Home Assistant Integration via MQTT

Overview

Integrating Meshtastic into Home Assistant unlocks powerful home automation possibilities: track family members on a mesh map, get alerts when a node goes offline, and trigger smart-home actions based on mesh events. The integration uses MQTT as the transport layer, with Mosquitto as the local broker.

Step 1: Install and Configure Mosquitto

In Home Assistant, navigate to **Settings → Add-ons → Add-on Store** and install the *Mosquitto broker* add-on. After installation, open its configuration tab and add a user:

```
logins: - username: meshuser
 password: yourpassword
```

Start the add-on and enable *Start on boot*. Note your Home Assistant host IP - your Meshtastic node will connect to this address.

Step 2: Configure Your Meshtastic Node for MQTT

Open the Meshtastic app, go to **Radio Configuration → MQTT** and set:

- **MQTT Server Address:** your HA host IP (e.g. `192.168.1.100`)
- **Username / Password:** the credentials you set above
- **Root Topic:** `msh/US` (or your region)
- **Uplink / Downlink Enabled:** on

Also ensure your node has WiFi configured under **Radio Configuration → Network**.

Step 3: MQTT Sensor YAML in Home Assistant

Add the following to your `configuration.yaml` (adjust node IDs as needed):

```
mqtt:
 sensor: - name: "Node !a1b2c3d4 Battery"
 state_topic: "msh/US/2/json/telemetry/!a1b2c3d4"
 value_template: "{{ value_json.payload.device_metrics.battery_level }}"
 unit_of_measurement: "%"
 device_class: battery - name: "Node !a1b2c3d4 Latitude"
 state_topic: "msh/US/2/json/position/!a1b2c3d4"
 value_template: "{{ value_json.payload.latitude_i | float / 1e7 }}" - name: "Node !a1b2c3d4
Longitude"
 state_topic: "msh/US/2/json/position/!a1b2c3d4"
 value_template: "{{ value_json.payload.longitude_i | float / 1e7 }}"
```

Reload your YAML configuration after saving.

Step 4: Automations

Alert when a node hasn't been heard in 30 minutes:

```
automation: - alias: "Mesh node offline alert"
 trigger:
 platform: state
 entity_id: sensor.node_a1b2c3d4_battery
 to: unavailable
 for: "00:30:00"
 action:
 service: notify.mobile_app_your_phone
 data:
 message: "Node !a1b2c3d4 has not reported in 30 minutes"
```

Notify on critical battery:

```
 - alias: "Mesh node low battery"
 trigger:
```

```
  platform: numeric_state
  entity_id: sensor.node_a1b2c3d4_battery
  below: 15
  action:
  service: notify.mobile_app_your_phone
  data:
  message: "Node !a1b2c3d4 battery critical: {{ states('sensor.node_a1b2c3d4_battery') }}%"
```

Trigger lights when a family member arrives home via mesh position:

```
  - alias: "Welcome home via mesh"
  trigger:
  platform: zone
  entity_id: device_tracker.mesh_family_member
  zone: zone.home
  event: enter
  action:
  service: light.turn_on
  target:
  entity_id: light.porch
```

To use zone-based presence, create a `device_tracker` that updates from the latitude/longitude sensors using a template or the MQTT device tracker integration.

Step 5: Lovelace Dashboard

For a mesh map view, install the **Map card** in Lovelace. Add a card of type `map` and reference your device tracker entities. You can also use the *auto-entities* HACS card to dynamically list all mesh node sensors. For richer visualization, some community members use Grafana with the InfluxDB integration to feed in MQTT telemetry.

Revision #4
Created 2026-05-03 06:43:25 UTC by Mesh America Admin
Updated 2026-05-03 13:38:31 UTC by Mesh America Admin