English: Victron Energy Integration with Node-RED and Amazon Alexa

This document describes the integration of a Victron system (Venus OS / Cerbo GX) with Amazon Alexa using a Cloudflare tunnel and Node-RED.

Access Prerequisite: To execute these steps, you must enable full access on your Victron device:

1. Go to **Settings > General > Access Level**.
2. Change the level to **Superuser**. Superuser access is obtained by holding the user button pressed for a few seconds until the level changes.
3. In the same menu, activate **SSH on LAN**. Without this, the terminal will not allow the connection; update to your own password to access via terminal.

Storage Note: All software (cloudflared), scripts, and logs must reside on an **SD Memory Card** formatted in FAT32, usually mounted at the path /media/YOUR-SD-CARD/. Commands are executed via an SSH Terminal. All commands and programming can be copied and pasted to facilitate integration; you will only need to change certain parameters in the Node-RED nodes according to your equipment and batteries.

1. Cloudflare Tunnel Configuration

1. In the **Cloudflare Dashboard**, go to **Zero Trust > Networks > Tunnels**.
2. Create a new tunnel (e.g., "victron").
3. In the **Public Hostname** tab:
   - **Subdomain:** victron
   - **Domain:** Your registered domain.
   - **Service:** http://localhost:1880
   - Copy the provided **Token** for the startup script.

2. Local Device Configuration (SSH Terminal)

Create the control script on the SD card to ensure the system does not saturate with duplicate processes.

- **File:** /media/mmcblk0p1/start_tunnel.sh
- **Command to create the file:**

```
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
```

Execution permissions: `chmod +x /run/media/mmcblk0p1/start_tunnel.sh`

Automation After Reboot To make the tunnel "indestructible", we will use crontab. The script is now intelligent: if the tunnel is fine, it does nothing; if it dropped, it brings it up. Run crontab -e and add these lines at the end:

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

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

3. Amazon Alexa Configuration (Developer Console)

Access **developer.amazon.com** to configure the Skill: **A. Create the Skill**

- **Create Skill > Name:** Victron Casa.
- **Default Language:** Select your preferred language (e.g., English (US)).
- **Model:** Select **Custom**. **Hosting:** Select **Provision your own**.

B. Interaction Model (Intents) Configure the activation phrases:

- BateriaIntent: "what is the battery status?", "battery", "charge percentage".
- SolarIntent: "how much are the panels producing?", "solar production", "solar watts".
- **Important:** Click "Build Model" after saving.

C. Endpoint

- **Endpoint > Select HTTPS.**
- **Default Region:** https://your-domain.com/api/alexa (Cloudflare URL).
- **SSL Certificate:** Select: "My endpoint is a sub-domain that has a wildcard certificate...".

4. Node-RED Logic

Import the JSON provided in the manual. You can customize Alexa's response texts in the "Generate Smart Response" node.


```
[{"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. Diagnosis (SSH)

- ps | grep cloudflared: Verifies if the process is active.
- tail -f /media/mmcblk0p1/tunnel.log: Monitors the connection

Disclaimer: This manual is designed to facilitate integration for users with basic programming knowledge. The use of this information and the execution of the provided commands are the sole responsibility of the user. No liability is assumed for hardware damage, network security failures, or service interruptions due to improper configuration.