"wires":[[]]},
{"id":"5b162252b72956d8","type":"victron-input-
battery","z":"3240ada185f2413b","service":"com.victronenergy.battery/
512","path":"/Soc","serviceObj":{"service":"com.victronenergy.battery/
512","name":"RUiXU RX Battery","communityTag":"battery"},"pathObj":{"path":"/
Soc","type":"float","name":"State of charge (%)"},"name":"SOC
Battery","onlyChanges":false,"roundValues":"1","rateLimit":0,"outputs":1,"condit
ionalMode":false,"outputTrue":"true","outputFalse":"false","debounce":"2000","x"
:330,"y":140,"wires":[["a77968dacbf0c65e"]]},
{"id":"ff40400a00ffbde9","type":"victron-input-
solarcharger","z":"3240ada185f2413b","service":"com.victronenergy.solarcharger/
279","path":"/Yield/Power","serviceObj":
{"service":"com.victronenergy.solarcharger/279","name":"SmartSolar Charger MPPT
150/45 rev3","communityTag":"solarcharger"},"pathObj":{"path":"/Yield/
Power","type":"float","name":"PV Power (W)"},"name":"Solar
Power","onlyChanges":false,"roundValues":"1","rateLimit":0,"outputs":1,"conditio
nalMode":false,"outputTrue":"true","outputFalse":"false","debounce":"2000","x":3
40,"y":220,"wires":[["8c6a9162d6321944"]]}]
```

5. Diagnose (SSH)

- ps | grep cloudflared: Prüft, ob der Prozess aktiv ist.
- tail -f /media/mmcblk0p1/tunnel.log: Überwacht die Verbindung.

Haftungsausschluss: Dieses Handbuch wurde entwickelt, um die Integration durch Benutzer mit Grundkenntnissen in der Programmierung zu erleichtern. Die Verwendung dieser Informationen und die Ausführung der bereitgestellten Befehle liegen in der alleinigen Verantwortung des Benutzers. Es wird keine Haftung für Hardwareschäden, Netzwerksicherheitsfehler oder Dienstunterbrechungen aufgrund unsachgemäßer Konfiguration übernommen.